aws

Guide de l’utilisateur

EC2 Image Builder

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

EC2 Image Builder: Guide de l’utilisateur

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques et la présentation commerciale d’Amazon ne peuvent être utilisées en relation avec un produit ou un service qui n’est pas d’Amazon, d’une manière susceptible de créer une confusion parmi 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 qu'Image Builder ? 1
Caractéristiques d'Image Builder 2
Systèmes d'exploitation pris en charge 4
Formats d'image pris en charge 4
Quotas par défaut 4
AWS Régions et points de terminaison 5
Concepts 5
Tarification 8
Connexe Services AWS 9
Comment fonctionne Image Builder 9
Éléments de l'AMI 10
Gestion des composants 11
Ressources créées 12
Distribution 12
Partage de ressources 13
Conformité d' 13
Gestion des versions sémantique 14
Gestion des versions génériques dans les politiques de cycle de vie 15
Utilisation des références de version 15
Modèles de référence des versions disponibles 16
Ressources qui renvoient des références de version 16
Préparez-vous 18
Rôle lié au service Image Builder 18
Exigences de configuration 19
Référentiel de conteneurs pour les pipelines d'images de conteneurs 20
Hôte dédié pour les images macOS 20
Conditions préalables à l'IAM 20
Conditions préalables requises pour l'agent Systems Manager 22
Tutoriels Image Builder 23
Créez votre première image 23
Création d'un composant personnalisé avec des paramètres d'entrée 24
Utiliser les paramètres de Systems Manager avec Image Builder 24
Déployez des exemples de projets avec une infrastructure en tant que code 24
Assistant de pipeline : création d'une AMI 24

Étape 1 : Spécifier les détails du pipeline 25
Étape 2 : Choisissez la recette 26
Étape 3 : Définition de la configuration de l'infrastructure (facultatif) 28
Étape 4 : définir les paramètres de distribution (facultatif) 29
Étape 5 : Vérifier 30
Étape 6 : Nettoyer 30
Assistant de pipeline : création d'une image de conteneur 32
Étape 1 : Spécifier les détails du pipeline 33
Étape 2 : Choisissez la recette 33
Étape 3 : Définition de la configuration de l'infrastructure (facultatif) 37
Étape 4 : définir les paramètres de distribution (facultatif) 38
Étape 5 : Vérifier 38
Étape 6 : Nettoyer 39
Composant personnalisé avec paramètres 41
Utiliser des paramètres dans le document de votre composant YAML 41
Définissez les paramètres des composants dans une recette Image Builder depuis la console 46
Utilisez un paramètre d'image de base dans votre recette 47
Étape 1 : rechercher ou créer un paramètre de magasin de paramètres 47
Étape 2 : Configuration des autorisations IAM (facultatif) 48
Étape 3 : Création d'une recette d'image utilisant le paramètre 50
Exemples de projets sur GitHub 50
Éléments 52
Répertorier et afficher les composants 54
Liste des composants d'Image Builder 55
Répertorier les versions de compilation des composants à partir du AWS CLI 58
Obtenez des informations sur les composants à partir du AWS CLI 58
Obtenez des informations détaillées sur la politique relative aux composants à partir du AWS CLI 59
Utiliser des AWS Marketplace composants 59
Découvrez AWS Marketplace components 60
Abonnez-vous à AWS Marketplace components 62
Utiliser AWS Marketplace des composants dans une recette 62
Utiliser des composants gérés 63
Installation de l'application Windows dans le package du distributeur 64
Durcissement CIS 69

Composants durcissants STIG 70
Développez des composants personnalisés 118
Création d'un document de composant YAML 119
Créez des composants personnalisés avec Image Builder 123
AWSTOE application de gestion de composants 134
AWSTOE downloads 135
Régions prises en charge 137
Référence des commandes 139
Configuration manuelle 146
Utiliser le cadre documentaire des composants 159
Modules d'action 215
Configurer l'entrée 331
Ressources relatives aux images 336
Répertorier les images et créer des versions 336
Liste des images 337
Répertorier les images en attente d'action 343
Lister les versions de génération d'images 345
Afficher les détails des ressources d'images 348
Afficher les détails de l'image dans la console Image Builder 349
Pour plus d'informations sur la politique relative aux images, consultez le AWS CLI 356
Créez des images personnalisées avec Image Builder 357
Annuler la création d'image à partir du AWS CLI 359
Importation et exportation d'images de machines virtuelles 360
Importer une machine virtuelle dans Image Builder 360
Distribuez des disques de machine virtuelle à partir de votre image créée à partir du AWS CLI 366
Importer des images de disque ISO 366
Considérations de sécurité pour l'importation d'images de disque ISO 366
Systèmes d'exploitation pris en charge pour l'importation d'images disque ISO 367
Conditions préalables 367
Paramètres d'importation facultatifs 370
Importer une image disque ISO dans Image Builder 371
Lancer une instance depuis l'AMI de sortie 376
Configuration réseau minimale requise pour un VPC privé 377
Étapes suivantes 378
Résoudre les problèmes liés aux importations d'images de disque ISO 378

Gérez les résultats de sécurité ... 379
Configuration des scans de sécurité ... 380
Gérez les résultats de sécurité ... 382
Nettoyez les ressources d'Image Builder ... 384
Gérer le cycle de vie des images ... 385
Conditions préalables ... 387
Création d'un rôle IAM pour la gestion du cycle de vie d'Image Builder ... 387
Création d'un rôle IAM pour la gestion du cycle de vie entre comptes d'Image Builder ... 389
Répertorier les politiques de ... 390
Afficher les détails de la politique ... 393
Afficher les détails de la politique de cycle de vie dans la console Image Builder ... 393
Créez des politiques de cycle de vie ... 396
Création de politiques de cycle de vie (image AMI) ... 396
Création de politiques de cycle de vie (image du conteneur) ... 401
Comment fonctionne l'exécution ... 405
Calendrier d'exécution ... 406
Statuts d'exécution ... 406
Comment les ressources sont évaluées ... 407
Comment fonctionne la rétention pour les règles de suppression ... 408
Gestion manuelle de l'état des ressources ... 409
Annulation d'une exécution ... 409
Comment les politiques de cycle de vie gèrent les ressources modifiées de l'extérieur ... 409
Surveillance des résultats d'exécution ... 410
Comment fonctionnent les règles du cycle de vie ... 411
Priorité des règles et ordre d'évaluation ... 412
Règles d'exclusion du cycle de vie des AMI ... 413
Les balises en tant que sélecteurs de ressources ou règles d'exclusion ... 415
Comment est calculée la rétention ... 417
Comment les actions du cycle de vie affectent les ressources associées ... 418
Transitions d'état des ressources autorisées ... 419
Afficher le détail des règles de gestion du cycle de vie ... 420
Configuration d'images personnalisées ... 421
Recettes ... 421
Comparaison des recettes avec images et des recettes en contenants ... 423
Versionnage des recettes ... 424
Contraintes et limites des recettes ... 425

Répertorier et afficher des recettes illustrées 426
Répertorier et consulter les recettes de contenants 430
Création d'une nouvelle version d'une recette imagée 433
Création d'une nouvelle version d'une recette de contenant 453
Suivez la lignée des AMI à l'aide de filigranes 467
nettoyer des ressources ; 471
Configurations d'infrastructure 471
Placement et location 474
Répertorier et afficher les configurations d'infrastructure 474
Créer une configuration d'infrastructure 476
Mettre à jour une configuration d'infrastructure 483
Supprime une configuration d'infrastructure 486
AWS PrivateLink Points de terminaison VPC 486
Paramètres de distribution 492
Répertorier et afficher les configurations de distribution 495
Création et mise à jour de la distribution AMI 496
Création et mise à jour de la distribution d'images de conteneurs 512
Configurer la distribution d'AMI entre comptes 516
Spécifier un modèle de lancement d'AMI 525
Utiliser une distribution AMI améliorée 528
Partagez des ressources 530
Propriétaires des ressources 531
Conditions préalables au partage des ressources Image Builder 531
Consommateurs de ressources 532
Création d'un partage de ressources 532
Option 1 : créer un partage de ressources RAM 533
Option 2 : appliquer une politique de ressources et promouvoir vers un partage de ressources existant 533
Annuler le partage d'une ressource 536
Balisage des ressources 537
Marquez une ressource depuis AWS CLI 537
Supprimer le tag d'une ressource du AWS CLI 538
Répertoriez toutes les balises d'une ressource spécifique à partir du AWS CLI 538
Supprimer des ressources 540
Console : Supprimer des ressources 540
Supprimer des ressources (AWS CLI) 542

Flux de travail liés 544
Structure du flux de travail : étapes 545
Fonctionnement d'un flux de travail 547
Surveiller l'exécution des flux de 548
Contraintes liées aux ressources du flux 548
Accès au service 549
Utiliser des flux de travail gérés 550
Répertorier les flux de 551
Création d'un flux de travail d'images 555
Création d'un document de flux de travail YAML 559
Structure d'un document de flux de travail YAML 559
Actions par étapes 573
Variables dynamiques 629
Instructions conditionnelles 635
Gérer les pipelines 641
Configuration de l'exécution du pipeline 642
Désactiver automatiquement un pipeline défaillant 642
Configurer la journalisation du pipeline 643
Exécuter le pipeline manuellement 644
Utiliser des expressions cron 645
Répertorier et afficher les pipelines 651
Répertorier les pipelines d'images à partir du AWS CLI 651
Obtenez des informations détaillées sur le pipeline d'images à partir du AWS CLI 651
Création et mise à jour de pipelines (AMI) 652
Créez un pipeline AMI à partir du AWS CLI 653
Mettre à jour le pipeline depuis la console 658
Mettre à jour le pipeline à partir du AWS CLI 661
Création et mise à jour de pipelines (conteneur) 663
Créez un pipeline à partir du AWS CLI 664
Mettre à jour le pipeline depuis la console 668
Mettre à jour le pipeline à partir du AWS CLI 672
Configuration des flux de travail du pipeline 674
Définissez des groupes de test pour les flux de travail de test 675
Définissez les paramètres du flux de travail dans un pipeline Image Builder depuis la console 679

Spécifiez le rôle de service IAM qu'Image Builder utilise pour exécuter des actions de flux de travail ... 679
Associer des flux de travail à un pipeline ... 680
Utiliser EventBridge ... 682
EventBridge termes ... 683
Afficher EventBridge les règles de votre pipeline Image Builder ... 684
Déclenchez la création d'un pipeline selon un calendrier ... 685
Déclencher la création d'un pipeline à partir d'un modèle d'événement ... 688
Réagissez aux événements envoyés par Image Builder ... 690
Intégrer des produits et des services ... 693
Amazon EventBridge ... 696
Messages d'événements envoyés par Image Builder ... 698
Amazon Inspector ... 703
AWS Marketplace ... 705
AWS Marketplace abonnements dans Image Builder ... 706
Découvrez AWS Marketplace produits d'image depuis la console Image Builder ... 707
Utilisez un AWS Marketplace produit d'image dans les recettes Image Builder ... 709
Amazon Simple Notification Service ... 711
Rubriques SNS cryptées ... 711
Format de message SNS ... 713
Produits de conformité ... 718
Surveiller les événements et les journaux ... 720
CloudTrail journaux ... 720
Informations sur Image Builder dans CloudTrail ... 720
CloudWatch Journaux ... 722
Sécurité dans Image Builder ... 724
Protection des données ... 725
Votre responsabilité en ce qui concerne les images de disque ISO importées ... 726
Chiffrement et gestion des clés ... 726
Stockage de données ... 732
Inter-network Confidentialité du trafic ... 733
Gestion de l'identité et des accès ... 733
Public ciblé ... 733
Authentification par des identités ... 734
Comment Image Builder fonctionne avec les politiques et les rôles IAM ... 734
Gérez les périmètres de données ... 745

politiques basées sur l'identité 746
Autorisations de flux de travail personnalisées 749
Politiques basées sur une ressource 751
Politiques gérées 752
Rôles liés à un service 773
Résolution des problèmes 775
Validation de conformité 778
Résilience 778
Sécurité de l'infrastructure 779
Gestion des correctifs 780
Bonnes pratiques 782
Nettoyage requis après la construction 783
Remplacez le script de nettoyage de Linux 798
Ignorer le script de nettoyage de macOS 802
Résoudre les problèmes liés à Image Builder 806
Résoudre les problèmes liés aux builds de pipelines 806
Scénarios de résolution des problèmes 808
Résoudre les problèmes avec l'IA 813
Conditions préalables 814
Configurer la résolution des problèmes 815
Résoudre les problèmes liés à une compilation qui a échoué 815
Résoudre les problèmes liés à plusieurs pipelines défaillants 816
Ce que Troubleshoot étudie 817
Considérations 817
Résoudre les problèmes à l'aide d'un agent de codage IA 818
Acquérez la compétence 819
Considérations 819
Historique de la documentation 820
..... dcccxlvi

Qu'est-ce qu'Image Builder ?

EC2 Image Builder est une solution entièrement Service AWS gérée qui vous permet d'automatiser la création, la gestion et le déploiement d'images personnalisées, sécurisées up-to-date et de serveur. Vous pouvez utiliser le Console de gestion AWS AWS Command Line Interface, ou APIs pour créer des images personnalisées dans votre Compte AWS.

Vous êtes propriétaire des images personnalisées créées par Image Builder dans votre compte. Vous pouvez configurer des pipelines pour automatiser les mises à jour et l'application de correctifs au système pour les images que vous possédez. Vous pouvez également exécuter une commande autonome pour créer une image avec les ressources de configuration que vous avez définies.

L'assistant de pipeline Image Builder peut vous guider tout au long des étapes de création d'une image personnalisée, comme suit :

Étape 1 : Spécifier les détails du pipeline

- Donnez un nom à votre pipeline et ajoutez des balises.
- Définissez les métadonnées et les paramètres d'analyse des vulnérabilités.
- Établissez un calendrier pour votre pipeline.

Étape 2 : Personnalisez votre image

Vous pouvez sélectionner une recette existante ou en créer une nouvelle.

- Choisissez une image de base pour vos personnalisations.
- Ajoutez ou supprimez des logiciels de votre image de base.
- Personnalisez les paramètres et les scripts avec des composants de build.
- Sélectionnez les composants de test à exécuter.

Étape 3 : Définissez votre flux de travail

Un flux de travail d'image définit la séquence d'étapes qu'Image Builder exécute pendant les étapes de création et de test du processus de création d'image. Cela fait partie de l'infrastructure globale du flux de travail Image Builder.

Étape 4 : Configuration de l'infrastructure de construction

- Sélectionnez un rôle IAM à associer au profil d'instance pour les instances lancées par Image Builder pendant le processus de création de l'image.
- Sélectionnez un ou plusieurs types d'instances qui peuvent être appliqués au lancement.
- Sélectionnez une rubrique Amazon Simple Notification Service (SNS) pour recevoir des notifications d'Image Builder.
- Spécifiez le VPC, le sous-réseau et les groupes de sécurité qui s'appliquent au processus de création d'image.
- Sélectionnez les paramètres de résolution des problèmes, tels que l'endroit où Image Builder écrit les journaux, et s'il convient de mettre fin à l'instance de génération en cas d'échec (par défaut) ou de la maintenir en cours d'exécution pour un dépannage ultérieur.

Étape 5 : définir la distribution des images

- Sélectionnez l' Régions AWS endroit où Image Builder distribue votre Amazon Machine Image (AMI) ou votre image de conteneur.
- Si votre pipeline Image Builder crée une AMI, Image Builder prend également en charge la configuration suivante :
  - Sélectionnez une clé KMS à utiliser pour le chiffrement.
  - Configurez le partage d'AMI entre Comptes AWS et Organizations.
  - Associez une licence autogérée License Manager à votre image distribuée.
  - Configurez un modèle de lancement pour votre image.

Caractéristiques d'Image Builder

EC2 Image Builder fournit les fonctionnalités suivantes :

Augmentez la productivité et réduisez les opérations liées à la mise en conformité et à la création up-to-date d'images

Image Builder réduit le travail nécessaire à la création et à la gestion d'images à grande échelle en automatisant vos pipelines de création. Vous pouvez automatiser vos builds en indiquant vos préférences en matière de calendrier d'exécution des builds. L'automatisation réduit les coûts

opérationnels liés à la maintenance de vos logiciels avec les derniers correctifs du système d'exploitation.

Augmenter le temps de disponibilité des services

Image Builder donne accès à des composants de test que vous pouvez utiliser pour tester vos images avant le déploiement. Vous pouvez également créer des composants de test personnalisés avec AWS Task Orchestrator and Executor (AWSTOE) et les utiliser. Image Builder distribue votre image uniquement si tous les tests configurés ont réussi.

Rehaussez la barre de sécurité pour les déploiements

Image Builder vous permet de créer des images qui éliminent toute exposition inutile aux vulnérabilités de sécurité des composants. Vous pouvez appliquer des paramètres de AWS sécurité pour créer des out-of-the-box images sécurisées répondant aux critères de sécurité internes et industriels. Image Builder fournit également des ensembles de paramètres pour les entreprises des secteurs réglementés. Vous pouvez utiliser ces paramètres pour créer rapidement et facilement des images conformes aux normes STIG. Pour une liste complète des composants STIG disponibles via Image Builder, consultezAmazon a géré les composants de renforcement STIG pour Image Builder.

Application centralisée et suivi du lignage

Grâce aux intégrations intégrées à Image Builder AWS Organizations, vous pouvez appliquer des politiques qui empêchent les comptes d'exécuter des instances uniquement si elles sont approuvées AMIs.

Partage simplifié des ressources entre Comptes AWS

EC2 Image Builder s' AWS Resource Access Manager intègre àAWS RAM() pour vous permettre de partager certaines ressources avec Compte AWS n'importe qui ou AWS Organizations via. Les ressources EC2 Image Builder qui peuvent être partagées sont les suivantes :

- Éléments
- Images
- Recettes d'images
- Recettes de contenants

Pour de plus amples informations, veuillez consulter Partagez les ressources d'Image Builder avec AWS RAM.

Systèmes d'exploitation pris en charge

Image Builder prend en charge les versions de système d'exploitation suivantes :

Système d'exploitation/distribution	Versions prises en charge
Amazon Linux	2 et 2023
CentOS	7 et 8
CentOS Stream	8
macOS	12 x (Monterey), 13 x (Ventura), 14 x (Sonoma), 15 x (Sequoia), 26 x (Tahoe)
Red Hat Enterprise Linux (RHEL)	7, 8, 9 et 10
Serveur SUSE Linux Enterprise (SUSE)	12, 15 et 16
Ubuntu	18,04 LTS, 20,04 LTS, 22,04 LTS et 24,04 LTS
Windows Server	2012 R2, 2016, 2019, 2022 et 2025

Formats d'image pris en charge

Pour vos images personnalisées qui créent une Amazon Machine Image (AMI), vous pouvez choisir une AMI existante comme point de départ. Pour les images de conteneur Docker, vous pouvez choisir entre des images publiques hébergées sur DockerHub, des images de conteneur existantes dans Amazon ECR ou des images de conteneur gérées par Amazon comme point de départ.

Quotas par défaut

Pour consulter les quotas par défaut pour Image Builder, consultez la section Points de terminaison et quotas d'Image Builder.

AWS Régions et points de terminaison

Pour consulter les points de terminaison de service pour Image Builder, consultez la section Points de terminaison et quotas d'Image Builder.

Concepts

Les termes et concepts suivants sont essentiels à votre compréhension et à votre utilisation d'EC2 Image Builder.

AMI

L'Amazon Machine Image (AMI) est l'unité de base du déploiement dans Amazon EC2 et constitue l'un des types d'images que vous pouvez créer avec Image Builder. Une AMI est une image de machine virtuelle préconfigurée qui contient le système d'exploitation (OS) et le logiciel préinstallé pour déployer les instances EC2. Pour plus d'informations, consultez Amazon Machine Images (AMI).

Pipeline d'images

Un pipeline d'images fournit un cadre d'automatisation permettant de créer des images sécurisées AMIs et des images de conteneur AWS. Le pipeline d'images Image Builder est associé à une recette d'image ou à une recette de conteneur qui définit les phases de création, de validation et de test du cycle de vie d'une création d'image.

Un pipeline d'image peut être associé à une configuration d'infrastructure qui définit l'emplacement de création de votre image. Vous pouvez définir des attributs, tels que le type d'instance, les sous-réseaux, les groupes de sécurité, la journalisation et d'autres configurations liées à l'infrastructure. Vous pouvez également associer votre pipeline d'images à une configuration de distribution pour définir la manière dont vous souhaitez déployer votre image.

Image gérée

Une image gérée est une ressource d'Image Builder composée d'une AMI ou d'une image de conteneur, ainsi que de métadonnées, telles que la version et la plate-forme. L'image gérée est utilisée par les pipelines Image Builder pour déterminer l'image de base à utiliser pour la génération. Dans ce guide, les images gérées sont parfois appelées « images », mais une image n'est pas la même chose qu'une AMI.

Recette d'images

Une recette Image Builder est un document qui définit l’image de base et les composants à appliquer à l’image de base pour produire la configuration souhaitée pour l’image AMI en sortie. Vous pouvez utiliser une recette d'image pour dupliquer les builds. Les recettes d'images Image Builder peuvent être partagées, ramifiées et modifiées à l'aide de l'assistant de console, de l' AWS CLI API ou de l'API. Vous pouvez utiliser des recettes d'images avec votre logiciel de contrôle de version pour conserver des recettes d'images versionnées partageables.

Recette de contenant

Une recette de conteneur 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 de conteneur de sortie. Vous pouvez utiliser une recette de conteneur pour dupliquer des builds. Vous pouvez partager, associer et modifier des recettes d'images Image Builder à l'aide de l'assistant de console, de l' AWS CLI API ou de l'API. Vous pouvez utiliser des recettes de conteneurs avec votre logiciel de contrôle de version pour gérer des recettes de conteneurs versionnées et partageables.

Image de base

L'image de base est l'image sélectionnée et le système d'exploitation utilisés dans votre image ou votre document de recette de conteneur, ainsi que les composants. L'image de base et les définitions de composants combinées produisent la configuration souhaitée pour l'image de sortie.

Éléments

Un composant définit la séquence d'étapes requises pour personnaliser une instance avant la création de l'image (composant de génération) ou pour tester une instance lancée à partir de l'image créée (composant de test).

Un composant est créé à partir d'un document YAML ou JSON déclaratif en texte brut qui décrit la configuration d'exécution pour créer et valider, ou tester une instance produite par votre pipeline. Les composants s'exécutent sur l'instance à l'aide d'une application de gestion des composants. L'application de gestion des composants analyse les documents et exécute les étapes souhaitées.

Après leur création, un ou plusieurs composants sont regroupés à l'aide d'une recette d'image ou d'une recette de conteneur afin de définir le plan de création et de test d'une machine virtuelle ou d'une image de conteneur. Vous pouvez utiliser des composants publics détenus et gérés par AWS, ou vous pouvez créer les vôtres. Pour plus d'informations sur les composants, consultezComment Image Builder utilise AWS Task Orchestrator and Executor application pour gérer les composants.

Document sur les composants

Document YAML ou JSON déclaratif en texte brut qui décrit la configuration d'une personnalisation que vous pouvez appliquer à votre image. Le document est utilisé pour créer un composant de construction ou de test.

Étapes d'exécution

EC2 Image Builder comporte deux étapes d'exécution : construction et test. Chaque phase d'exécution comporte une ou plusieurs phases dont la configuration est définie par le document du composant.

Phases de configuration

La liste suivante indique les phases exécutées pendant les phases de construction et de test :

Étape de construction :

Phase de génération

Un pipeline d'images commence par la phase de génération de l'étape de génération lorsqu'il s'exécute. L'image de base est téléchargée et la configuration spécifiée pour la phase de génération du composant est appliquée pour créer et lancer une instance.

Phase de validation

Une fois qu'Image Builder a lancé l'instance et appliqué toutes les personnalisations de la phase de création, la phase de validation commence. Au cours de cette phase, Image Builder s'assure que toutes les personnalisations fonctionnent comme prévu, en fonction de la configuration spécifiée par le composant pour la phase de validation. Si la validation de l'instance réussit, Image Builder arrête l'instance, crée une image, puis passe à la phase de test.

Étape de test :

Phase de test

Au cours de cette phase, Image Builder lance une instance à partir de l'image qu'il a créée une fois la phase de validation terminée avec succès. Image Builder exécute des composants de test au cours de cette phase pour vérifier que l'instance est saine et fonctionne comme prévu.

Phase de test de l'hôte du conteneur

Une fois qu'Image Builder a exécuté la phase de test pour tous les composants que vous avez sélectionnés dans la recette du conteneur, Image Builder exécute cette phase pour

les flux de travail du conteneur. La phase de test de l'hôte du conteneur peut exécuter des tests supplémentaires qui valident la gestion du conteneur et les configurations d'exécution personnalisées.

Flux de travail

Les flux de travail définissent la séquence d'étapes qu'Image Builder exécute lorsqu'il crée une nouvelle image. Toutes les images ont des flux de travail de création, de test et de distribution.

Types de flux de travail

`BUILD`

Couvre la configuration de la phase de construction pour chaque image créée.

`TEST`

Couvre la configuration de la phase de test pour chaque image créée.

`DISTRIBUTION`

Couvre la configuration de la phase de distribution pour chaque image créée.

Tarification

L'utilisation d'EC2 Image Builder pour créer des images d'AMI ou de conteneur personnalisées est gratuite. Cependant, la tarification standard s'applique aux autres services utilisés dans le processus. La liste suivante inclut l'utilisation de certaines images Services AWS qui peuvent entraîner des coûts lors de la création, du développement, du stockage et de la distribution de votre AMI ou de vos images de conteneur personnalisées, en fonction de votre configuration.

- Lancement d'une instance EC2
- Stockage des journaux sur Amazon S3
- Validation d'images avec Amazon Inspector
- Stockage des instantanés Amazon EBS pour votre AMIs
- Stockage d'images de conteneurs dans Amazon ECR
- Transférer et extraire des images de conteneurs vers et depuis Amazon ECR
- Si Systems Manager Advanced Tier est activé et que les instances Amazon EC2 s'exécutent avec une activation sur site, des ressources peuvent vous être facturées via Systems Manager

Connexe Services AWS

EC2 Image Builder en utilise Services AWS d'autres pour créer des images, en fonction de la configuration de votre recette Image Builder. Pour plus d'informations sur l'intégration de produits et de services pour vos images personnalisées, consultezIntégrez des produits et services dans Image Builder.

Comment fonctionne EC2 Image Builder

Lorsque vous utilisez la console EC2 Image Builder pour créer un pipeline d'images personnalisé, le système vous guide tout au long des étapes suivantes.

1. Spécifiez les détails du pipeline : entrez les informations relatives à votre pipeline, telles que le nom, la description, les balises et le calendrier d'exécution des builds automatisés. Vous pouvez choisir des versions manuelles, si vous préférez.
2. Choisir une recette : choisissez entre créer une AMI ou créer une image de conteneur. Pour les deux types d'images de sortie, vous entrez le nom et la version de votre recette, sélectionnez une image de base et choisissez les composants à ajouter pour la création et les tests. Vous pouvez également choisir la gestion automatique des versions, afin de toujours utiliser la dernière version du système d'exploitation (OS) disponible pour votre image de base. Les recettes de conteneur définissent également Dockerfiles et le référentiel Amazon ECR cible pour votre image de conteneur Docker en sortie.

   > **Note**
   >
   > Les composants sont les éléments de base utilisés par une recette d'image ou une recette de conteneur. Par exemple, des packages pour l'installation, des étapes de renforcement de la sécurité et des tests. L'image de base et les composants sélectionnés constituent une recette d'image.

3. Définissez la configuration de l'infrastructure : Image Builder lance EC2 des instances dans votre compte pour personnaliser les images et exécuter des tests de validation. Les paramètres de configuration de l'infrastructure spécifient les détails de l'infrastructure pour les instances qui s'exécuteront dans votre Compte AWS système pendant le processus de génération.
4. Définissez les paramètres de distribution : choisissez les AWS régions dans lesquelles distribuer votre image une fois la génération terminée et tous les tests réussis. Le pipeline distribue

automatiquement votre image dans la région où il exécute la génération, et vous pouvez ajouter une distribution d'image pour d'autres régions.

Les images que vous créez à partir de votre image de base personnalisée se trouvent dans votre Compte AWS. Vous pouvez configurer votre pipeline d'images pour produire des versions mises à jour et corrigées de votre image en saisissant un calendrier de génération. Lorsque le build est terminé, vous pouvez recevoir une notification via Amazon Simple Notification Service (SNS). Outre la production d'une image finale, l'assistant de console Image Builder génère une recette qui peut être utilisée avec les systèmes de contrôle de version existants et les pipelines de integration/continuous déploiement continu (CI/CD) pour une automatisation reproductible. Vous pouvez partager et créer de nouvelles versions de votre recette.

Contenu de la section

- Éléments de l'AMI
- Gestion des composants
- Ressources créées
- Distribution
- Partage de ressources
- Conformité d'

Éléments de l'AMI

Une Amazon Machine Image (AMI) est une image de machine virtuelle (VM) préconfigurée qui contient le système d'exploitation et le logiciel nécessaires au déploiement des EC2 instances.

Une AMI inclut les éléments suivants :

- Modèle pour le volume racine de la machine virtuelle. Lorsque vous lancez une EC2 machine virtuelle Amazon, le volume du périphérique racine contient l'image permettant de démarrer l'instance. Lorsque le stockage d'instance est utilisé, le périphérique racine est un volume de stockage d'instance créé à partir d'un modèle dans Amazon S3. Pour plus d'informations, consultez Amazon EC2 Root Device Volume.
- Lorsque Amazon EBS est utilisé, le périphérique racine est un volume EBS créé à partir d'un instantané EBS.
- Autorisations de lancement qui déterminent Comptes AWS ce qui peut être lancé VMs avec l'AMI.

- Bloquez les données de mappage des périphériques qui spécifient les volumes à associer à l'instance après le lancement.
- Un identifiant de ressource unique pour chaque région, pour chaque compte.
- Charges utiles de métadonnées, telles que les balises, et propriétés, telles que la région, le système d'exploitation, l'architecture, le type de périphérique racine, le fournisseur, les autorisations de lancement, le stockage pour le périphérique racine et le statut de signature.
- Signature AMI pour les images Windows afin de les protéger contre toute altération non autorisée. Pour plus d'informations, consultez la section Documents d'identité de l'instance.

Gestion des composants

EC2 Image Builder utilise une application de gestion des composants AWS Task Orchestrator and Executor (AWSTOE) qui vous aide à orchestrer des flux de travail complexes, à modifier les configurations système et à tester vos systèmes à l'aide de composants de script basés sur YAML. Comme AWSTOE il s'agit d'une application autonome, elle ne nécessite aucune configuration supplémentaire. Il peut fonctionner sur n'importe quelle infrastructure cloud et sur site. Pour commencer à l'utiliser en AWSTOE tant qu'application autonome, voirConfiguration manuelle pour développer des composants personnalisés avec AWSTOE.

Image Builder permet AWSTOE d'effectuer toutes les activités sur instance. Il s'agit notamment de créer et de valider votre image avant de prendre un instantané, et de tester l'instantané pour vous assurer qu'il fonctionne comme prévu avant de créer l'image finale. Pour plus d'informations sur la façon dont Image AWSTOE Builder gère ses composants, consultezUtilisez des composants pour personnaliser l'image de votre Image Builder. Pour plus d'informations sur la création de composants avec AWSTOE, consultezComment Image Builder utilise AWS Task Orchestrator and Executor application pour gérer les composants.

Test d'image

Vous pouvez utiliser des composants de AWSTOE test pour valider votre image et vous assurer qu'elle fonctionne comme prévu avant de créer l'image finale.

En général, chaque composant de test consiste en un document YAML contenant un script de test, un binaire de test et des métadonnées de test. Le script de test contient les commandes d'orchestration permettant de démarrer le binaire de test, qui peut être écrit dans n'importe quelle langue prise en charge par le système d'exploitation. Les codes d'état de sortie indiquent le résultat

du test. Les métadonnées du test décrivent le test et son comportement ; par exemple, le nom, la description, les chemins vers le binaire de test et la durée prévue.

Ressources créées

Lorsque vous créez un pipeline, aucune ressource externe à Image Builder n'est créée, sauf si ce qui suit est vrai :

- Lorsqu'une image est créée dans le cadre du planning du pipeline
- Lorsque vous choisissez Run Pipeline dans le menu Actions de la console Image Builder
- Lorsque vous exécutez l'une de ces commandes depuis l'API ou AWS CLI : StartImagePipelineExecution ou CreateImage

Les ressources suivantes sont créées au cours du processus de création de l'image :

Pipelines d'images AMI

- EC2 instance (temporaire)
- Association d'inventaire de Systems Manager (via le gestionnaire d'état de Systems Manager si cette option `EnhancedImageMetadata` est activée) sur l' EC2 instance
- Amazon EC2 AMI
- Le snapshot Amazon EBS associé à Amazon AMI EC2

Pipelines d'images de conteneurs

- Conteneur Docker exécuté sur une EC2 instance (temporaire)
- L'association d'inventaire de Systems Manager (via Systems Manager State Manager) `EnhancedImageMetadata` est activée) sur l' EC2 instance
- Image de conteneur Docker
- Dockerfile

Une fois l'image créée, toutes les ressources temporaires sont supprimées.

Distribution

EC2 Image Builder peut distribuer AMIs ou stocker des images dans n'importe quelle AWS région. L'image est copiée dans chaque région que vous spécifiez dans le compte utilisé pour créer l'image.

Pour les images de sortie d'AMI, vous pouvez définir des autorisations de lancement d'AMI afin de contrôler Comptes AWS les personnes autorisées à lancer des EC2 instances avec l'AMI créée. Par exemple, vous pouvez rendre l'image privée, publique ou partager avec des comptes spécifiques. Si vous distribuez l'AMI à d'autres régions et que vous définissez des autorisations de lancement pour d'autres comptes, les autorisations de lancement sont propagées AMIs dans toutes les régions dans lesquelles l'AMI est distribuée.

Vous pouvez également utiliser votre AWS Organizations compte pour imposer des restrictions aux comptes membres afin de lancer des instances uniquement si elles sont approuvées et conformes AMIs. Pour plus d'informations, consultez la section Gérer le Comptes AWS au sein de votre organisation.

Pour mettre à jour vos paramètres de distribution à l'aide de la console Image Builder, suivez les étapes de Créez une nouvelle version d'image de la recette depuis la console ouCréez une nouvelle version de recette de conteneur à l'aide de la console.

Partage de ressources

Pour partager des composants, des recettes ou des images avec d'autres comptes ou au sein d'autres comptes AWS Organizations, consultezPartagez les ressources d'Image Builder avec AWS RAM.

Conformité d'

Pour les benchmarks du Center for Internet Security (CIS), EC2 Image Builder utilise Amazon Inspector pour évaluer l'exposition, les vulnérabilités et les écarts par rapport aux meilleures pratiques et aux normes de conformité. Par exemple, Image Builder évalue l'accessibilité involontaire au réseau, la connectivité Internet publique non corrigée CVEs et l'activation de la connexion root à distance. Amazon Inspector est proposé en tant que composant de test que vous pouvez choisir d'ajouter à votre recette d'images. Pour plus d'informations sur Amazon Inspector, consultez le guide de l'utilisateur d'Amazon Inspector. Pour plus d'informations, consultez le Center for Internet Security (CIS) Benchmarks.

Image Builder fournit des composants de renforcement STIG pour vous aider à créer plus efficacement des images conformes aux normes STIG de base. Ces composants STIG détectent les erreurs de configuration et exécutent un script de correction. L'utilisation de composants conformes aux normes STIG est gratuite. Pour une liste complète des composants STIG disponibles via Image Builder, consultezAmazon a géré les composants de renforcement STIG pour Image Builder.

Versionnage sémantique dans Image Builder

Image Builder utilise le versionnement sémantique pour organiser les ressources et s'assurer qu'elles sont uniques. IDs La version sémantique comporte quatre nœuds :

<major>. *<minor>*. *<patch>*/<build>

Vous pouvez attribuer des valeurs aux trois premiers et filtrer en les appliquant tous.

Le versionnement sémantique est inclus dans le nom de ressource Amazon (ARN) de chaque objet, au niveau qui s'applique à cet objet comme suit :

1. Versionless ARNs et Name ARNs n'incluent aucune valeur spécifique dans aucun des nœuds. Les nœuds sont soit complètement omis, soit spécifiés sous forme de caractères génériques, par exemple : x.x.x.
2. La version ARNs n'a que les trois premiers nœuds :<major>. <minor>. <patch>
3. La version ARNs de construction comporte les quatre nœuds et pointe vers une version spécifique pour une version spécifique d'un objet.

Affectation : Pour les trois premiers nœuds, vous pouvez attribuer n'importe quelle valeur entière positive, ou zéro, avec une limite supérieure de 2^30-1, ou 1073741823 pour chaque nœud. Image Builder attribue automatiquement le numéro de compilation au quatrième nœud.

Modèles : vous pouvez utiliser n'importe quel modèle numérique qui respecte les exigences d'attribution pour les nœuds que vous pouvez attribuer. Par exemple, vous pouvez choisir un modèle de version de logiciel, tel que 1.0.0, ou une date, telle que 2021.01.01.

Sélection : avec le versionnement sémantique, vous pouvez utiliser des caractères génériques (x) pour spécifier les versions ou les nœuds les plus récents lors de la sélection de l'image de base ou des composants de votre recette. Lorsque vous utilisez un caractère générique dans un nœud, tous les nœuds situés à droite du premier caractère générique doivent également être des caractères génériques.

Par exemple, étant donné les versions récentes suivantes : 2.2.4, 1.7.8 et 1.6.8, la sélection de version à l'aide de caractères génériques produit les résultats suivants :

- `x.x.x`= 2,2,4
- `1.x.x`= 1,7.8

- `1.6.x`= 1,6.8
- `x.2.x`n'est pas valide et produit une erreur
- `1.x.8`n'est pas valide et produit une erreur

Gestion des versions génériques dans les politiques de cycle de vie

Vous pouvez utiliser des modèles génériques pour les versions sémantiques lors de la sélection de recettes de politique de cycle de vie afin de cibler plusieurs versions d'une recette avec une seule politique. Cela simplifie la gestion du cycle de vie en éliminant le besoin de créer des politiques distinctes pour chaque version de recette.

Les modèles génériques suivants sont pris en charge pour les versions de recettes de politique de cycle de vie :

- `x.x.x`— Correspond à toutes les versions de la recette.
- `1.x.x`— Correspond à toutes les versions mineures et patchs de la version majeure 1.
- `1.0.x`— Correspond à toutes les versions de patch de la version 1.0.

Lorsqu'une politique de cycle de vie comportant des modèles génériques est exécutée, Image Builder résout les caractères génériques en fonction de toutes les versions de recette correspondantes au moment de l'exécution. Cela crée une liste immuable de versions pour cette exécution. Les nouvelles versions de recettes créées après le début de l'exécution de la politique sont automatiquement incluses dans la prochaine exécution planifiée.

Pour plus d'informations sur la création de politiques de cycle de vie avec des versions génériques, consultez. Créez des politiques de cycle de vie

Utilisation des références de version

Les références de version sont des chaînes ready-to-use ARN qui incorporent des modèles génériques basés sur la version sémantique de la ressource que vous avez créée ou récupérée. Au lieu d'écrire du code personnalisé pour analyser ARNs et insérer des caractères génériques, Image Builder s'en charge pour vous.

Lorsque vous créez ou récupérez des ressources Image Builder, Image Builder fournit automatiquement des modèles de version préconstruits ARNs avec des caractères génériques dans l'`latestVersionReferences`objet. Il n'est donc plus nécessaire d'analyser et de reconstruire

manuellement ARNs lorsque vous souhaitez référencer des ressources à l'aide de modèles de version génériques.

Par exemple, lorsque vous créez un composant avec une version`1.2.3`, Image Builder renvoie :

```
{
    "componentBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
my-component/1.2.3/1",
    "latestVersionReferences": {
        "latestVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-
component/x.x.x",
        "latestMajorVersionArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-component/1.x.x",
        "latestMinorVersionArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-component/1.2.x",
        "latestPatchVersionArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-component/1.2.3"
    }
}
```

Modèles de référence des versions disponibles

L'`latestVersionReferences`objet contient quatre modèles d'ARN :

- latestVersionArn (x.x.x) - Résout toujours à la dernière version absolue.
- atestMajorVersionArn (1.x.x) : résout les dernières versions mineures et patchs d'une version majeure.
- latestMinorVersionArn (1.2.x) : résout la dernière version du correctif dans une version mineure spécifique.
- latestPatchVersionArn (1.2.3) - Fait référence à une version sémantique spécifique et renvoie à la dernière version de construction pour les ressources prenant en charge plusieurs versions de construction.

Ressources qui renvoient des références de version

Les références de version sont renvoyées par `Create` et `Get` APIs pour toutes les ressources Image Builder versionnées :

- Composants -`CreateComponent`, `GetComponent`
- Recettes d'images -`CreateImageRecipe`, `GetImageRecipe`
- Recettes en contenants -`CreateContainerRecipe`, `GetContainerRecipe`
- Des photos -`CreateImage`, `GetImage`
- Flux de travail -`CreateWorkflow`, `GetWorkflow`

Remarque : Pour les flux de travail gérés par Image Builder, seuls `latestVersionArn (x.x.x)` les flux de travail gérés par Image Builder sont renvoyés, car Image Builder exige que vous utilisiez toujours la dernière version des flux de travail gérés par Image Builder.

Préparez-vous à créer des images personnalisées avec Image Builder

Avant de créer des images avec EC2 Image Builder, vérifiez que vous remplissez les conditions préalables suivantes pour créer un pipeline d'images. Sauf indication contraire, ces conditions préalables sont requises pour tous les types de pipelines.

Conditions préalables

- Rôle lié au service Image Builder
- Exigences de configuration
- Référentiel de conteneurs pour les pipelines d'images de conteneurs
- Hôte dédié pour les images macOS
- Conditions préalables à l'IAM
- Conditions préalables requises pour l'agent Systems Manager

Une fois que vous avez rempli les conditions requises, vous pouvez gérer EC2 Image Builder à partir de l'une des interfaces suivantes.

- Console EC2 Image Builder
- Les commandes Image Builder dans AWS CLI
- Référence de l'API EC2 Image Builder
- AWS SDKs et outils

Rôle lié au service Image Builder

EC2 Image Builder utilise un rôle lié à un service pour accorder des autorisations à AWS d'autres services en votre nom. Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez votre première ressource Image Builder dans la console AWS de gestion AWS CLI, ou dans l' AWS API, Image Builder crée pour vous le rôle lié au service. Pour plus d'informations sur le rôle lié à un service créé par Image Builder dans votre compte, consultez. Utiliser des rôles liés à un service IAM pour Image Builder

Exigences de configuration

- Image Builder prend en charge AWS PrivateLink. Pour plus d'informations sur la configuration des points de terminaison VPC pour Image Builder, consultez. Image Builder et AWS PrivateLink Points de terminaison de VPC d'interface
- Les instances utilisées par Image Builder pour créer des images de conteneur doivent disposer d'un accès Internet pour les télécharger AWS CLI depuis Amazon S3 et pour télécharger une image de base depuis le référentiel Docker Hub, le cas échéant. Image Builder utilise le AWS CLI pour obtenir le Dockerfile à partir de la recette du conteneur, où il est stocké sous forme de données.
- Les instances utilisées par Image Builder pour créer des images et exécuter des tests doivent avoir accès au service Systems Manager. Les exigences d'installation dépendent de votre système d'exploitation.

  Pour connaître les exigences d'installation de votre image de base, choisissez l'onglet correspondant au système d'exploitation de votre image de base.

  Linux

  Pour les instances Linux Amazon EC2, Image Builder installe l'agent Systems Manager sur l'instance de build s'il n'est pas déjà présent, et le supprime avant de créer l'image.

  Windows

  Image Builder n'installe pas l'agent Systems Manager sur les instances de build Windows Server d'Amazon EC2. Si votre image de base n'est pas préinstallée avec l'agent Systems Manager, vous devez lancer une instance à partir de votre image source, installer manuellement Systems Manager sur l'instance et créer une nouvelle image de base à partir de votre instance.

  Pour installer manuellement l'agent Systems Manager sur votre instance Amazon EC2 Windows Server, consultez la section Installation manuelle de l'agent Systems Manager sur les instances EC2 pour Windows Server dans le guide de l'AWS Systems Manager utilisateur.

Référentiel de conteneurs pour les pipelines d'images de conteneurs

Pour les pipelines d'images de conteneurs, la recette définit la configuration des images Docker produites et stockées dans le référentiel de conteneurs cible. Vous devez créer le référentiel cible avant de créer la recette de conteneur pour votre image Docker.

Image Builder utilise Amazon ECR comme référentiel cible pour les images de conteneurs. Pour créer un référentiel Amazon ECR, suivez les étapes décrites dans la section Création d'un référentiel dans le guide de l'utilisateur d'Amazon Elastic Container Registry.

Hôte dédié pour les images macOS

Les instances Mac Amazon EC2 nécessitent un hôte dédié exécuté sur un type d'instance métallique. Avant de créer une image macOS personnalisée, vous devez attribuer un hôte dédié à votre compte. Pour plus d'informations sur les instances Mac et une liste des types d'instances compatibles nativement avec le système d'exploitation macOS, consultez la section Instances Mac Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

Lorsque vous avez créé un hôte dédié, vous pouvez configurer les paramètres dans la ressource de configuration de l'infrastructure pour votre image. La configuration de l'infrastructure inclut des propriétés de placement dans lesquelles vous pouvez spécifier l'hôte, le groupe de placement d'hôtes ou la zone de disponibilité où doivent être placées les instances lancées depuis votre image.

Conditions préalables à l'IAM

Le rôle IAM que vous associez à votre profil d'instance doit être autorisé à exécuter les composants de génération et de test inclus dans votre image. Les politiques de rôle IAM suivantes doivent être associées au rôle IAM associé au profil d'instance :

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- Amazon SSMManaged InstanceCore

Si vous configurez la journalisation, le profil d'instance spécifié dans la configuration de votre infrastructure doit disposer d'`s3:PutObject`autorisations pour le compartiment cible (`arn:aws:s3:::BucketName/*`). Par exemple :

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "UploadFileToS3Bucket",
   "Effect": "Allow",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::bucket-name/*"
  }
 ]
}
```

Attacher une stratégie

Les étapes suivantes vous guident tout au long du processus d'attachement des politiques IAM à un rôle IAM pour accorder les autorisations précédentes.

1. Connectez-vous à la console de AWS gestion et ouvrez la console IAM à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de gauche, choisissez Politiques.
3. Filtrer la liste des politiques avec EC2InstanceProfileForImageBuilder
4. Sélectionnez la bullet à côté de la politique, puis dans la liste déroulante des actions de stratégie, sélectionnez Joindre.
5. Sélectionnez le nom du rôle IAM auquel vous souhaitez associer la politique.
6. Choisissez Attach policy (Attacher la politique).
7. Répétez les étapes 3 à 6 pour les SSMManaged InstanceCore politiques EC2InstanceProfileForImageBuilderECRContainerBuildset celles d'Amazon.

> **Note**
>
> Si vous souhaitez copier une image créée avec Image Builder vers un autre compte, vous devez créer le `EC2ImageBuilderDistributionCrossAccountRole` rôle dans tous les comptes cibles et associer la politique Ec2ImageBuilderCrossAccountDistributionAccess

policy gérée au rôle. Pour de plus amples informations, veuillez consulter Partagez les ressources d'Image Builder avec AWS RAM.

Conditions préalables requises pour l'agent Systems Manager

EC2 Image Builder AWS Systems Manager exécute l'agent (Systems Manager) sur les instances EC2 qu'il lance pour créer et tester votre image. Image Builder collecte des informations supplémentaires sur l'instance utilisée pendant la phase de construction avec Systems Manager Inventory. Ces informations incluent le nom et la version du système d'exploitation (OS), ainsi que la liste des packages et de leurs versions respectives tels qu'indiqués par votre système d'exploitation.

Pour refuser la collecte de ces informations, sélectionnez la méthode qui correspond à votre environnement préféré :

- Console Image Builder : désélectionnez la case à cocher Activer la collecte améliorée de métadonnées.
- AWS CLI— Spécifiez l'`--no-enhanced-image-metadata-enabled`option
- API Image Builder ou SDKs — Définissez le `enhancedImageMetadataEnabled` paramètre sur`false`.

Image Builder envoie des actions `RunCommand` à votre instance de génération et de test dans le cadre du flux de travail de création et de test d'image. Vous ne pouvez pas refuser l'utilisation de `RunCommand` pour envoyer des actions à votre instance de build et de test.

Apprenez à créer des images personnalisées à l'aide des didacticiels Image Builder

Il existe de nombreuses manières de créer des images et des composants personnalisés avec EC2 Image Builder. Les didacticiels vous aident à découvrir les principaux concepts d'Image Builder. Chaque didacticiel présente un cas d'utilisation avec des étapes que vous pouvez suivre pour la première fois. Les instructions utilisent des valeurs par défaut dans la mesure du possible pour faciliter l'apprentissage du processus global. Après avoir utilisé l'un des didacticiels, vous pouvez explorer d'autres moyens de personnaliser vos propres images.

Créez votre première image

Les didacticiels suivants vous montrent comment créer votre première image à l'aide de l'assistant de console Image Builder. À la fin du didacticiel, vous aurez créé l'ensemble suivant de ressources Image Builder. La dernière étape du didacticiel consiste à nettoyer les ressources que vous avez créées.

- Une recette d'image pour votre Amazon Machine Image (AMI) ou une recette de conteneur pour votre image de conteneur.
- Une ressource de configuration d'infrastructure avec des paramètres par défaut.
- Une ressource de paramètres de distribution avec des paramètres par défaut qui distribue la sortie à la région source (votre compte dans la région que vous utilisez pour exécuter l'assistant de console).
- Un pipeline d'images qui utilise les ressources répertoriées pour créer votre image de sortie avec les flux de travail de création d'images par défaut.
- Une AMI de sortie ou une image de conteneur.

Tutoriels Console Wizard

- Assistant de pipeline : création d'une AMI
- Assistant de pipeline : création d'une image de conteneur

Création d'un composant personnalisé avec des paramètres d'entrée

Le didacticiel suivant vous montre comment créer un composant personnalisé qui définit les paramètres d'entrée, puis comment définir les valeurs à partir de votre recette Image Builder.

Composant personnalisé avec paramètres

Utiliser les paramètres de Systems Manager avec Image Builder

Le didacticiel suivant explique comment créer un AWS Systems Manager paramètre Parameter Store et l'utiliser dans une recette d'image.

Utilisez un paramètre d'image de base dans votre recette

Vous pouvez également utiliser les paramètres Parameter Store dans les paramètres de distribution AMI pour stocker l'ID de votre image de sortie et dans des composants personnalisés. Pour plus d'informations, consultez Création et mise à jour des configurations de distribution AMI la section pour les distributions et Utiliser les paramètres du magasin de paramètres de Systems Manager pour les composants personnalisés.

Déployez des exemples de projets avec une infrastructure en tant que code

Lorsque vous êtes prêt à définir vos ressources d'images sous forme de code, le référentiel d'exemples Image Builder GitHub fournit des exemples de projets complets pour les cas d'utilisation courants. Les exemples couvrent des scénarios tels que les pipelines AMI dorés, la distribution entre comptes et les builds de VPC privés. Pour de plus amples informations, veuillez consulter Découvrez des exemples de projets Image Builder sur GitHub.

Tutoriel : Création d'un pipeline d'images avec une AMI de sortie à partir de l'assistant de console Image Builder

Ce didacticiel explique comment créer un pipeline automatique pour créer et gérer une image EC2 Image Builder personnalisée à l'aide de l'assistant de console Create image pipeline. Pour vous aider à suivre les étapes de manière efficace, les paramètres par défaut sont utilisés lorsqu'ils sont disponibles et les sections facultatives sont ignorées.

Créer un flux de travail de pipeline d'images

- Étape 1 : Spécifier les détails du pipeline
- Étape 2 : Choisissez la recette
- Étape 3 : Définition de la configuration de l'infrastructure (facultatif)
- Étape 4 : définir les paramètres de distribution (facultatif)
- Étape 5 : Vérifier
- Étape 6 : Nettoyer

Étape 1 : Spécifier les détails du pipeline

1. Ouvrez la console https://console.aws.amazon.com/imagebuilder/EC2 Image Builder à l'adresse.
2. Pour commencer à créer votre pipeline, choisissez Create image pipeline.
3. Dans la section Général, entrez le nom de votre pipeline (obligatoire).

> **Tip**
>
> La collecte de métadonnées améliorée est activée par défaut. Pour garantir la compatibilité entre les composants et les images de base, maintenez-le activé.

4. Dans la section Créer un calendrier, vous pouvez conserver les valeurs par défaut des options de planification. Notez que le fuseau horaire indiqué pour le calendrier par défaut est le temps universel coordonné (UTC). Pour plus d'informations sur l'heure UTC et pour connaître le décalage correspondant à votre fuseau horaire, consultez la section Abréviations des fuseaux horaires — Liste mondiale.

   Pour les paramètres de mise à jour des dépendances, choisissez l'option Exécuter le pipeline à l'heure planifiée s'il existe des mises à jour des dépendances. Ce paramètre oblige votre pipeline à vérifier les mises à jour avant de démarrer la génération. S'il n'y a aucune mise à jour, la construction planifiée du pipeline est ignorée.

> **Note**
>
> Pour vous assurer que votre pipeline reconnaît les mises à jour et les builds des dépendances comme prévu, vous devez utiliser le versionnement sémantique (x.x.x) pour votre image de base et vos composants. Pour en savoir plus sur le versionnement

sémantique des ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

5. Choisissez Next pour passer à l'étape suivante.

Étape 2 : Choisissez la recette

1. Image Builder utilise par défaut la recette existante dans la section Recette. Pour la première fois, choisissez l'option Créer une nouvelle recette.
2. Dans la section Type d'image, choisissez l'option Amazon Machine Image (AMI) pour créer un pipeline d'images qui produira et distribuera une AMI.
3. Dans la section Général, entrez les cases obligatoires suivantes :
   - Nom — le nom de votre recette
   - Version : version de votre recette (utilisez le format) <major>. <minor>. <patch>, où major, minor et patch sont des valeurs entières). Les nouvelles recettes commencent généralement par`1.0.0`.
4. Dans la section Image source, conservez les valeurs par défaut pour Sélectionner une image, Système d'exploitation de l'image (OS) et Origine de l'image. Il en résulte une liste des systèmes Linux AMIs gérés par Amazon. Pour ce didacticiel, sélectionnez l'`Amazon Linux 2 x86`image.
   a. Dans le menu déroulant Nom de l'image, choisissez une image.
   b. Conservez la valeur par défaut pour les options de gestion automatique des versions (utilisez la dernière version du système d'exploitation disponible).

   > **Note**
   >
   > Ce paramètre garantit que votre pipeline utilise le versionnement sémantique pour l'image de base, afin de détecter les mises à jour des dépendances pour les tâches planifiées automatiquement. Pour en savoir plus sur le versionnement sémantique des ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

5. Dans la section Configuration de l'instance, conservez les valeurs par défaut pour l'agent Systems Manager. Image Builder conserve donc l'agent Systems Manager une fois la

compilation et les tests terminés, afin d'inclure l'agent Systems Manager dans votre nouvelle image.

Laissez les données utilisateur vides pour ce didacticiel. Vous pouvez utiliser cette zone à d'autres moments pour fournir des commandes ou un script de commande à exécuter lorsque vous lancez votre instance de build. Toutefois, il remplace toutes les commandes qu'Image Builder aurait pu ajouter pour garantir l'installation de Systems Manager. Lorsque vous l'utilisez, assurez-vous que l'agent Systems Manager est préinstallé sur votre image de base ou que vous incluez l'installation dans vos données utilisateur.

6. Dans la section Composants, vous pouvez choisir de ne pas ajouter de composants et de continuer. Si vous souhaitez ajouter des composants, dans le panneau Créer des composants, choisissez Ajouter des composants de génération, puis sélectionnez-les dans la liste `Amazon managed` de filtres du propriétaire du composant. Cela ouvre un panneau de sélection sur le côté droit de l'interface de la console dans lequel vous pouvez parcourir et filtrer les composants disponibles.

   Pour ce didacticiel, choisissez un composant qui met à jour Linux avec les dernières mises à jour de sécurité, comme suit :

   a. Filtrez les résultats en saisissant le mot `update` dans la barre de recherche située en haut du panneau.

   b. Cochez la case correspondant au composant de `update-linux` construction.

   c. Conservez la valeur par défaut pour les options de gestion des versions (utilisez la dernière version disponible).

      > **Note**
      >
      > Ce paramètre garantit que votre pipeline utilise le versionnement sémantique pour le composant sélectionné, afin de détecter les mises à jour de dépendance pour les tâches planifiées automatiquement. Pour en savoir plus sur le versionnement sémantique des ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

   d. Choisissez Ajouter à la recette pour ajouter le composant à votre recette. Cela ferme le panneau de sélection des composants.

   e. De retour dans le panneau Créer des composants, le composant que vous avez ajouté s'affiche.

7. Réorganiser les composants (facultatif)

   Si vous avez choisi plusieurs composants à inclure dans votre image, vous pouvez utiliser cette drag-and-drop action pour les réorganiser dans l'ordre dans lequel ils doivent s'exécuter pendant le processus de création.

   > **Note**
   >
   > Les composants de durcissement CIS ne respectent pas les règles de classement des composants standard des recettes Image Builder. Les composants de renforcement CIS sont toujours exécutés en dernier pour garantir que les tests de référence s'exécutent par rapport à votre image de sortie.

   a. Répétez les étapes précédentes pour ajouter le `update-linux-kernel-5` composant à votre recette.

   b. Le composant que vous venez d'ajouter possède un paramètre d'entrée pour la version du noyau. Pour développer les paramètres des options de version ou des paramètres d'entrée, vous pouvez cliquer sur la flèche à côté du nom du paramètre. Pour étendre tous les paramètres de tous les composants sélectionnés, vous pouvez activer ou désactiver le bouton Tout étendre. Pour plus d'informations sur l'utilisation des paramètres d'entrée dans vos composants et leur définition dans vos recettes, consultezTutoriel : Création d'un composant personnalisé avec des paramètres d'entrée.

   c. Choisissez l'un des composants et faites-le glisser vers le haut ou vers le bas pour modifier l'ordre dans lequel les composants seront exécutés.

   d. Pour supprimer le `update-linux-kernel-5` composant, choisissez X dans le coin supérieur droit de la boîte du composant.

      Répétez cette étape pour supprimer tous les autres composants que vous avez éventuellement ajoutés, en ne laissant que le `update-linux` composant sélectionné.

8. Choisissez Next pour passer à l'étape suivante.

Étape 3 : Définition de la configuration de l'infrastructure (facultatif)

Image Builder lance des instances EC2 dans votre compte pour personnaliser les images et exécuter des tests de validation. Les paramètres de configuration de l'infrastructure spécifient les détails de

l'infrastructure pour les instances qui s'exécuteront dans votre Compte AWS système pendant le processus de génération.

Dans la section Configuration de l'infrastructure, les options de configuration sont définies par défaut sur`Create infrastructure configuration using service defaults`. Cela crée un rôle IAM et un profil d'instance associé pour les instances de build et de test EC2 utilisées pour configurer votre image. Pour plus d'informations sur les paramètres de configuration de l'infrastructure, consultez le CreateInfrastructureConfigurationmanuel de référence de l'API EC2 Image Builder.

Pour ce didacticiel, nous utilisons les paramètres par défaut.

> **Note**
>
> Pour spécifier un sous-réseau à utiliser pour un VPC privé, vous pouvez créer votre propre configuration d'infrastructure personnalisée ou utiliser des paramètres que vous avez déjà créés.

- Choisissez Next pour passer à l'étape suivante.

Étape 4 : définir les paramètres de distribution (facultatif)

Les configurations de distribution incluent le nom de l'AMI de sortie, les paramètres régionaux spécifiques pour le chiffrement, les autorisations de lancement Comptes AWS, les organisations et unités organisationnelles (OUs) qui peuvent lancer l'AMI de sortie, ainsi que les configurations de licence.

Dans la section Paramètres de distribution, les options de configuration sont définies par défaut sur`Create distribution settings using service defaults`. Cette option distribuera l'AMI de sortie à la région actuelle. Pour plus d'informations sur la configuration de vos paramètres de distribution, consultezGérer les paramètres de distribution d'Image Builder.

Pour ce didacticiel, nous utilisons les paramètres par défaut.

- Choisissez Next pour passer à l'étape suivante.

Étape 5 : Vérifier

La section Révision affiche tous les paramètres que vous avez configurés. Pour modifier les informations d'une section donnée, cliquez sur le bouton Modifier situé dans le coin supérieur droit de la section d'étape. Par exemple, si vous souhaitez modifier le nom de votre pipeline, cliquez sur le bouton Modifier dans le coin supérieur droit de la section Étape 1 : Détails du pipeline.

1. Lorsque vous avez revu vos paramètres, choisissez Create pipeline pour créer votre pipeline.
2. Vous pouvez voir les messages de réussite ou d'échec en haut de la page, lorsque vos ressources sont créées pour les paramètres de distribution, la configuration de l'infrastructure, votre nouvelle recette et le pipeline. Pour voir les détails d'une ressource, y compris son identifiant, choisissez Afficher les détails.
3. Après avoir consulté les détails d'une ressource, vous pouvez consulter les détails des autres ressources en choisissant le type de ressource dans le volet de navigation. Par exemple, pour voir les détails de votre nouveau pipeline, choisissez Image pipelines dans le volet de navigation. Si votre construction a réussi, votre nouveau pipeline est affiché dans la liste des pipelines d'images.

Étape 6 : Nettoyer

Votre environnement Image Builder, tout comme votre maison, a besoin d'un entretien régulier pour vous aider à trouver ce dont vous avez besoin et à accomplir vos tâches sans vous encombrer. Assurez-vous de nettoyer régulièrement les ressources temporaires que vous avez créées pour les tests. Sinon, vous pourriez oublier ces ressources et, plus tard, ne plus vous souvenir de leur utilisation. D'ici là, il se peut que vous ne sachiez pas si vous pouvez vous en débarrasser en toute sécurité.

> **Tip**
>
> Pour éviter les erreurs de dépendance lorsque vous supprimez des ressources, veillez à supprimer vos ressources dans l'ordre suivant :
>
> 1. Pipeline d'images
> 2. Recette d'images
> 3. Toutes les ressources restantes

Pour nettoyer les ressources que vous avez créées pour ce didacticiel, procédez comme suit :

Supprimer le pipeline

1. Pour voir la liste des pipelines de génération créés sous votre compte, choisissez Image pipelines dans le volet de navigation.
2. Cochez la case à côté du nom du pipeline pour sélectionner le pipeline que vous souhaitez supprimer.
3. En haut du panneau Pipelines d'images, dans le menu Actions, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer la recette

1. Pour voir la liste des recettes créées sous votre compte, choisissez Image recipes dans le volet de navigation.
2. Cochez la case à côté du nom de la recette pour sélectionner la recette que vous souhaitez supprimer.
3. En haut du panneau des recettes d'images, dans le menu Actions, choisissez Supprimer la recette.
4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer la configuration de l'infrastructure

1. Pour consulter la liste des configurations d'infrastructure créées sous votre compte, choisissez Configuration de l'infrastructure dans le volet de navigation.
2. Cochez la case à côté du nom de la configuration pour sélectionner la configuration d'infrastructure que vous souhaitez supprimer.
3. En haut du panneau de configuration de l'infrastructure, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer les paramètres de distribution

1. Pour voir la liste des paramètres de distribution créés sous votre compte, choisissez Paramètres de distribution dans le volet de navigation.

2. Cochez la case à côté du nom de la configuration pour sélectionner les paramètres de distribution que vous avez créés pour ce didacticiel.
3. En haut du panneau des paramètres de distribution, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer l'image

Suivez ces étapes pour vérifier que vous avez supprimé toutes les images créées à partir du pipeline du didacticiel. Il est peu probable que ce didacticiel crée une image à moins que suffisamment de temps ne se soit écoulé depuis que vous avez créé le pipeline qu'il exécute, conformément au calendrier de génération.

1. Pour voir la liste des images créées sous votre compte, choisissez Images dans le volet de navigation.
2. Choisissez la version de l'image que vous souhaitez supprimer. Cela ouvre la page des versions de génération d'images.
3. Cochez la case à côté de la version pour toute image que vous souhaitez supprimer. Vous pouvez sélectionner plusieurs versions d'image à la fois.
4. En haut du panneau des versions de génération d'images, choisissez Supprimer la version.
5. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Tutoriel : Création d'un pipeline d'images avec une image de conteneur Docker en sortie à partir de l'assistant de console Image Builder

Ce didacticiel explique comment créer un pipeline automatisé pour créer et gérer une image Docker EC2 Image Builder personnalisée à l'aide de l'assistant de console Créer un pipeline d'images. Pour vous aider à parcourir les étapes de manière efficace, les paramètres par défaut sont utilisés lorsqu'ils sont disponibles et les sections facultatives sont ignorées.

Créer un flux de travail de pipeline d'images

- Étape 1 : Spécifier les détails du pipeline
- Étape 2 : Choisissez la recette
- Étape 3 : Définition de la configuration de l'infrastructure (facultatif)

- Étape 4 : définir les paramètres de distribution (facultatif)
- Étape 5 : Vérifier
- Étape 6 : Nettoyer

Étape 1 : Spécifier les détails du pipeline

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Pour commencer à créer votre pipeline, choisissez Créer un pipeline d'images.
3. Dans la section Général, entrez le nom de votre pipeline (obligatoire).
4. Dans la section Créer un calendrier, vous pouvez conserver les valeurs par défaut pour les options de planification. Notez que le fuseau horaire indiqué pour le calendrier par défaut est le temps universel coordonné (UTC). Pour plus d'informations sur l'heure UTC et pour trouver le décalage correspondant à votre fuseau horaire, voir Abréviations des fuseaux horaires — Liste mondiale.

   Pour les paramètres de mise à jour des dépendances, choisissez l'option Exécuter le pipeline à l'heure planifiée s'il existe des mises à jour des dépendances. Ce paramètre oblige votre pipeline à vérifier les mises à jour avant de démarrer la compilation. S'il n'y a aucune mise à jour, la génération planifiée du pipeline est ignorée.

   > **Note**
   >
   > Pour vous assurer que votre pipeline reconnaît les mises à jour et les builds des dépendances comme prévu, vous devez utiliser le versionnement sémantique (x.x.x) pour votre image de base et vos composants. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

5. Cliquez sur Suivant pour passer à l'étape suivante.

Étape 2 : Choisissez la recette

1. Par défaut, Image Builder utilise la recette existante dans la section Recette. Pour la première fois, choisissez l'option Créer une nouvelle recette.

2. Dans la section Type d'image, choisissez l'option d'image Docker pour créer un pipeline de conteneurs qui produira une image Docker et la distribuera aux référentiels Amazon ECR dans les régions cibles.
3. Dans la section Général, saisissez les cases obligatoires suivantes :
   - Nom : nom de votre recette
   - Version : version de votre recette (utilisez le format) <major>. <minor>. <patch>, où major, minor et patch sont des valeurs entières). Les nouvelles recettes commencent généralement par`1.0.0`.
4. Dans la section Image source, conservez les valeurs par défaut pour Select image, Image Operating System (OS) et Image origin. Il en résulte une liste d'images de conteneurs Amazon Linux 2, gérées par Amazon, parmi lesquelles vous pouvez choisir pour votre image de base.
   a. Dans la liste déroulante Nom de l'image, choisissez une image.
   b. Conservez la valeur par défaut pour Auto-versioning les options (utilisez la dernière version du système d'exploitation disponible).

   > **Note**
   >
   > Ce paramètre garantit que votre pipeline utilise le contrôle de version sémantique pour l'image de base, afin de détecter les mises à jour des dépendances pour les tâches planifiées automatiquement. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder
5. Dans la section Composants, vous pouvez choisir de ne pas ajouter de composants et de continuer. Si vous souhaitez ajouter des composants, dans le panneau Créer des composants — Amazon Linux, vous pouvez parcourir les composants répertoriés sur la page. Utilisez le contrôle de pagination situé dans le coin supérieur droit pour parcourir les composants supplémentaires disponibles pour votre système d'exploitation d'image de base. Vous pouvez également rechercher des composants spécifiques ou créer votre propre composant de construction à l'aide du gestionnaire de composants.

   Pour ce didacticiel, choisissez un composant qui met à jour Linux avec les dernières mises à jour de sécurité, comme suit :

a. Filtrez les résultats en saisissant le mot `update` dans la barre de recherche située en haut du panneau.

b. Cochez la case correspondant au composant de `update-linux` compilation.

c. Faites défiler la page vers le bas et, dans le coin supérieur droit de la liste des composants sélectionnés, choisissez Tout développer.

d. Conservez la valeur par défaut pour les options de version (utilisez la dernière version du composant disponible).

> **Note**
>
> Ce paramètre garantit que votre pipeline utilise le contrôle de version sémantique pour le composant sélectionné, afin de détecter les mises à jour des dépendances pour les tâches planifiées automatiquement. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

Si vous aviez sélectionné un composant qui possède des paramètres d'entrée, les paramètres seraient également visibles dans cette zone. Les paramètres ne sont pas abordés dans ce didacticiel. Pour plus d'informations sur l'utilisation des paramètres d'entrée dans vos composants et leur définition dans vos recettes, consultezTutoriel : Création d'un composant personnalisé avec des paramètres d'entrée.

Réorganiser les composants (facultatif)

Si vous avez choisi plusieurs composants à inclure dans votre image, vous pouvez utiliser l'action glisser-déposer pour les réorganiser dans l'ordre dans lequel ils doivent être exécutés pendant le processus de création.

> **Note**
>
> Les composants de renforcement CIS ne suivent pas les règles d'ordre standard des composants dans les recettes d'Image Builder. Les composants de renforcement CIS sont toujours exécutés en dernier pour garantir que les tests de référence s'exécutent par rapport à votre image de sortie.

1. Revenez à la liste des composants disponibles.
2. Cochez la case correspondant au composant de `update-linux-kernel-mainline` construction (ou à tout autre composant de votre choix).
3. Faites défiler la page jusqu'à la liste des composants sélectionnés pour voir s'il y a au moins deux résultats.
4. Il est possible que la gestion des versions des composants récemment ajoutés ne soit pas étendue. Pour développer les options de version, vous pouvez soit choisir la flèche à côté des options de version, soit activer et désactiver le bouton Tout développer pour étendre la gestion des versions pour tous les composants sélectionnés.
5. Choisissez l'un des composants et faites-le glisser vers le haut ou vers le bas pour modifier l'ordre dans lequel les composants seront exécutés.
6. Pour supprimer le `update-linux-kernel-mainline` composant, sélectionnez-le X dans le coin supérieur droit de la zone du composant.
7. Répétez l'étape précédente pour supprimer tous les autres composants que vous auriez pu ajouter, en ne laissant que le `update-linux` composant sélectionné.

6. Dans la section Modèle Dockerfile, sélectionnez l'option Utiliser un exemple. Dans le panneau Contenu, notez les variables contextuelles dans lesquelles Image Builder place les informations de génération ou les scripts, en fonction de la recette de votre image de conteneur.

   Par défaut, Image Builder utilise les variables contextuelles suivantes dans votre Dockerfile.

   Image du parent (obligatoire)

   Au moment de la création, cette variable correspond à l'image de base de votre recette.

   Exemple :

   ```
   FROM
   {{{ imagebuilder:parentImage }}}
   ```

   environnements (obligatoire si des composants sont spécifiés)

   Au moment de la génération, cette variable est résolue en fonction des directives Dockerfile qui placent vos composants dans le contexte de génération. Ces directives copient les scripts

des composants dans l'image et définissent les variables d'environnement que ces scripts utilisent. Il prépare l'environnement mais n'exécute pas les composants que vous spécifiez. Placez cette variable avant la `components` variable.

Exemple :

```
{{{ imagebuilder:environments }}}
```

composants (en option)

Au moment de la génération, cette variable correspond à la commande qui exécute les scripts de génération et de test des composants pour les composants inclus dans la recette du conteneur, puis supprime ces scripts de l'image. Il s'agit de la variable qui exécute vos composants. Placez-le après la `environments` variable.

Exemple :

```
{{{ imagebuilder:components }}}
```

7. Dans la section Référentiel cible, indiquez le nom du référentiel Amazon ECR que vous avez créé comme condition préalable à ce didacticiel. Ce référentiel est utilisé comme paramètre par défaut pour la configuration de distribution dans la région où s'exécute le pipeline (région 1).

> **Note**
>
> Le référentiel cible doit exister dans Amazon ECR pour toutes les régions cibles avant la distribution.

8. Cliquez sur Suivant pour passer à l'étape suivante.

Étape 3 : Définition de la configuration de l'infrastructure (facultatif)

Image Builder lance des instances EC2 dans votre compte pour personnaliser les images et exécuter des tests de validation. Les paramètres de configuration de l'infrastructure spécifient les détails de l'infrastructure pour les instances qui seront exécutées dans votre Compte AWS ordinateur pendant le processus de génération.

Dans la section Configuration de l'infrastructure, les options de configuration sont définies par défaut sur`Create infrastructure configuration using service defaults`. Cela crée

un rôle IAM et un profil d'instance associé qui sont utilisés par les instances de build pour configurer vos images de conteneur. Vous pouvez également créer votre propre configuration d'infrastructure personnalisée ou utiliser les paramètres que vous avez déjà créés. Pour plus d'informations sur les paramètres de configuration de l'infrastructure, consultez CreateInfrastructureConfiguration la référence de l'API EC2 Image Builder.

Pour ce didacticiel, nous utilisons les paramètres par défaut.

- Cliquez sur Suivant pour passer à l'étape suivante.

Étape 4 : définir les paramètres de distribution (facultatif)

Les paramètres de distribution comprennent les régions cibles et le nom du référentiel Amazon ECR cible. Les images Docker de sortie sont déployées dans le référentiel Amazon ECR désigné dans chaque région.

Dans la section Paramètres de distribution, les options de configuration sont définies par défaut sur`Create distribution settings using service defaults`. Cette option distribuera l'image Docker de sortie au référentiel Amazon ECR spécifié dans votre recette de conteneur pour la région dans laquelle s'exécute votre pipeline (Région 1). Si vous le souhaitez`Create new distribution settings`, vous pouvez remplacer le référentiel ECR pour la région actuelle et ajouter d'autres régions à distribuer.

Pour ce didacticiel, nous utilisons les paramètres par défaut.

- Cliquez sur Suivant pour passer à l'étape suivante.

Étape 5 : Vérifier

La section Révision affiche tous les paramètres que vous avez configurés. Pour modifier les informations d'une section donnée, cliquez sur le bouton Modifier situé dans le coin supérieur droit de la section des étapes. Par exemple, si vous souhaitez modifier le nom de votre pipeline, cliquez sur le bouton Modifier en haut à droite de la section Étape 1 : Détails du pipeline.

1. Après avoir revu vos paramètres, choisissez Créer un pipeline pour créer votre pipeline.
2. Vous pouvez voir des messages de réussite ou d'échec en haut de la page, au fur et à mesure que vos ressources sont créées pour les paramètres de distribution, la configuration de

l'infrastructure, votre nouvelle recette et le pipeline. Pour voir les détails d'une ressource, y compris son identifiant, choisissez Afficher les détails.

3. Après avoir consulté les détails d'une ressource, vous pouvez consulter les détails des autres ressources en choisissant le type de ressource dans le volet de navigation. Par exemple, pour voir les détails de votre nouveau pipeline, choisissez Image pipelines dans le volet de navigation. Si votre compilation a été réussie, votre nouveau pipeline est affiché dans la liste des pipelines d'image.

Étape 6 : Nettoyer

Votre environnement Image Builder, tout comme votre maison, nécessite une maintenance régulière pour vous aider à trouver ce dont vous avez besoin et à effectuer vos tâches sans vous encombrer. Veillez à nettoyer régulièrement les ressources temporaires que vous avez créées pour les tests. Sinon, vous pourriez oublier ces ressources et, plus tard, ne plus vous rappeler à quoi elles ont été utilisées. D'ici là, il se peut que vous ne sachiez pas si vous pouvez vous en débarrasser en toute sécurité.

> **Tip**
>
> Pour éviter les erreurs de dépendance lorsque vous supprimez des ressources, veillez à les supprimer dans l'ordre suivant :
>
> 1. Pipeline d'images
> 2. Recette imagée
> 3. Toutes les ressources restantes

Pour nettoyer les ressources que vous avez créées pour ce didacticiel, procédez comme suit :

Supprimer le pipeline

1. Pour consulter la liste des pipelines de génération créés sous votre compte, choisissez Image pipelines dans le volet de navigation.
2. Cochez la case à côté de Nom du pipeline pour sélectionner le pipeline que vous souhaitez supprimer.
3. En haut du panneau Pipelines d'images, dans le menu Actions, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans la case et choisissez Supprimer.

Supprimer la recette du contenant

1. Pour consulter la liste des recettes de conteneurs créées sous votre compte, choisissez Recettes de conteneurs dans le volet de navigation.
2. Cochez la case à côté de Nom de la recette pour sélectionner la recette que vous souhaitez supprimer.
3. En haut du panneau Recettes du conteneur, dans le menu Actions, choisissez Supprimer la recette.
4. Pour confirmer la suppression, entrez `Delete` dans la case et choisissez Supprimer.

Supprimer la configuration de l'infrastructure

1. Pour consulter la liste des configurations d'infrastructure créées sous votre compte, choisissez Configuration de l'infrastructure dans le volet de navigation.
2. Cochez la case à côté de Nom de la configuration pour sélectionner la configuration d'infrastructure que vous souhaitez supprimer.
3. En haut du panneau Configurations de l'infrastructure, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans la case et choisissez Supprimer.

Supprimer les paramètres de distribution

1. Pour consulter la liste des paramètres de distribution créés sous votre compte, choisissez Paramètres de distribution dans le volet de navigation.
2. Cochez la case à côté de Nom de la configuration pour sélectionner les paramètres de distribution que vous avez créés pour ce didacticiel.
3. En haut du panneau des paramètres de distribution, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans la case et choisissez Supprimer.

Supprimer l'image

Suivez ces étapes pour vérifier que vous avez supprimé toute image créée à partir du pipeline du didacticiel. Il est peu probable que ce didacticiel crée une image à moins que suffisamment de temps ne se soit écoulé depuis la création de votre pipeline pour qu'il s'exécute, conformément au calendrier de construction.

1. Pour consulter la liste des images créées sous votre compte, choisissez Images dans le volet de navigation.
2. Choisissez la version de l'image que vous souhaitez supprimer. Cela ouvre la page des versions de création d'images.
3. Cochez la case à côté de la version pour chaque image que vous souhaitez supprimer. Vous pouvez sélectionner plusieurs versions d'image à la fois.
4. En haut du panneau Versions de création d'images, choisissez Supprimer la version.
5. Pour confirmer la suppression, entrez `Delete` dans la case et choisissez Supprimer.

Tutoriel : Création d'un composant personnalisé avec des paramètres d'entrée

Vous pouvez gérer les composants Image Builder, notamment créer et définir les paramètres des composants, directement depuis la console EC2 Image Builder AWS CLI, depuis ou depuis l'API Image Builder ou SDKs. Dans cette section, nous aborderons la création et l'utilisation de paramètres dans votre composant, ainsi que leur définition via la console Image Builder et AWS CLI les commandes lors de l'exécution.

> **Important**
>
> Les paramètres des composants sont des valeurs en texte brut et sont connectés AWS CloudTrail. Nous vous recommandons d'utiliser AWS Secrets Manager le AWS Systems Manager Parameter Store pour stocker vos secrets. Pour plus d'informations sur Secrets Manager, voir Qu'est-ce que Secrets Manager ? dans le guide de AWS Secrets Manager l'utilisateur. Pour plus d'informations sur AWS Systems Manager Parameter Store, voir AWS Systems Manager Parameter Store dans le guide de AWS Systems Manager l'utilisateur.

Utiliser des paramètres dans le document de votre composant YAML

Pour créer un composant, vous devez fournir un document de composant d'application YAML ou JSON. Le document contient le code qui s'exécute pendant les phases et les étapes que vous définissez pour personnaliser votre image. La recette qui fait référence au composant peut définir les paramètres pour personnaliser les valeurs lors de l'exécution, avec des valeurs par défaut qui prennent effet si le paramètre n'est pas défini sur une valeur spécifique.

Création d'un document de composant avec des paramètres d'entrée

Cette section explique comment définir et utiliser les paramètres d'entrée dans votre document de composant YAML.

Pour créer un document de composant d'application YAML qui utilise des paramètres et exécute des commandes dans vos instances de build ou de test Image Builder, suivez les étapes correspondant à votre système d'exploitation d'image :

Linux

Création d'un document de composant YAML

Utilisez un outil d'édition de fichiers pour créer un fichier de document de composant. Les exemples de documentation utilisent un fichier nommé *hello-world-test.yaml* qui inclut le contenu suivant :

```
# Document Start
#
name: "HelloWorldTestingDocument-Linux"
description: "Hello world document to demonstrate parameters."
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
```

```
            - echo "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

> **Tip**
>
> Utilisez un outil tel que ce validateur YAML en ligne ou une extension YAML lint dans votre environnement de code pour vérifier que votre code YAML est bien formé.

Windows

Création d'un document de composant YAML

Utilisez un outil d'édition de fichiers pour créer un fichier de document de composant. Les exemples de documentation utilisent un fichier nommé *hello-world-test.yaml* qui inclut le contenu suivant :

```
# Document Start
#
name: "HelloWorldTestingDocument-Windows"
description: "Hello world document to demonstrate parameters."
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
```

```
        inputs:
          commands:
            - Write-Host "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

> **Tip**
>
> Utilisez un outil tel que ce validateur YAML en ligne ou une extension YAML lint dans votre environnement de code pour vérifier que votre code YAML est bien formé.

macOS

Création d'un document de composant YAML

Utilisez un outil d'édition de fichiers pour créer un fichier de document de composant. Les exemples de documentation utilisent un fichier nommé *`hello-world-test.yaml`* qui inclut le contenu suivant :

```
# Document Start
#
name: "HelloWorldTestingDocument-macOS"
description: "Hello world document to demonstrate parameters."
```

```
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

> **Tip**
>
> Utilisez un outil tel que ce validateur YAML en ligne ou une extension YAML lint dans votre environnement de code pour vérifier que votre code YAML est bien formé.

Pour plus d'informations sur les phases, les étapes et la syntaxe des documents des composants de AWSTOE l'application, voir Utiliser des documents dans AWSTOE. Pour plus d'informations sur les paramètres et leurs exigences, consultez la Parameters section de la AWSTOE page Définir et référencer les variables.

Création d'un composant à partir du document du composant YAML

Quelle que soit la méthode utilisée pour créer un AWSTOE composant, le document du composant d'application YAML est toujours requis comme référence.

- Pour créer un composant directement à partir de votre document YAML avec la console Image Builder, consultezCréation d'un composant personnalisé depuis la console.
- Pour créer un composant à partir de la ligne de commande à l'aide de la create-component commande Image Builder, consultezCréez un composant personnalisé à partir du AWS CLI. Remplacez le nom du document YAML dans ces exemples par le nom de votre document YAML Hello World ()*`hello-world-test.yaml`*.

Définissez les paramètres des composants dans une recette Image Builder depuis la console

La définition des paramètres des composants fonctionne de la même manière pour les recettes d'images et les recettes de conteneurs. Lorsque vous créez une nouvelle recette ou une nouvelle version d'une recette, vous choisissez les composants à inclure dans les listes Composants de construction et Composants de test. Les listes de composants incluent les composants applicables au système d'exploitation de base que vous avez choisi pour votre image.

Une fois que vous avez sélectionné un composant, il s'affiche dans la section Composants sélectionnés, directement sous les listes de composants. Les options de configuration sont affichées pour chaque composant sélectionné. Si des paramètres d'entrée sont définis pour votre composant, ils sont affichés sous la forme d'une section extensible appelée Paramètres d'entrée.

Les paramètres suivants sont affichés pour chaque paramètre défini pour votre composant :

- Nom du paramètre (non modifiable) : nom du paramètre.
- Description (non modifiable) — Description du paramètre
- Type (non modifiable) : type de données pour la valeur du paramètre.
- Valeur : valeur du paramètre. Si vous utilisez ce composant pour la première fois dans cette recette et qu'une valeur par défaut a été définie pour le paramètre d'entrée, la valeur par défaut apparaît

dans la zone Valeur avec du texte grisé. Si aucune autre valeur n'est saisie, Image Builder utilise la valeur par défaut.

Utilisez un paramètre d'image de base dans votre recette

Lorsque vous créez une recette de personnalisation d'image, il existe plusieurs méthodes pour identifier l'image de base avec laquelle vous commencez. Si vous spécifiez l'identifiant Amazon Machine Image (AMI) pour votre image de base et que cette image de base est mise à jour, son ID AMI peut changer et vous devrez mettre à jour votre recette pour qu'elle corresponde.

Au lieu de modifier votre recette chaque fois que l'ID de l'image de base change, vous pouvez définir un AWS Systems Manager paramètre Parameter Store (paramètre SSM) pour stocker la valeur de l'ID AMI de votre image de base, puis utiliser le paramètre pour spécifier l'image de base dans votre recette. Pour les AMI AWS gérées, vous pouvez utiliser un paramètre public pour la dernière version.

Ce didacticiel explique le processus de création d'un paramètre d'ID AMI et son utilisation dans une recette d'image. Les étapes d'Image Builder décrites dans ce didacticiel sont basées sur la console.

Table des matières

- Étape 1 : rechercher ou créer un paramètre de magasin de paramètres
- Étape 2 : Configuration des autorisations IAM (facultatif)
- Étape 3 : Création d'une recette d'image utilisant le paramètre

Étape 1 : rechercher ou créer un paramètre de magasin de paramètres

Le processus de cette étape dépend du type d'AMI que vous spécifiez pour votre image de base. Pour les AMI AWS gérées, vous pouvez utiliser un paramètre public qui fait référence à la version actuelle. Il se peut que certains paramètres ne soient pas disponibles dans tous les cas Régions AWS.

Pour commencer, ouvrez l'onglet correspondant à votre AMI.

AWS managed AMI

Si votre image de base est une AMI AWS gérée, vous pouvez utiliser des paramètres publics pour spécifier l'ID de l'AMI, plutôt que de créer votre propre paramètre. Pour trouver le paramètre public de votre AMI, consultez la section Découverte des paramètres publics dans le Guide de AWS Systems Manager l'utilisateur.

Custom AMI

Pour créer un paramètre d'ID AMI, suivez les instructions de création de paramètres de magasin de paramètres dans Systems Manager avec la console AWS CLI, ou PowerShell. Fournissez les valeurs suivantes pour vous assurer que la valeur du paramètre est un ID d'AMI.

Niveau de paramètres : `Standard`

Type : `String`

Type de données : Sélectionnez`aws:ec2:image`. Lorsque vous spécifiez ce type, le système valide la valeur saisie pour s'assurer qu'il s'agit d'un ID d'AMI.

Valeur : entrez un ID AMI valide (par exemple,***`ami-1234567890abcdef1`***).

Étape 2 : Configuration des autorisations IAM (facultatif)

Pour utiliser un paramètre de magasin de paramètres de Systems Manager (paramètre SSM), qu'il soit public ou privé, les actions de magasin de paramètres de Systems Manager suivantes doivent être spécifiées dans un rôle IAM, le paramètre étant répertorié en tant que ressource. Le rôle lié au service Image Builder donne l'autorisation d'obtenir des paramètres publics ou d'obtenir ou de mettre à jour des paramètres privés dotés d'un `/imagebuilder/` préfixe. Pour les paramètres privés qui ne possèdent pas ce préfixe, vous pouvez ajouter une autorisation à votre rôle d'exécution.

- `ssm:GetParameter`— Cette action vous permet d'utiliser un paramètre SSM pour spécifier l'image de base de votre recette.
- `ssm:PutParameter`— Cette action vous permet de stocker l'ID de l'AMI de sortie dans un paramètre SSM lors de la distribution. La définition de la politique est identique, mais ce didacticiel n'inclut pas l'action put dans l'exemple de stratégie.

1. Création d'un rôle personnalisé (facultatif)

   Lorsque vous créez un pipeline ou que vous utilisez la commande create-image dans le AWS CLI, vous ne pouvez spécifier qu'un seul rôle d'exécution d'Image Builder. Si vous avez défini un rôle d'exécution du flux de travail Image Builder, vous devez ajouter des autorisations de fonctionnalité supplémentaires à ce rôle. Dans le cas contraire, vous devez créer un nouveau rôle personnalisé incluant les autorisations requises. Si un rôle d'exécution personnalisé est déjà défini, vous pouvez ignorer cette étape.

> **Important**
>
> Nous vous recommandons de ne pas transmettre le rôle AWSServiceRoleForImageBuilderlié au service en tant que rôle d'exécution. Créez plutôt un rôle IAM personnalisé et associez la politique EC2ImageBuilderExecutionPolicy AWS gérée. Cette politique accorde les mêmes autorisations que celles dont Image Builder a besoin pour appeler Services AWS en votre nom. L'utilisation d'un rôle personnalisé vous permet de contrôler totalement les autorisations utilisées par Image Builder. Cela permet également de maintenir vos politiques de contrôle des services (SCP) et vos politiques de contrôle des ressources (RCP) en vigueur pour les opérations qu'Image Builder effectue en votre nom.

Suivez le processus de création d'un rôle pour déléguer des autorisations à un AWS service dans le Guide de Gestion des identités et des accès AWS l'utilisateur.

2. Ajoutez des autorisations à votre rôle personnalisé

   Pour ajouter les autorisations des paramètres SSM à votre rôle personnalisé, suivez la section Mettre à jour la politique d'autorisations pour un processus de rôle dans le Guide de l'Gestion des identités et des accès AWS utilisateur.

   L'exemple de politique suivant montre l'`ssm:GetParameter`action avec un paramètre créé dans votre compte.

   JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "PrivateParameterCustomRole",
   "Effect": "Allow",
   "Action": "ssm:GetParameter",
   "Resource": "arn:aws:ssm:*:111122223333:parameter/parameter-name"
  }
 ]
}
```

Pour plus d'informations sur les ressources de paramètres publics, consultez la section Appeler les paramètres publics de l'AMI dans le guide de AWS Systems Manager l'utilisateur.

Étape 3 : Création d'une recette d'image utilisant le paramètre

1. Ouvrez la console https://console.aws.amazon.com/imagebuilder/EC2 Image Builder à l'adresse.
2. Choisissez Recettes d'images, puis choisissez Créer une recette d'images dans la page de liste.
3. Remplissez la section Image de base comme suit :
   a. Choisissez l'option Utiliser une AMI personnalisée. Cela affiche des champs supplémentaires dans lesquels vous pouvez entrer l'ID d'AMI ou un paramètre SSM contenant l'ID d'AMI.
   b. Choisissez l'option de paramètre SSM.
   c. Dans le champ du paramètre SSM, entrez le nom du paramètre ou le nom de ressource Amazon (ARN) du paramètre que vous avez créé à l'étape 1. Si vous entrez le nom, le préfixe ne figurera pas dans la console.
4. Complétez le reste de la configuration de la recette selon vos besoins.

> **Note**
>
> Si vous définissez l'image parent via d'autres interfaces, telles que le AWS CLI, le nom du paramètre doit comporter le préfixe `ssm:` (par exemple,`ssm:/ImageBuilder-Tutorial/BaseAMI`.

Découvrez des exemples de projets Image Builder sur GitHub

L'équipe EC2 Image Builder gère le référentiel d'exemples d'Image Builder sur GitHub. Le référentiel contient des exemples de projets complets et déployables qui montrent comment implémenter des cas d'utilisation courants avec l'infrastructure en tant que code. Des exemples sont disponibles sous forme de modèles AWS CloudFormation (CloudFormation), d'applications AWS Cloud Development Kit (AWS CDK), de modules Terraform et de documents de composants AWS Task Orchestrator and Executor (AWSTOE).

Chaque échantillon est autonome et possède son propre fichier README qui couvre les prérequis, les étapes de déploiement, le fonctionnement de l'échantillon, les tests et le nettoyage. Les

didacticiels de ce chapitre présentent le processus de création d'images à partir de la console. Les exemples de projets présentent les mêmes concepts exprimés sous forme de code que vous pouvez déployer, inspecter et adapter.

Le référentiel comprend des exemples pour les domaines suivants :

- Pour commencer : le plus petit pipeline d'images complet, fourni à la fois comme CloudFormation modèle et comme AWS CDK application dans l'`quick-start`exemple.
- Use-case kits : des pipelines complets pour des scénarios courants, tels qu'un pipeline AMI doré avec des correctifs planifiés et une actualisation du groupe Auto Scaling. D'autres kits couvrent la distribution d'AMI entre comptes, les builds de VPC privés, les politiques de nettoyage du cycle de vie et les pipelines d'images de base de conteneurs publiés sur Amazon Elastic Container Registry.
- Flux de travail d'images : exemples de flux de travail personnalisés qui ajoutent des portes d'approbation humaines aux étapes de création et de distribution, et s'intègrent AWS Step Functions pour la validation des AMI hors bande.
- Couverture de la plateforme : des pipelines Golden Image pour Windows Server, macOS et Linux.
- Composants et débogage : modèles de AWSTOE composants tels que le redémarrage et la reprise et les secrets de construction. Inclut également un script qui teste les documents des composants localement. Une procédure pas à pas de débogage permet de diagnostiquer une version qui échoue intentionnellement.

Pour la liste complète des exemples, consultez l'index dans le référentiel README sur GitHub. Les échantillons sont sous licence MIT-0, et l'équipe examine les problèmes et les demandes d'extraction du mieux possible.

Utilisez des composants pour personnaliser l'image de votre Image Builder

Image Builder utilise l'application de gestion des composants AWS Task Orchestrator and Executor (AWSTOE) pour orchestrer des flux de travail complexes. Les composants de génération et de test qui fonctionnent avec l' AWSTOE application sont basés sur des documents YAML qui définissent les scripts permettant de personnaliser ou de tester votre image. Pour les images AMI, Image Builder installe les composants et l'application de gestion des AWSTOE composants sur ses instances de génération et de test Amazon EC2. Pour les images de conteneurs, les composants et l'application de gestion des AWSTOE composants sont installés à l'intérieur du conteneur en cours d'exécution.

Image Builder utilise AWSTOE pour effectuer toutes les activités sur l'instance. Aucune configuration supplémentaire n'est requise pour interagir AWSTOE lorsque vous exécutez les commandes d'Image Builder ou que vous utilisez la console Image Builder.

> **Note**
>
> Lorsqu'un composant géré par Amazon atteint la fin de sa durée de support, il n'est plus maintenu. Environ quatre semaines avant que cela ne se produise, tous les comptes utilisant le composant reçoivent une notification et une liste des recettes concernées sur leur compte depuis leur compte Tableau de bord AWS Health. Pour en savoir plus AWS Health, consultez le Guide de AWS Health l'utilisateur.

Étapes du flux de travail pour créer une nouvelle image

Le flux de travail Image Builder pour créer de nouvelles images comprend les deux étapes distinctes suivantes.

1. Étape de génération (pré-capture d'écran) : pendant la phase de génération, vous apportez des modifications à l'instance de génération Amazon EC2 qui exécute votre image de base, afin de créer la base de référence pour votre nouvelle image. Par exemple, votre recette peut inclure des composants qui installent une application ou modifient les paramètres du pare-feu du système d'exploitation.

   Les phases suivantes de votre document de composant s'exécutent pendant la phase de génération :

- build
- valider

Une fois cette étape terminée avec succès, Image Builder crée un instantané ou une image de conteneur qu'il utilise pour la phase de test et au-delà.

2. Étape de test (après la capture d'écran) — Au cours de la phase de test, il existe certaines différences entre les images qui créent des AMI et les images de conteneurs. Pour les flux de travail AMI, Image Builder lance une instance EC2 à partir de l'instantané qu'il a créé lors de la dernière étape de la phase de génération. Des tests sont exécutés sur la nouvelle instance pour valider les paramètres et s'assurer que l'instance fonctionne comme prévu. Pour les flux de travail de conteneurs, les tests s'exécutent sur la même instance que celle utilisée pour la création.

   La phase suivante de votre document de composant s'exécute pour chaque composant inclus dans la recette pendant la phase de test de création de l'image :

   - test

   Cette phase de composant s'applique à la fois aux types de composants Build et Test. Une fois cette étape terminée avec succès, Image Builder peut créer et distribuer votre image finale à partir de l'instantané ou de l'image du conteneur.

> **Note**
>
> Alors que la structure de l' AWSTOE application vous permet de définir de nombreuses phases dans un document de composant, Image Builder applique des règles strictes concernant les phases à exécuter et les étapes pendant lesquelles il les exécute. Pour qu'un composant soit exécuté pendant la phase de création de l'image, le document du composant doit définir au moins l'une des phases suivantes : `build` ou`validate`. Pour qu'un composant soit exécuté pendant la phase de test d'image, le document du composant doit définir la `test` phase, et aucune autre phase.
> Comme Image Builder exécute les étapes de manière indépendante, le chaînage des références dans les documents de composants ne peut pas dépasser les limites des étapes. Vous ne pouvez pas enchaîner une valeur entre une phase exécutée pendant la phase de génération et une phase exécutée pendant la phase de test. Vous pouvez toutefois définir des paramètres d'entrée pour la cible prévue et transmettre des valeurs via la ligne de commande. Pour plus d'informations sur la définition des paramètres des composants dans

vos recettes Image Builder, consultezTutoriel : Création d'un composant personnalisé avec des paramètres d'entrée.

Pour faciliter le dépannage de votre instance de build ou de test, AWSTOE créez un dossier journal contenant le document d'entrée et les fichiers journaux permettant de suivre ce qui se passe chaque fois qu'un composant est exécuté. Si vous avez configuré un compartiment Amazon S3 dans la configuration de votre pipeline, les journaux y sont également écrits. Pour plus d'informations sur les documents YAML et la sortie des journaux, consultezUtilisez la commande AWSTOE cadre de documentation sur les composants pour les composants personnalisés.

> **Tip**
>
> Lorsque vous devez suivre de nombreux composants, le balisage vous permet d'identifier un composant ou une version spécifique en fonction des balises que vous lui avez attribuées. Pour plus d'informations sur le balisage de vos ressources à l'aide des commandes d'Image Builder AWS CLI, consultez la Balisage des ressources section de ce guide.

Cette section explique comment répertorier, afficher, créer et importer des composants à l'aide de la console Image Builder ou des commandes du AWS CLI.

Rubriques

- Répertorier et afficher les détails des composants
- Utilisation AWS Marketplace composants pour personnaliser votre image
- Utilisez des composants gérés pour personnaliser l'image de votre Image Builder
- Développez des composants personnalisés pour votre image Image Builder
- Comment Image Builder utilise AWS Task Orchestrator and Executor application pour gérer les composants

Répertorier et afficher les détails des composants

Cette section explique comment trouver des informations et afficher des détails sur les composants que vous utilisez dans vos recettes EC2 Image Builder.

Détails des composants

- Liste des composants d'Image Builder
- Répertorier les versions de compilation des composants à partir du AWS CLI
- Obtenez des informations sur les composants à partir du AWS CLI
- Obtenez des informations détaillées sur la politique relative aux composants à partir du AWS CLI

Liste des composants d'Image Builder

Vous pouvez utiliser l'une des méthodes suivantes pour répertorier et filtrer les composants d'Image Builder.

Console de gestion AWS

Pour afficher la liste des composants du Console de gestion AWS, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Sélectionnez Composants dans le volet de navigation. Par défaut, Image Builder affiche la liste des composants appartenant à votre compte.
3. Vous pouvez éventuellement filtrer en fonction de la propriété des composants. Pour voir les composants auxquels vous n'êtes pas propriétaire, mais auxquels vous avez accès, développez la liste déroulante des types de propriétaires et sélectionnez l'une des valeurs. La liste des types de propriétaires se trouve dans la barre de recherche, à côté de la zone de texte de recherche. Vous pouvez choisir parmi les valeurs suivantes :
   - AWS Marketplace— Composants directement associés à un abonnement à un AWS Marketplace produit.
   - Démarrage rapide (géré par Amazon) : composants accessibles au public créés et gérés par Amazon.
   - M'appartenant : composants que vous avez créés. Il s'agit de la sélection par défaut.
   - Partagés avec moi : composants que d'autres personnes ont créés et partagés avec vous depuis leur compte.
   - Géré par des tiers : composants appartenant à un tiers et auxquels vous vous êtes abonné AWS Marketplace.

AWS CLI

L'exemple suivant montre comment utiliser la list-components commande pour renvoyer la liste des composants Image Builder appartenant à votre compte.

```
aws imagebuilder list-components
```

Vous pouvez éventuellement filtrer en fonction de la propriété des composants. L'attribut owner définit à qui appartient les composants que vous souhaitez répertorier. Par défaut, cette demande renvoie la liste des composants appartenant à votre compte. Pour filtrer les résultats par propriétaire de composant, spécifiez l'une des valeurs suivantes avec le `--owner` paramètre lorsque vous exécutez la list-components commande.

Valeurs du propriétaire des composants

- `AWSMarketplace`
- `Amazon`
- `Self`
- `Shared`
- `ThirdParty`

Les exemples suivants montrent la list-components commande avec le `--owner` paramètre permettant de filtrer les résultats.

```
aws imagebuilder list-components --owner Self
{
    "requestId": "012a3456-b789-01cd-e234-fa5678b9012b",
    "componentVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.0",
            "name": "sample-component01",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
        },
        {
```

```
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.1",
            "name": "sample-component01",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

```
aws imagebuilder list-components --owner Amazon
```

```
aws imagebuilder list-components --owner Shared
```

```
aws imagebuilder list-components --owner ThirdParty
```

Lorsque vous répertoriez les composants avec la `--owner AWSMarketplace` valeur, vous pouvez filtrer les résultats par code produit. Spécifiez le `productCodes` filtre avec une valeur au format`marketplace:`*`productCodeId`*, comme illustré dans l'exemple suivant :

```
aws imagebuilder list-components --owner AWSMarketplace --filters
 name=productCodes,values=marketplace:examplecode123
```

> **Note**
>
> Image Builder renvoie les AWS Marketplace ARN des composants sous l'`aws-marketplace`espace de noms, avec un propriétaire de. `AWSMarketplace` Lorsque vous appelez get-component un AWS Marketplace composant, la réponse inclut le propriétaire mais omet les données du composant. Les données des composants ne sont disponibles que lorsque l'instance de build appelante a droit au produit.

Répertorier les versions de compilation des composants à partir du AWS CLI

L'exemple suivant montre comment utiliser la list-component-build-versions commande pour répertorier les versions de génération de composants qui ont une version sémantique spécifique. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

```
aws imagebuilder list-component-build-versions --component-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/example-component/1.0.1
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
            "name": "examplecomponent",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "description": "An example component that builds, validates and tests an
 image",
            "changeDescription": "Updated version.",
            "dateCreated": "2020-02-19T18:53:45.940Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

Obtenez des informations sur les composants à partir du AWS CLI

L'exemple suivant montre comment utiliser la get-component commande pour obtenir les détails d'un composant lorsque vous spécifiez son Amazon Resource Name (ARN).

```
aws imagebuilder get-component --component-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1/1
   {
```

```
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11112",
    "component": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
        "name": "examplecomponent",
        "version": "1.0.1",
        "type": "BUILD",
        "platform": "Linux",
        "owner": "123456789012",
        "data": "name: HelloWorldTestingDocument\ndescription: This is hello world
 testing document... etc.\"\n",
        "encrypted": true,
        "dateCreated": "2020-09-24T16:58:24.444Z",
        "tags": {}
    }
}
```

Obtenez des informations détaillées sur la politique relative aux composants à partir du AWS CLI

L'exemple suivant montre comment utiliser la get-component-policy commande pour obtenir les détails de la politique d'un composant lorsque vous spécifiez l'ARN du composant.

```
aws imagebuilder get-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1
```

Utilisation AWS Marketplace composants pour personnaliser votre image

Outre une large sélection d'images créées par des éditeurs de logiciels indépendants (ISV), elle AWS Marketplace propose des composants que vous pouvez utiliser pour personnaliser vos propres images Image Builder. Vous devez vous abonner à ces AWS Marketplace composants avant de pouvoir les utiliser dans votre recette d'image pour créer une nouvelle image.

Lorsque vous spécifiez un AWS Marketplace composant dans une recette d'image, Image Builder valide l'abonnement et effectue des vérifications de dépendance afin de s'assurer que vous disposez des ressources nécessaires pour l'utiliser. Lorsque la validation est réussie, Image Builder crée

des téléchargements sécurisés pour le composant et ses artefacts à utiliser lors de la création de pipelines d'images.

Image Builder découvre les métadonnées du AWS Marketplace catalogue à l'aide de ses propres informations d'identification de service. Vous n'avez donc pas besoin d'accorder à Image Builder des autorisations supplémentaires pour découvrir AWS Marketplace des composants. Toutefois, votre compte doit être abonné à un produit pour pouvoir utiliser ses composants dans une recette.

Découvrez AWS Marketplace components

Vous pouvez découvrir les composants AWS Marketplace logiciels à utiliser dans vos recettes sur la page Découvrir les produits de la console Image Builder, comme suit.

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Dans le volet de navigation, choisissez Découvrir les produits dans la AWS Marketplace section.
3. Sélectionnez l'onglet Components (Composants). Cet onglet répertorie tous les AWS Marketplace produits qui utilisent l'option de livraison qui inclut les composants associés AWS Marketplace.
4. Pour rechercher des produits logiciels spécifiques qui incluent des composants, vous pouvez saisir une partie du nom dans la barre de recherche ou filtrer par `StatusOperating System`,`Publisher`, ou`Categories`. La barre de recherche contient également des commandes de pagination pour vos résultats.

Résultats

Chaque AWS Marketplace produit possède son propre panneau de détails qui comprend les informations suivantes.

Le nom et le logo du AWS Marketplace produit

Le nom du produit logiciel est lié au détail du produit dans AWS Marketplace. Vous pouvez cliquer sur le lien pour en savoir plus sur le produit AWS Marketplace. Vous pouvez également consulter un résumé des options d'abonnement et vous abonner directement à partir des résultats de recherche à l'aide du bouton Afficher les options d'abonnement si vous avez déjà effectué vos recherches.

Version

Il contient la version principale du composant.

Système d'exploitation

Système d'exploitation sur lequel le composant est conçu pour fonctionner.

Editeur

L'éditeur du composant. Ceci est lié à la page détaillée de l'éditeur dans AWS Marketplace. La page détaillée de l'éditeur s'ouvre dans un nouvel onglet de votre navigateur.

Catégories

Une ou plusieurs catégories de AWS Marketplace produits qui s'appliquent au composant.

Statut

Indique si vous êtes abonné à ce produit. Si vous n'êtes pas abonné, vous pouvez choisir Afficher les options d'abonnement pour voir un résumé des options d'abonnement pour le AWS Marketplace produit, et éventuellement vous abonner directement depuis la console Image Builder.

Composants associés

Si une ou plusieurs versions du AWS Marketplace produit sont incluses dans votre abonnement, elles sont affichées dans la section Composants associés. La section est initialement réduite et affiche le nombre de composants associés. Vous pouvez développer la section pour obtenir plus de détails.

> **Note**
>
> Le composant du Center for Internet Security (CIS) associé à leur produit AWS Marketplace d'imagerie n'apparaît pas dans les résultats des produits Discover. Si vous vous abonnez à leur produit d'image, le composant associé est affiché sur la page Abonnements, et en tant que composant tiers dans la boîte de dialogue Créer une recette d'image. Pour plus d'informations sur le composant, consultezcomposants de durcissement CIS.

Abonnez-vous à AWS Marketplace components

Une fois que vous avez trouvé un AWS Marketplace produit contenant des composants que vous souhaitez utiliser dans vos recettes, vous pouvez vous y abonner directement depuis la console Image Builder, comme suit, ou vous pouvez vous y abonner depuis la AWS Marketplace console.

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Dans le volet de navigation, choisissez Découvrir les produits dans la AWS Marketplace section.
3. Sélectionnez l'onglet Components (Composants). Cet onglet répertorie tous les AWS Marketplace produits qui utilisent l'option de livraison qui inclut les composants associés AWS Marketplace.
4. Pour rechercher un AWS Marketplace produit spécifique, entrez une partie de son nom dans la barre de recherche. Si vous connaissez l'éditeur, mais que vous ne connaissez pas le nom exact du produit ni comment l'épeler, vous pouvez également filtrer par `Publisher` pour obtenir une liste des produits disponibles chez l'éditeur.
5. Sélectionnez le produit auquel vous souhaitez vous abonner dans la liste des résultats, puis choisissez Afficher les options d'abonnement. Cela montre un résumé des options d'abonnement pour le AWS Marketplace produit.
6. Sélectionnez S'abonner pour vous abonner au produit sans quitter la console Image Builder. Vous êtes informé que l'abonnement est en cours de traitement. Une fois que vous êtes inscrit, le statut est mis à jour à`Subscribed`.

Pour plus d'informations sur les AWS Marketplace produits auxquels vous êtes actuellement abonné, consultez les étapes relatives à la console décrites dansAWS Marketplace abonnements dans Image Builder.

Utilisez un AWS Marketplace composant d'une recette d'image Image Builder

Vous pouvez utiliser AWS Marketplace des composants dans vos recettes d'images Image Builder de la même manière que vous utilisez les autres types de composants. Pour la plupart des composants associés à un produit d' AWS Marketplace image, la catégorie de propriété est`AWS Marketplace`. Par exemple, pour utiliser un composant de build d'un AWS Marketplace produit auquel vous êtes

abonné, choisissez Ajouter des composants **`AWS Marketplace`** de build, puis sélectionnez-le dans la liste. Cela ouvre un panneau de sélection sur le côté droit de l'interface de la console qui répertorie les AWS Marketplace composants.

> **Note**
>
> Si vous recherchez le composant de durcissement CIS`Third party managed`, sélectionnez-le plutôt dans la liste des propriétaires. `AWS Marketplace`

> **Important**
>
> Une recette qui comprend un ou plusieurs AWS Marketplace composants doit également inclure au moins un composant de construction. Image Builder renvoie une erreur si vous créez une recette qui fait référence à des AWS Marketplace composants mais ne comporte aucun composant de génération. Les recettes de contenants ne prennent pas en charge AWS Marketplace les composants. Pour de plus amples informations, veuillez consulter Contraintes relatives aux recettes des contenants.

Pour plus d'informations sur la sélection, l'organisation et la configuration des paramètres de vos composants, consultezCréation d'une nouvelle version d'une recette imagée.

Utilisez des composants gérés pour personnaliser l'image de votre Image Builder

Les composants gérés sont créés par AWS, parfois en partenariat avec une organisation tierce, telle que le Center for Internet Security (CIS), par exemple. Lorsque vous utilisez des composants gérés dans vos recettes d'images ou de conteneurs, Amazon fournit les dernières versions des composants auxquelles des correctifs et d'autres mises à jour ont été appliqués. Pour obtenir la liste des composants ou pour obtenir des informations sur les composants, consultezRépertorier et afficher les détails des composants.

La liste suivante des composants AWS gérés en vedette inclut un composant que vous pouvez utiliser lorsque vous vous abonnez à des AMI renforcées CIS via le AWS Marketplace.

Composants en vedette

- Installation d'une application de composants gérés par le package du distributeur pour les images Windows d'Image Builder
- composants de durcissement CIS
- Amazon a géré les composants de renforcement STIG pour Image Builder

Installation d'une application de composants gérés par le package du distributeur pour les images Windows d'Image Builder

AWS Systems Manager Distributor vous aide à empaqueter et à publier des logiciels sur des nœuds AWS Systems Manager gérés. Vous pouvez empaqueter et publier vos propres logiciels ou utiliser le distributeur pour rechercher et publier les packages logiciels AWS fournis par les agents. Pour plus d'informations sur Systems Manager Distributor, consultez la section AWS Systems Manager Distributor dans le guide de AWS Systems Manager l'utilisateur.

Composants gérés pour le distributeur

Les composants gérés par Image Builder suivants utilisent AWS Systems Manager Distributor pour installer des packages d'applications sur des instances Windows.

- Le composant `distributor-package-windows` géré utilise AWS Systems Manager Distributor pour installer les packages d'applications que vous spécifiez sur votre instance de génération d'image Windows. Pour configurer les paramètres lorsque vous incluez ce composant dans votre recette, consultezConfiguration `distributor-package-windows` en tant que composant autonome.
- Le `aws-vss-components-windows` composant utilise AWS Systems Manager Distributor pour installer le `AwsVssComponents` package sur votre instance de génération d'image Windows. Pour configurer les paramètres lorsque vous incluez ce composant dans votre recette, consultezConfiguration `aws-vss-components-windows` en tant que composant autonome.

Pour plus d'informations sur l'utilisation des composants gérés dans votre recette Image Builder, consultez Création d'une nouvelle version d'une recette imagée les recettes d'images ou Création d'une nouvelle version d'une recette de contenant les recettes de conteneurs. Pour plus d'informations sur le `AwsVssComponents` package, consultez la section Créer un instantané cohérent avec les applications VSS dans le guide de l'utilisateur Amazon EC2.

Conditions préalables

Avant d'utiliser les composants Image Builder qui s'appuient sur Systems Manager Distributor pour installer des packages d'applications, vous devez vous assurer que les conditions préalables suivantes sont remplies.

- Les composants Image Builder qui utilisent Systems Manager Distributor pour installer des packages d'applications sur votre instance doivent être autorisés à appeler l'API Systems Manager. Avant d'utiliser les composants d'une recette Image Builder, vous devez créer la politique et le rôle IAM qui accordent l'autorisation. Pour configurer les autorisations, consultezConfigurer les autorisations du distributeur Systems Manager.

> **Note**
>
> Image Builder ne prend actuellement pas en charge les packages Systems Manager Distributor qui redémarrent l'instance. Par exemple, les packages`AWSNVMe`,`AWSPVDrivers`, et `AwsEnaNetworkDriver` Distributor redémarrent l'instance et ne sont donc pas autorisés.

Configurer les autorisations du distributeur Systems Manager

Le `distributor-package-windows` composant et les autres composants qui l'utilisent, tels que`aws-vss-components-windows`, nécessitent une autorisation supplémentaire sur l'instance de build pour s'exécuter. L'instance de build doit être en mesure d'appeler l'API Systems Manager pour démarrer une installation de Distributor et demander le résultat.

Suivez ces procédures Console de gestion AWS pour créer une politique et un rôle IAM personnalisés autorisant les composants Image Builder à installer les packages Systems Manager Distributor à partir de l'instance de build.

Étape 1 : créer une politique

Créez une politique IAM pour les autorisations des distributeurs.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation, sélectionnez Politiques, puis Créer une politique.
3. Sur la page Créer une politique, choisissez l'onglet JSON, puis remplacez le contenu par défaut par la politique JSON suivante, en remplaçant la partition, la région et l'ID de compte si nécessaire, ou en utilisant des caractères génériques.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDistributorSendCommand",
            "Effect": "Allow",
            "Action": "ssm:SendCommand",
            "Resource": [
                "arn:aws:ssm:*::document/AWS-ConfigureAWSPackage",
                "arn:aws:ec2:*:111122223333:instance/*"
            ]
        },
        {
            "Sid": "AllowGetCommandInvocation",
            "Effect": "Allow",
            "Action": "ssm:GetCommandInvocation",
            "Resource": "*"
        }
    ]
}
```

4. Choisissez Examiner une stratégie.
5. Pour Nom, saisissez un nom pour identifier la stratégie, par exemple *InvokeDistributor* ou tout autre nom de votre choix.
6. (Facultatif) Pour Description, saisissez une description de l'objectif du rôle.
7. Choisissez Créer une stratégie.

Étape 2 : Création d'un rôle

Créez un rôle IAM pour les autorisations de distributeur.

1. Dans le volet de navigation de la console IAM, choisissez Roles, puis Create role.
2. Sous Select type of trusted entity (Sélectionner le type d'entité approuvée), choisissez Service AWS.
3. Immédiatement sous Choisir le service qui utilisera ce rôle, choisissez EC2, puis Suivant : Autorisations.

4. Sous Sélectionner votre cas d'utilisation, choisissez EC2, puis Suivant : Autorisations.
5. Dans la liste des politiques, cochez la case à côté d'Amazon SSMManaged InstanceCore. (Entrez `SSM` dans la zone de recherche si vous avez besoin d'une liste plus petite.)
6. Dans cette liste de politiques, cochez la case à côté de EC2InstanceProfileForImageBuilder. (Entrez `ImageBuilder` dans la zone de recherche si vous avez besoin d'une liste plus petite.)
7. Choisissez Next: Tags (Suivant : Balises).
8. (Facultatif) Ajoutez une ou plusieurs paires clé-valeur de balise pour organiser, suivre ou contrôler l'accès pour ce rôle, puis choisissez Suivant : Révision.
9. Pour Nom du rôle, entrez un nom pour le rôle, par exemple *`InvokeDistributor`* ou autre, en fonction de vos préférences.
10. (Facultatif) Pour Description du rôle, remplacez le texte par défaut par une description de l'objectif de ce rôle.
11. Sélectionnez Créer un rôle. Le système vous renvoie à la page Rôles.

Étape 3 : associer la politique au rôle

La dernière étape pour configurer vos autorisations de distributeur consiste à associer la politique IAM au rôle IAM.

1. Sur la page Rôles de la console IAM, choisissez le rôle que vous venez de créer. La page Résumé du rôle s'ouvre.
2. Choisissez Attach Policies (Attacher des stratégies).
3. Recherchez la politique que vous avez créée dans la procédure précédente et cochez la case à côté du nom.
4. Choisissez Attach policy (Attacher la politique).

Utilisez ce rôle dans la ressource de configuration de l'infrastructure Image Builder pour toute image incluant des composants utilisant Systems Manager Distributor. Pour de plus amples informations, veuillez consulter Créer une configuration d'infrastructure.

Configuration **`distributor-package-windows`** en tant que composant autonome

Pour utiliser le `distributor-package-windows` composant dans une recette, définissez les paramètres suivants qui configurent le package à installer.

> **Note**
>
> Avant d'utiliser le `distributor-package-windows` composant dans une recette, vous devez vous assurer que toutes les conditions Conditions préalables sont respectées.

- Action (obligatoire) — Spécifiez si vous souhaitez installer ou désinstaller le package. Les valeurs valides sont `Install` et `Uninstall`. La valeur par défaut est. `Install`
- PackageName(Obligatoire) : nom du package du distributeur à installer ou à désinstaller. Pour obtenir la liste des noms de packages valides, consultezTrouver des packages pour distributeurs.
- PackageVersion(Facultatif) — Version du package du distributeur à installer. PackageVersion par défaut, c'est la version recommandée.
- AdditionalArguments(Facultatif) — Chaîne JSON contenant les paramètres supplémentaires à fournir à votre script pour installer, désinstaller ou mettre à jour un package. Pour plus d'informations, consultez AdditionalArguments dans la section AWS:ConfigurePackage Inputs de la page de référence du plug-in du document Systems Manager Command.

Configuration **`aws-vss-components-windows`** en tant que composant autonome

Lorsque vous utilisez le `aws-vss-components-windows` composant dans une recette, vous pouvez éventuellement définir le `PackageVersion` paramètre pour utiliser une version spécifique du `AwsVssComponents` package. Lorsque vous omettez ce paramètre, le composant utilise par défaut la version recommandée du `AwsVssComponents` package.

> **Note**
>
> Avant d'utiliser le `aws-vss-components-windows` composant dans une recette, vous devez vous assurer que toutes les conditions Conditions préalables sont respectées.

Trouver des packages pour distributeurs

Amazon et des tiers fournissent des packages publics que vous pouvez installer avec Systems Manager Distributor.

Pour consulter les packages disponibles dans le Console de gestion AWS, connectez-vous à la AWS Systems Manager console et choisissez Distributor dans le volet de navigation. La page

Distributeur affiche tous les packages mis à votre disposition. Pour plus d'informations sur la liste des packages disponibles avec le AWS CLI, voir Afficher les packages (ligne de commande) dans le guide de AWS Systems Manager l'utilisateur.

Vous pouvez également créer vos propres packages privés de distribution Systems Manager. Pour plus d'informations, voir Création d'un package dans le guide de AWS Systems Manager l'utilisateur.

composants de durcissement CIS

Le Center for Internet Security (CIS) est une organisation communautaire à but non lucratif. Leurs experts en cybersécurité travaillent ensemble pour élaborer des directives de sécurité informatique qui protègent les organisations publiques et privées contre les cybermenaces. Leur ensemble de bonnes pratiques mondialement reconnu, connu sous le nom de CIS Benchmarks, aide les organisations informatiques du monde entier à configurer leurs systèmes en toute sécurité. Pour les articles tendance, les articles de blog, les podcasts, les webinaires et les livres blancs, consultez CIS Insights sur le site Web du Center for Internet Security.

Evaluations CIS

CIS crée et gère un ensemble de directives de configuration, connues sous le nom de benchmarks CIS, qui fournissent les meilleures pratiques de configuration pour des technologies spécifiques, notamment les systèmes d'exploitation, les plateformes cloud, les applications, les bases de données, etc. Les benchmarks CIS sont reconnus comme une norme industrielle par des organisations et des normes telles que PCI DSS, HIPAA, DoD Cloud Computing SRG, FISMA, DFARS et FEDRAMP. Pour en savoir plus, consultez la section CIS Benchmarks sur le site Web du Center for Internet Security.

composants de durcissement CIS

Lorsque vous vous abonnez à une image renforcée CIS dans AWS Marketplace, vous avez également accès au composant de renforcement associé qui exécute un script pour appliquer les directives CIS Benchmarks de niveau 1 à votre configuration. L'organisation CIS possède et entretient les composants de durcissement CIS afin de s'assurer qu'ils reflètent les dernières directives.

> **Note**
>
> Les composants de durcissement CIS ne respectent pas les règles de classement des composants standard des recettes Image Builder. Les composants de renforcement CIS sont

toujours exécutés en dernier pour garantir que les tests de référence s'exécutent par rapport à votre image de sortie.

Amazon a géré les composants de renforcement STIG pour Image Builder

Les guides de mise en œuvre technique de sécurité (STIGs) sont des normes de renforcement de la configuration créées par la Defense Information Systems Agency (DISA) pour sécuriser les systèmes d'information et les logiciels. Pour rendre vos systèmes conformes aux normes STIG, vous devez installer, configurer et tester différents paramètres de sécurité.

Image Builder fournit des composants de renforcement STIG pour vous aider à créer plus efficacement des images conformes aux normes STIG de base. Ces composants STIG détectent les erreurs de configuration et exécutent un script de correction. L'utilisation de composants conformes aux normes STIG est gratuite.

Important

À quelques exceptions près, sauf indication contraire dans les paramètres, les composants de renforcement STIG n'installent pas de packages tiers. Si des packages tiers sont déjà installés sur l'instance et STIGs qu'Image Builder prend en charge certains d'entre eux pour ce package, le composant de renforcement les applique.

Cette page répertorie tous les STIG pris en charge par Image Builder et appliqués aux instances EC2 lancées par Image Builder lorsque vous créez et testez une nouvelle image. Si vous souhaitez appliquer des paramètres STIG supplémentaires à votre image, vous pouvez créer un composant personnalisé pour la configurer. Pour plus d'informations sur les composants personnalisés et sur la façon de les créer, consultezUtilisez des composants pour personnaliser l'image de votre Image Builder.

Lorsque vous créez une image, les composants de durcissement STIG enregistrent s'ils STIGs sont pris en charge ou ignorés. Nous vous recommandons de consulter les journaux Image Builder de vos images qui utilisent des composants de renforcement STIG. Pour plus d'informations sur la façon d'accéder aux journaux d'Image Builder et de les consulter, consultezRésoudre les problèmes liés aux builds de pipelines.

Niveaux de conformité

- Élevé (Catégorie I)

  Risque le plus grave. Inclut toute vulnérabilité pouvant entraîner une perte de confidentialité, de disponibilité ou d'intégrité.

- Moyen (Catégorie II)

  Inclut toute vulnérabilité susceptible d'entraîner une perte de confidentialité, de disponibilité ou d'intégrité, mais les risques peuvent être atténués.

- Faible (catégorie III)

  Toute vulnérabilité qui dégrade les mesures de protection contre la perte de confidentialité, de disponibilité ou d'intégrité.

Rubriques

- Composants de renforcement Windows STIG
- Journal d'historique des versions de STIG pour Windows
- Composants de renforcement Linux STIG
- Journal d'historique des versions de STIG pour Linux
- Composant de validation de conformité SCAP

Composants de renforcement Windows STIG

AWSTOE Les composants de renforcement de Windows STIG sont conçus pour les serveurs autonomes et appliquent une stratégie de groupe locale. Les composants de renforcement conformes aux STIG installent et mettent à jour les certificats du ministère de la Défense (DoD). Ils suppriment également les certificats inutiles pour maintenir la conformité aux STIG. Actuellement, les lignes de base STIG sont prises en charge pour les versions suivantes de Windows Server : 2012 R2, 2016, 2019, 2022 et 2025.

Cette section répertorie les paramètres actuels de chacun des composants de renforcement de Windows STIG, suivis d'un journal de l'historique des versions.

Windows STIG Niveau faible (catégorie III)

La liste suivante contient les paramètres STIG que le composant de renforcement applique à votre infrastructure. Si aucun paramètre pris en charge n'est applicable à votre infrastructure, le composant de renforcement ignore ce paramètre et passe à autre chose. Par exemple, certains paramètres STIG peuvent ne pas s'appliquer aux serveurs autonomes. Les politiques spécifiques à l'organisation peuvent également affecter les paramètres appliqués par le composant de renforcement, par exemple l'obligation pour les administrateurs de revoir les paramètres du document.

Pour obtenir la liste complète de Windows STIGs, consultez la bibliothèque de STIGs documents. Pour plus d'informations sur l'affichage de la liste complète, veuillez consulter la rubrique Outils d'affichage STIG.

- Windows Server 2025 STIG version 1, version 1

  V-278082, V-278083, V-278084, V-278085, V-278098, V-278104, V-278110 et V-278231
- Windows Server 2022 STIG version 2 version 7

  V-254335, V-254336, V-254337, V-254338, V-254351, V-254357, V-254363 et V-254481
- Windows Server 2019 STIG version 3 version 7

  V-205691, V-205819, V-205858, V-205859, V-205860, V-205870, V-205871 et V-205923
- Windows Server 2016 STIG version 2 version 10

  V-224916, V-224917, V-224918, V-224919, V-224931, V-224942 et V-225060
- Windows Server 2012 R2 MS STIG version 3 version 5

  V-225250, V-225318, V-225319, V-225324, V-225327, V-225328, V-225330, V-225331, V-225332, V-225333, V-225334, V-225335, V-225336, V-225342, V-225343, V-225355, V-225357, V-225358, V-225359, V-225360, V-225362, V-225363, V-225376, V-225392, V-225394, V-225412, V-225459, V-225460, V-225462, V-225468, V-225473, V-225476, V-225479, V-225480, V-225481, V-225482, V-225483, V-225484, V-225485, V-225487, V-225488, V-225489, V-225490, V-225511, V-225514, V-225525, V-225526, V-225536 et V-225537
- Microsoft .NET Framework 4.0 STIG version 2 version 7

  Aucun paramètre STIG n'est appliqué au Microsoft .NET Framework pour les vulnérabilités de catégorie III.
- Pare-feu Windows STIG version 2, version 2

V-241994, V-241995, V-241996, V-241999, V-242000, V-242001, V-242006, V-242007 et V-242008

- Internet Explorer 11 STIG version 2 version 6

  V-223056 et V-223078

- Microsoft Edge STIG version 2 version 4 (Windows Server 2022 et 2025 uniquement)

  V-235727, V-235731, V-235751, V-235752 et V-235765

- Microsoft Defender STIG version 2, version 7

  Aucun paramètre STIG n'est appliqué aux vulnérabilités de catégorie III de Microsoft Antivirus.

Windows STIG Niveau moyen (catégorie II)

La liste suivante contient les paramètres STIG que le composant de renforcement applique à votre infrastructure. Si aucun paramètre pris en charge n'est applicable à votre infrastructure, le composant de renforcement ignore ce paramètre et passe à autre chose. Par exemple, certains paramètres STIG peuvent ne pas s'appliquer aux serveurs autonomes. Les politiques spécifiques à l'organisation peuvent également affecter les paramètres appliqués par le composant de renforcement, par exemple l'obligation pour les administrateurs de revoir les paramètres du document.

Pour obtenir la liste complète de Windows STIGs, consultez la bibliothèque de STIGs documents. Pour plus d'informations sur l'affichage de la liste complète, veuillez consulter la rubrique Outils d'affichage STIG.

> **Note**
>
> Les composants de renforcement Windows STIG Medium incluent tous les paramètres STIG répertoriés qui AWSTOE s'appliquent aux composants de renforcement faible de Windows STIG, en plus des paramètres STIG répertoriés spécifiquement pour les vulnérabilités de catégorie II.

- Windows Server 2025 STIG version 1, version 1

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

V-278015, V-278016, V-278019, V-278020, V-278021, V-278022, V-278023, V-278024, V-278025, V-278026, V-278033, V-278034, V-278035, V-278036, V-278037, V-278038, V-278039, V-278047, V-278048, V-278049, V-278050, V-278051, V-278052, V-278053, V-278054, V-278055, V-278056, V-278057, V-278058, V-278059, V-278060, V-278061, V-278062, V-278063, V-278064, V-278065, V-278066, V-278067, V-278068, V-278069, V-278070, V-278071, V-278072, V-278073, V-278074, V-278075, V-278076, V-278077, V-278078, V-278079, V-278080, V-278088, V-278089, V-278091, V-278092, V-278093, V-278094, V-278095, V-278096, V-278097, V-278102, V-278103, V-278105, V-278106, V-278107, V-278108, V-278109, V-278111, V-278112, V-278113, V-278114, V-278115, V-278116, V-278117, V-278118, V-278119, V-278120, V-278122, V-278123, V-278124, V-278126, V-278127, V-278129, V-278130, V-278131, V-278165, V-278168, V-278169, V-278170, V-278171, V-278174, V-278180, V-278181, V-278182, V-278183, V-278184, V-278185, V-278187, V-278188, V-278189, V-278192, V-278193, V-278194, V-278195, V-278198, V-278199, V-278200, V-278201, V-278202, V-278203, V-278204, V-278205, V-27820626, V-278209, V-278210, V-278211, V-278212, V-278213, V-278214, V-278218, V-278220, V-278221, V-278222, V-278223, V-278226, V-278227, V-278228, V-278229, V-278230, V-278232, V-278233, V-278234, V-278235, V-278236, V-278237, V-278238, V-278239, V-278240, V-278241, V-278243, V-278244, V-278245, V-278247, V-278248, V-278249, V-278251, V-278252, V-278253, V-278254, V-278255, V-278256, V-278257, V-278258, V-278259, V-278260, V-278261, V-278262, V-279916, V-279917, V-279918, V-279919, V-279920, V-279921, V-279922 et V-279923

- Windows Server 2022 STIG version 2 version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-254247, V-254269, V-254270, V-254271, V-254272, V-254273, V-254274, V-254276, V-254277, V-254278, V-254285, V-254286, V-254287, V-254288, V-254289, V-254290, V-254291, V-254292, V-254296, V-254297, V-254298, V-254299, V-254300, V-254301, V-254302, V-254303, V-254304, V-254305, V-254307, V-254309, V-254311, V-254313, V-254314, V-254315, V-254316, V-254319, V-254320, V-254321, V-254322, V-254323, V-254324, V-254325, V-254326, V-254327, V-254328, V-254329, V-254330, V-254331, V-254332, V-254334, V-254339, V-254341, V-254342, V-254344, V-254345, V-254346, V-254347, V-254348, V-254349, V-254350, V-254355, V-254356, V-254358, V-254359, V-254360, V-254361, V-254362, V-254365, V-254366, V-254367, V-254368, V-254369, V-254370, V-254371, V-254372, V-254373, V-254375, V-254376, V-254377, V-254379, V-254380, V-254382, V-254383, V-254384, V-254431, V-254434, V-254435, V-254436, V-254438, V-254439, V-254440, V-254442, V-254443, V-254444, V-254445, V-254447, V-254448, V-254449, V-254450, V-254451, V-254452, V-254453, V-254454, V-254456, V-254459, V-254460, V-254461, V-254462,

V-254463, V-254464, V-254468, V-254470, V-254471, V-254472, V-254473, V-254476, V-254477, V-254478, V-254479, V-254480, V-254482, V-254483, V-254484, V-254485, V-254486, V-254487, V-254488, V-254489, V-254491, V-254493, V-254494, V-254495, V-254497, V-254498, V-254499, V-254501, V-254502, V-254503, V-254504, V-254505, V-254506, V-254508, V-254509, V-254510, V-254511, V-254512, V-278942, V-278943, V-278944, V-278945, V-278946, V-278947, V-278948 et V-278949

- Windows Server 2019 STIG version 3 version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-205625, V-205626, V-205627, V-205629, V-205630, V-205633, V-205634, V-205636, V-205637, V-205638, V-205639, V-205640, V-205641, V-205642, V-205643, V-205644, V-205648, V-205649, V-205650, V-205651, V-205652, V-205655, V-205656, V-205659, V-205660, V-205662, V-205671, V-205672, V-205673, V-205675, V-205676, V-205678, V-205679, V-205680, V-205681, V-205682, V-2055682, V-2055682 V-205684, V-205685, V-205686, V-205687, V-205688, V-205689, V-205690, V-205692, V-205693, V-205694, V-205697, V-205698, V-205708, V-205709, V-205712, V-205714, V-205716, V-205716, V-205709, V-205712, V-205714, V-205716, V-205716, V-205709, V-205712, V-205714, V-205716, V-205716, V-205709, V-205712, V-205714, V-205716, V-205716, V-205709, V-205712, V-205714, V-205716, V-5717, V-205718, V-205719, V-205720, V-205722, V-205730, V-205731, V-205733, V-205747, V-205748, V-205749, V-205751, V-205752, V-205754, V-205756, V-205758, V-205759, V-205760, V-205761, V-205762, V-205763, V-205764, V-205765, V-205766, V-205767, V-205768, V-205769, V-205770, V-205771, V-205772, V-205773, V-205774, V-205776, V-205777, V-205778, V-205779, V-20577 80, V-205781, V-205782, V-205783, V-205784, V-205795, V-205796, V-205797, V-205798, V-205801, V-205808, V-205809, V-205810, V-205811, V-205812, V-205814, V-205815, V-205816, V-205816, V-205811, V-205812, V-205813, V-205814, V-205815, V-205816, V-205816, V-205811 5817, V-205821, V-205822, V-205823, V-205824, V-205825, V-205826, V-205827, V-205828, V-205830, V-205832, V-205833, V-205835, V-205836, V-205837, V-205838, V-205842, V-205861, V-205863, V-205865, V-205866, V-205867, V-205868, V-205869, V-205872, V-205873, V-205874, V-205909, V-205910, V-205911, V-205912, V-205915, V-205916, V-205917, V-205918, V-205920, V-205921, V-205922, V-205925, V-257503, V-278934, V-278935, V-278936, V-278937, V-278938, V-278939, V-278940 et V-278941

- Windows Server 2016 STIG version 2 version 10

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

V-224850, V-224851, V-224852, V-224853, V-224854, V-224855, V-224856, V-224857, V-224858, V-224859, V-224866, V-224867, V-224868, V-224869, V-224870, V-224871, V-224872, V-224873, V-224877, V-224877, V-224871 V-224879, V-224880, V-224881, V-224882, V-224883, V-224884, V-224885, V-224886, V-224888, V-224890, V-224892, V-224893, V-224894, V-224895, V-224896, V-224897, V-224900, V-224901, V-224902, V-224902, V-224896 903, V-224904, V-224905, V-224906, V-224907, V-224908, V-224909, V-224910, V-224911, V-224912, V-224913, V-224914, V-224915, V-224920, V-224921, V-224922, V-224924, V-224925, V-224926, V-224927, V-224928, V-224929, V-224930, V-224935, V-224936, V-224937, V-224938, V-224939, V-224940, V-224941, V-224943, V-224944, V-224945, V-224946, V-224947, V-224948, V-224949, V-224944 V-224952, V-224953, V-224955, V-224956, V-224957, V-224959, V-224960, V-224962, V-224963, V-224965, V-224966, V-224967, V-224968, V-224969, V-224987, V-224988, V-224989, V-224995, V-224996, V-224997, V-224998, V-224999, V-225000, V-225001, V-225002, V-225003, V-225004, V-225005, V-225008, V-225009, V-225010, V-225011, V-225013, V-225014, V-225015, V-225016, V-225017, V-225018, V-225019, V-225020, V-225021, V-225022, V-225023, V-225024, V-225026, V-225027, V-225028, V-225029, V-225030, V-225031, V-225032, V-225033, V-225034, V-225035, V-225038, V-225039, V-225040, V-225041, V-225042, V-225043, V-225047, V-225049, V-225050, V-225051, V-225052, V-225055, V-225056, V-225057, V-225058, V-225061, V-225062, V-225063, V-225064, V-225065, V-225066, V-225067, V-225068, V-225070, V-225072, V-225073, V-225074, V-225076, V-225077, V-225078, V-225080, V-225081, V-225082, V-225083, V-225084, V-225085, V-225086, V-225087, V-225088, V-225089, V-225092, V-225093 et V-257502

- Windows Server 2012 R2 MS STIG version 3 version 5

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-225239, V-225259, V-225260, V-225261, V-225263, V-225264, V-225265, V-225266, V-225267, V-225268, V-225269, V-225270, V-225271, V-225272, V-225273, V-225275, V-225276, V-225277, V-225278, V-225279, V-225280, V-225281, V-225282, V-225283, V-225284, V-225285, V-225286, V-225287, V-225288, V-225289, V-225290, V-225291, V-225292, V-225293, V-225294, V-225295, V-225296, V-225297, V-225298, V-225299, V-225300, V-225301, V-225302, V-225303, V-225304, V-225305, V-225314, V-225315, V-225316, V-225317, V-225325, V-225326, V-225329, V-225337, V-225338, V-225339, V-225340, V-225341, V-225344, V-225345, V-225346, V-225347, V-225348, V-225349, V-225350, V-225351, V-225352, V-225353, V-225356, V-225367, V-225368, V-225369, V-225370, V-225371, V-225372, V-225373, V-225374, V-225375, V-225377, V-225378, V-225379, V-225380, V-225381, V-225382, V-225383, V-225384, V-225385, V-225386, V-225389, V-225391, V-225393, V-225395, V-225397, V-225398, V-225400, V-225401, V-225402, V-225404, V-225405,

V-225406, V-225407, V-225408, V-225409, V-225410, V-225411, V-225413, V-225414, V-225415, V-225441, V-225442, V-225443, V-225448, V-225452, V-225453, V-225454, V-225455, V-225456, V-225457, V-225458, V-225461, V-225463, V-225464, V-225469, V-225470, V-225471, V-225472, V-225474, V-225475, V-225477, V-225478, V-225486, V-225494, V-225500, V-225501, V-225502, V-225503, V-225504, V-225506, V-225508, V-225509, V-225510, V-225513, V-225515, V-225516, V-225517, V-225518, V-225519, V-225520, V-225521, V-225522, V-225523, V-225524, V-225527, V-225528, V-225529, V-225530, V-225531, V-225532, V-225533, V-225534, V-225535, V-225538, V-225539, V-225540, V-225541, V-225542, V-225543, V-225544, V-225545, V-225546, V-225548, V-225549, V-225550, V-225551, V-225553, V-225554, V-225555, V-225557, V-225558, V-225559, V-225560, V-225561, V-225562, V-225563, V-225564, V-225565, V-225566, V-225567, V-225568, V-225569, V-225570, V-225571, V-225572, V-225573 et V-225574

- Microsoft .NET Framework 4.0 STIG version 2 version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-225223, V-225230, V-225235 et V-225238

- Pare-feu Windows STIG version 2, version 2

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-241989, V-241990, V-241991, V-241993, V-241998, V-242003, V-242004 et V-242005

- Internet Explorer 11 STIG version 2 version 6

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-223015, V-223017, V-223018, V-223019, V-223020, V-223021, V-223022, V-223023, V-223024, V-223025, V-223026, V-223027, V-223028, V-223030, V-223031, V-223032, V-223033, V-223034, V-223035, V-223036, V-223037, V-223038, V-223039, V-223040, V-223041, V-223042, V-223043, V-223044, V-223045, V-223046, V-223048, V-223049, V-223050, V-223051, V-223052, V-223053, V-223054, V-223055, V-223057, V-223058, V-223059, V-223060, V-223061, V-223062, V-223063, V-223064, V-223065, V-223066, V-223067, V-223068, V-223069, V-223070, V-223071, V-223072, V-223073, V-223074, V-223075, V-223076, V-223077, V-223079, V-223080, V-223081, V-223082, V-223083, V-223084, V-223085, V-223086, V-223087, V-223088, V-223089, V-223090, V-223091, V-223092, V-223093, V-223094, V-223095, V-223096, V-223097, V-223098, V-223099, V-223100, V-223101, V-223102, V-223103, V-223104, V-223105, V-223106, V-223107, V-223108, V-223109,

V-223110, V-223111, V-223112, V-223113, V-223114, V-223115, V-223116, V-223116, V-223112 117, V-223118, V-223119, V-223120, V-223121, V-223123, V-223125, V-223126, V-223127, V-223128, V-223129, V-223130, V-223131, V-223132, V-223133, V-223134, V-223135, V-223136, V-223137, V-223138, V-223139, V-223140, V-223141, V-223142, V-223143, V-223144, V-223145, V-223146, V-223147, V-223148, V-223149, V-250540 et V-250541

- Microsoft Edge STIG version 2 version 4 (Windows Server 2022 et 2025 uniquement)

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-235720, V-235721, V-235723, V-235724, V-235725, V-235726, V-235728, V-235729, V-235730, V-235732, V-235733, V-235734, V-235735, V-235736, V-235737, V-235738, V-235739, V-235740, V-235741, V-235742, V-235743, V-235744, V-235745, V-235746, V-235747, V-235748, V-235749, V-235750, V-235754, V-235756, V-235760, V-235761, V-235763, V-235764, V-235766, V-235767, V-235768, V-235769, V-235770, V-235771, V-235772, V-235773, V-235774 et V-246736

- Microsoft Defender STIG version 2, version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible), ainsi que :

  V-213427, V-213429, V-213430, V-213431, V-213432, V-213433, V-213434, V-213435, V-213436, V-213437, V-213438, V-213439, V-213440, V-213441, V-213442, V-213443, V-213444, V-213446, V-213447, V-213448, V-213448, V-213448 V-213450, V-213451, V-213454, V-213455, V-213456, V-213457, V-213458, V-213459, V-213460, V-213461, V-213462, V-213463, V-213464, V-213465, V-213466, V-278647, V-278648, V-278649, V-278650, V-278651, V-278652, V-278652 278653, V-278654, V-278655, V-278656, V-278658, V-278659, V-278660, V-278661, V-278662, V-278668, V-278669, V-278672, V-278674, V-278675, V-278676, V-278677, V-278678, V-278679, V-278680 et V-278863

Windows STIG Niveau élevé (catégorie I)

La liste suivante contient les paramètres STIG que le composant de renforcement applique à votre infrastructure. Si aucun paramètre pris en charge n'est applicable à votre infrastructure, le composant de renforcement ignore ce paramètre et passe à autre chose. Par exemple, certains paramètres STIG peuvent ne pas s’appliquer aux serveurs autonomes. Les politiques spécifiques à l'organisation peuvent également affecter les paramètres appliqués par le composant de renforcement, par exemple l'obligation pour les administrateurs de revoir les paramètres du document.

Pour obtenir la liste complète de Windows STIGs, consultez la bibliothèque de STIGs documents. Pour plus d'informations sur l'affichage de la liste complète, veuillez consulter la rubrique Outils d'affichage STIG.

> **Note**
>
> Les composants de renforcement élevé de Windows STIG incluent tous les paramètres STIG répertoriés qui AWSTOE s'appliquent aux composants de renforcement Windows STIG Low et Windows STIG Medium, en plus des paramètres STIG répertoriés spécifiquement pour les vulnérabilités de catégorie I.

- Windows Server 2025 STIG version 1, version 1

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

  V-278040, V-278099, V-278100, V-278101, V-278121, V-278125, V-278128, V-278196, V-278215, V-278216, V-278217, V-278219, V-278225, V-278242, V-278246 et V-278250

- Windows Server 2022 STIG version 2 version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

  V-254250, V-254293, V-254352, V-254353, V-254354, V-254374, V-254378, V-254381, V-254446, V-254466, V-254467, V-254469, V-254474, V-254475, V-254492, V-254496 et V-254500

- Windows Server 2019 STIG version 3 version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

  V-205653, V-205654, V-205663, V-205711, V-205713, V-205724, V-205725, V-205750, V-205753, V-205757, V-205802, V-205804, V-205805, V-205806, V-205849, V-205908, V-205914 et V-205919

- Windows Server 2016 STIG version 2 version 10

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

V-224831, V-224874, V-224932, V-224933, V-224934, V-224954, V-224958, V-224961, V-225025, V-225045, V-225046, V-225048, V-225053, V-225054, V-225071, V-225079 et V-225091

- Windows Server 2012 R2 MS STIG version 3 version 5

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

  V-225274, V-225354, V-225364, V-225365, V-225366, V-225390, V-225396, V-225399, V-225444, V-225449, V-225491, V-225492, V-225493, V-225496, V-225497, V-225498, V-225505, V-225507, V-225547, V-225552 et V-225556

- Microsoft .NET Framework 4.0 STIG version 2 version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) dans le Microsoft .NET Framework. Aucun paramètre STIG supplémentaire ne s'applique pour les vulnérabilités de catégorie I.

- Pare-feu Windows STIG version 2, version 2

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

  V-241992, V-241997 et V-242002

- Internet Explorer 11 STIG version 2 version 6

  V-252910

- Microsoft Edge STIG version 2 version 4 (Windows Server 2022 et 2025 uniquement)

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

  V-235758 and V-235759

- Microsoft Defender STIG version 2, version 7

  Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégories II et III (moyennes et faibles), ainsi que :

  V-213426, V-213428, V-213452 et V-213453

Journal d'historique des versions de STIG pour Windows

Cette section enregistre l'historique des versions des composants de renforcement de Windows pour les mises à jour trimestrielles de STIG. Pour voir les modifications et les versions publiées pendant un trimestre, choisissez le titre pour développer les informations.

Modifications apportées au premier trimestre 2026 - 10/03/2026 :

Ajout du support pour Windows Server 2025 et mise à jour de toutes les versions applicables STIGs au premier trimestre 2026.

Fenêtres STIG-BUILD

- Windows Server 2025 STIG version 1, version 1
- Windows Server 2022 STIG version 2 version 7
- Windows Server 2019 STIG version 3 version 7
- Windows Server 2016 STIG version 2 version 10
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 STIG version 2 version 7
- Pare-feu Windows STIG version 2, version 2
- Internet Explorer 11 STIG version 2 version 8
- Microsoft Edge STIG version 2 version 4 (Windows Server 2022 et 2025 uniquement)

Modifications apportées au troisième trimestre 2025 – 09/04/2025 (aucune modification) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du troisième trimestre 2025.

Modifications apportées au deuxième trimestre 2025 – 26/06/2025 :

Versions STIG mises à jour et STIG appliquées pour la version du deuxième trimestre 2025 comme suit :

STIG-Build-Windows-Low version 2025.2.x

- Windows Server 2022 STIG version 2 version 4
- Windows Server 2019 STIG Version 3 Release 4

- Windows Server 2016 STIG version 2 version 10
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 6
- Pare-feu Windows STIG version 2, version 2
- Internet Explorer 11 STIG version 2 version 5
- Microsoft Edge STIG Version 2 Release 2 (Windows Server 2022 seulement)

STIG-Build-Windows-Medium version 2025.2.x

- Windows Server 2022 STIG version 2 version 4
- Windows Server 2019 STIG Version 3 Release 4
- Windows Server 2016 STIG version 2 version 10
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 6
- Pare-feu Windows STIG version 2, version 2
- Internet Explorer 11 STIG version 2 version 5
- Microsoft Edge STIG Version 2 Release 2 (Windows Server 2022 seulement)
- Defender STIG Version 2 Release 4

STIG-Build-Windows-High version 2025.2.x

- Windows Server 2022 STIG version 2 version 4
- Windows Server 2019 STIG Version 3 Release 4
- Windows Server 2016 STIG version 2 version 10
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 6
- Pare-feu Windows STIG version 2, version 2
- Internet Explorer 11 STIG version 2 version 5
- Microsoft Edge STIG Version 2 Release 2 (Windows Server 2022 seulement)
- Defender STIG Version 2 Release 4

Modifications apportées au premier trimestre 2025 au 05/04/2025 :

Mise à jour de STIG pour Internet Explorer 11 STIG Version 2 Release 5 pour tous les composants STIG pour la version du premier trimestre 2025.

- STIG-Build-Windows-Low version 2025.1.x
- STIG-Build-Windows-Medium version 2025.1.x
- STIG-Build-Windows-High version 2025.1.x

Modifications apportées au quatrième trimestre 2024 - 02/04/2025 :

Versions STIG mises à jour et STIG appliqués pour la version du quatrième trimestre 2024 comme suit :

STIG-Build-Windows-Low version 2024.4.0

- Windows Server 2022 STIG version 2 version 2
- Windows Server 2019 STIG Version 3 Release 2
- Windows Server 2016 STIG version 2 version 9
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 utilisant STIG version 2 sortie 2
- Pare-feu Windows STIG version 2, version 2
- Internet Explorer 11 STIG version 2 version 5
- Microsoft Edge STIG Version 2 Release 2 (Windows Server 2022 seulement)

STIG-Build-Windows-Medium version 2024.4.0

- Windows Server 2022 STIG version 2 version 2
- Windows Server 2019 STIG Version 3 Release 2
- Windows Server 2016 STIG version 2 version 9
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 utilisant STIG version 2 sortie 2
- Pare-feu Windows STIG version 2, version 2
- Internet Explorer 11 STIG version 2 version 5
- Microsoft Edge STIG Version 2 Release 2 (Windows Server 2022 seulement)

- Defender STIG Version 2 Release 4

STIG-Build-Windows-High version 2024.4.0

- Windows Server 2022 STIG version 2 version 2
- Windows Server 2019 STIG Version 3 Release 2
- Windows Server 2016 STIG version 2 version 9
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 utilisant STIG version 2 sortie 2
- Pare-feu Windows STIG version 2, version 2
- Internet Explorer 11 STIG version 2 version 5
- Microsoft Edge STIG Version 2 Release 2 (Windows Server 2022 seulement)
- Defender STIG Version 2 Release 4

Changements du troisième trimestre 2024 - 10/04/2023 (aucun changement) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du troisième trimestre 2024.

Modifications apportées au deuxième trimestre 2024 – 05/10/2024 (aucune modification) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du deuxième trimestre 2024.

Changements du premier trimestre 2024 - 02/06/2024 (aucun changement) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du premier trimestre 2024.

Changements du quatrième trimestre 2023 - 12/04/2023 (aucun changement) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du quatrième trimestre 2023.

Modifications apportées au troisième trimestre 2023 – 10/04/2023 (aucune modification) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du troisième trimestre 2023.

Modifications apportées au deuxième trimestre 2023 – 05/03/2023 (aucune modification) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du deuxième trimestre 2023.

Modifications apportées au premier trimestre 2023 – 27/03/2023 (aucune modification) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du premier trimestre 2023.

Modifications apportées au quatrième trimestre 2022 – 01/02/2023 :

Versions STIG mises à jour et STIG appliqués pour la version du quatrième trimestre 2022 comme suit :

STIG-Build-Windows-Low version 2022.4.x

- Windows Server 2022 utilisant STIG version 1 sortie 1
- Windows Server 2019 utilisant STIG version 2 sortie 5
- Windows Server 2016 utilisant STIG version 2 sortie 5
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 utilisant STIG version 2 sortie 2
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 utilisant STIG version 2 sortie 3
- Microsoft Edge STIG Version 1 Release 6 (Windows Server 2022 seulement)

STIG-Build-Windows-Medium version 2022.4.x

- Windows Server 2022 utilisant STIG version 1 sortie 1
- Windows Server 2019 utilisant STIG version 2 sortie 5
- Windows Server 2016 utilisant STIG version 2 sortie 5
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 utilisant STIG version 2 sortie 2
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 utilisant STIG version 2 sortie 3
- Microsoft Edge STIG Version 1 Release 6 (Windows Server 2022 seulement)
- Defender STIG Version 2 Release 4 (Windows Server 2022 seulement)

STIG-Build-Windows-High version 2022.4.x

- Windows Server 2022 utilisant STIG version 1 sortie 1
- Windows Server 2019 utilisant STIG version 2 sortie 5
- Windows Server 2016 utilisant STIG version 2 sortie 5
- Windows Server 2012 R2 MS utilisant STIG version 3 sortie 5
- Microsoft .NET Framework 4.0 utilisant STIG version 2 sortie 2
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 utilisant STIG version 2 sortie 3
- Microsoft Edge STIG Version 1 Release 6 (Windows Server 2022 seulement)
- Defender STIG Version 2 Release 4 (Windows Server 2022 seulement)

Modifications apportées au troisième trimestre 2022 – 30/09/2022 (aucune modification) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du troisième trimestre 2022.

Modifications apportées au deuxième trimestre 2022 – 08/02/2022 :

Versions STIG mises à jour et STIG appliqués pour la version du deuxième trimestre 2022.

STIG-Build-Windows-Low version 1.5.x

- Windows Server 2019 STIG version 2 version 4
- Windows Server 2016 STIG version 2 version 4
- Windows Server 2012 R2 MS STIG version 3 version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 STIG version 1, version 19

STIG-Build-Windows-Medium version 1.5.x

- Windows Server 2019 STIG version 2 version 4
- Windows Server 2016 STIG version 2 version 4
- Windows Server 2012 R2 MS STIG version 3 version 3

- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 STIG version 1, version 19

STIG-Build-Windows-High version 1.5.x

- Windows Server 2019 STIG version 2 version 4
- Windows Server 2016 STIG version 2 version 4
- Windows Server 2012 R2 MS STIG version 3 version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 STIG version 1, version 19

Modifications apportées au premier trimestre 2022 – 08/02/2022 (aucune modification) :

Aucune modification n'a été apportée au composant Windows STIG pour la version du premier trimestre 2022.

Modifications apportées au quatrième trimestre 2021 – 20/12/2021 :

Versions STIG mises à jour et STIG appliqués pour la version du quatrième trimestre 2021 comme suit :

STIG-Build-Windows-Low version 1.5.x

- Windows Server 2019 STIG version 2 version 3
- Windows Server 2016 STIG version 2 version 3
- Windows Server 2012 R2 MS STIG version 3 version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 STIG version 1, version 19

STIG-Build-Windows-Medium version 1.5.x

- Windows Server 2019 STIG version 2 version 3

- Windows Server 2016 STIG version 2 version 3
- Windows Server 2012 R2 MS STIG version 3 version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 STIG version 1, version 19

STIG-Build-Windows-High version 1.5.x

- Windows Server 2019 STIG version 2 version 3
- Windows Server 2016 STIG version 2 version 3
- Windows Server 2012 R2 MS STIG version 3 version 3
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Pare-feu Windows utilisant STIG version 2 sortie 1
- Internet Explorer 11 STIG version 1, version 19

Modifications apportées au troisième trimestre 2021 au 30/09/2021 :

Versions STIG mises à jour et STIG appliqués pour la version du troisième trimestre 2021.

STIG-Build-Windows-Low version 1.4.x

- Windows Server 2019 STIG version 2 version 2
- Windows Server 2016 STIG version 2 version 2
- Windows Server 2012 R2 MS STIG version 3 version 2
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Windows Firewall STIG Version 1 Release 7
- Internet Explorer 11 STIG version 1, version 19

STIG-Build-Windows-Medium version 1.4.x

- Windows Server 2019 STIG version 2 version 2
- Windows Server 2016 STIG version 2 version 2
- Windows Server 2012 R2 MS STIG version 3 version 2
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1

- Windows Firewall STIG Version 1 Release 7
- Internet Explorer 11 STIG version 1, version 19

STIG-Build-Windows-High version 1.4.x

- Windows Server 2019 STIG version 2 version 2
- Windows Server 2016 STIG version 2 version 2
- Windows Server 2012 R2 MS STIG version 3 version 2
- Microsoft .NET Framework 4.0 STIG Version 2 Release 1
- Windows Firewall STIG Version 1 Release 7
- Internet Explorer 11 STIG version 1, version 19

Composants de renforcement Linux STIG

Cette section contient des informations sur les composants de renforcement de Linux STIG, suivies d'un journal de l'historique des versions. Si la distribution Linux ne possède pas ses propres paramètres STIG, le composant de renforcement applique les paramètres RHEL.

Les composants Linux ont des paramètres d'entrée facultatifs qui vous aident à personnaliser les comportements suivants pour votre instance Linux.

- Niveau (chaîne) Si aucune valeur n'est spécifiée, la valeur par défaut est `High` et appliquera toutes les configurations Low, Medium et High applicables.
- InstallPackages (chaîne) Si la valeur est`No`, le composant n'installe aucun package logiciel supplémentaire. Si la valeur est égale à `Yes` ce chiffre, le composant installe des progiciels supplémentaires nécessaires pour une conformité maximale. La valeur par défaut est `No`.
- SetDoDConsentBannière (chaîne) Si la valeur est`No`, la bannière de consentement du DoD ne s'affiche pas lorsque vous vous connectez à une instance sur laquelle l'un des composants STIG Linux est installé. Si la valeur est « oui »`Yes`, la bannière de consentement du DoD s'affiche avant que vous ne vous connectiez lorsque vous vous connectez à une instance sur laquelle l'un des composants STIG Linux est installé. Vous devez confirmer la bannière avant de pouvoir vous connecter. La valeur par défaut est `No`.

  Pour un exemple de bannière de consentement, consultez la Disclaimer Department of Defense Privacy and Consent Notice qui apparaît lorsque vous accédez au site Web de DLA Document Services.

Le composant de renforcement applique les paramètres STIG pris en charge à l'infrastructure basée sur la distribution Linux, comme suit :

Paramètres STIG de Red Hat Enterprise Linux (RHEL) 7

- RHEL 7
- CentOS 7
- Amazon Linux (2AL2)

Réglages RHEL 8 STIG

- RHEL 8
- CentOS 8

Réglages RHEL 9 STIG

- RHEL 9
- CentOS Stream 9

Linux STIG Niveau faible (catégorie III)

La liste suivante contient les paramètres STIG que le composant de renforcement applique à votre infrastructure. Si aucun paramètre pris en charge n'est applicable à votre infrastructure, le composant de renforcement ignore ce paramètre et passe à autre chose. Par exemple, certains paramètres STIG peuvent ne pas s'appliquer aux serveurs autonomes. Les politiques spécifiques à l'organisation peuvent également affecter les paramètres appliqués par le composant de renforcement, par exemple l'obligation pour les administrateurs de revoir les paramètres du document.

Pour une liste complète, consultez la bibliothèque de STIGs documents. Pour plus d'informations sur l'affichage de la liste complète, veuillez consulter la rubrique Outils d'affichage STIG.

RHEL 7 STIG Version 3 Release 15

- RHEL 7/CentOS 7/ AL2

  V-204452, V-204576 et V-204605

RHEL 8 STIG Version 2 Version 6

- RHEL 8/CentOS 8

  V-230241, V-230269, V-230270, V-230281, V-230285, V-230346, V-230381, V-230395, V-230468, V-230469, V-230485, V-230486, V-230491, V-230494, V-230495, V-230496, V-230497, V-230498, V-230499 et V-244527

RHEL 9 STIG version 2, version 7

- RHEL 9/CentOS Stream 9

  V-257782, V-257824, V-258138, V-258037, V-257880, V-258069, V-258076, V-258067, V-257946, V-257947, V-257795, V-257796 et V-258173

Amazon Linux 2023 STIG version 1, version 2

V-274141

SLES 12 STIG version 3, version 4

V-217108, V-217113, V-217140, V-217198, V-217209, V-217211, V-217212, V-217213, V-217214, V-217215, V-217216, V-217236, V-217237, V-217238, V-217239, V-217282 et V-255915

SLES 15 STIG version 2, version 6

V-234811, V-234850, V-234868, V-234873, V-234905, V-234907, V-234908, V-234909, V-234933, V-234934, V-234935, V-234936, V-234955, V-234963, V-234967 et V-255921

Ubuntu 18.04 STIG version 2, version 15

V-219163, V-219164, V-219165, V-219172, V-219173, V-219174, V-219175, V-219178, V-219179, V-219180, V-219210, V-219301, V-219327, V-219332 et V-219333

Ubuntu 20.04 STIG version 2 version 4

V-238202, V-238203, V-238221, V-238222, V-238223, V-238224, V-238226, V-238234, V-238235, V-238237, V-238308, V-238323, V-238357, V-238362 et V-238373

Ubuntu 22.04 STIG version 2, version 7

V-260472, V-260476, V-260479, V-260480, V-260481, V-260520, V-260521, V-260549, V-260550, V-260551, V-260552, V-260581 et V-260596

Ubuntu 24.04 STIG version 1 version 4

V-270645, V-270646, V-270664, V-270677, V-270690, V-270695, V-270706, V-270710, V-270734, V-270749, V-270752, V-270818 et V-270820

Linux STIG Niveau moyen (catégorie II)

La liste suivante contient les paramètres STIG que le composant de renforcement applique à votre infrastructure. Si aucun paramètre pris en charge n'est applicable à votre infrastructure, le composant de renforcement ignore ce paramètre et passe à autre chose. Par exemple, certains paramètres STIG peuvent ne pas s'appliquer aux serveurs autonomes. Les politiques spécifiques à l'organisation peuvent également affecter les paramètres appliqués par le composant de renforcement, par exemple l'obligation pour les administrateurs de revoir les paramètres du document.

Pour une liste complète, consultez la bibliothèque de STIGs documents. Pour plus d'informations sur l'affichage de la liste complète, veuillez consulter la rubrique Outils d'affichage STIG.

> **Note**
>
> Les composants de renforcement moyen de Linux STIG incluent tous les paramètres STIG répertoriés qui AWSTOE s'appliquent aux composants de renforcement faible de Linux STIG, en plus des paramètres STIG répertoriés spécifiquement pour les vulnérabilités de catégorie II.

RHEL 7 STIG Version 3 Release 15

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

- RHEL 7/CentOS 7/ AL2

  V-204405, V-204406, V-204407, V-204408, V-204409, V-204410, V-204411, V-204412, V-204413, V-204414, V-204415, V-204416, V-204417, V-204418, V-204420, V-204423, V-204426, V-204427, V-204431, V-204434, V-204435, V-204437, V-204449, V-204450, V-204451, V-204457, V-204466, V-204490, V-204491, V-204503, V-204507, V-204508, V-204510, V-204511, V-204512, V-204514, V-204515, V-204516, V-204517, V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V-204541, V-204542, V-204543, V-204545, V-204546, V-204547, V-204548, V-204549, V-204550, V-204551, V-204552, V-204553, V-204554, V-204555, V-204556, V-204558, V-204559, V-204560, V-204562, V-204563, V-204564, V-204565, V-204566, V-204567, V-204568,

V-204572, V-204579, V-204584, V-204585, V-204587, V-204588, V-204589, V-204590, V-204591, V-204592, V-204593, V-204596, V-204597, V-204598, V-204599, V-204600, V-204601, V-204602, V-204609, V-204610, V-204611, V-204612, V-204613, V-204614, V-204615, V-204616, V-204617, V-204622, V-204625, V-204630, V-204631, V-204633, V-233307, V-237634, V-237635, V-251703, V-255925, V-2559, V-2559, V-2559, V-2559, V-2559, V-255928 et V-256970

RHEL 8 STIG Version 2 Version 6

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

- RHEL 8/CentOS 8

V-230222, V-230228, V-230231, V-230233, V-230236, V-230237, V-230238, V-230239, V-230240, V-230243, V-230244, V-230245, V-230246, V-230247, V-230248, V-230249, V-230250, V-230255, V-230256, V-230257, V-230258, V-230259, V-230260, V-230261, V-230262, V-230266, V-230267, V-230268, V-230271, V-230273, V-230275, V-230276, V-230277, V-230278, V-230279, V-230280, V-230282, V-230282, V-230286, V-230287, V-230288, V-230290, V-230291, V-230296, V-230298, V-230310, V-230310, V-230290 311, V230312, V230313, V230314, V230315, V230316, V230318, V230319, V230320, V230321, V-230322, V-230324, V-230325, V-230326, V-230327, V-230330, V-230332, V-230333, V-230335, V-230337, V-230339, V-230341, V-230343, V-230345, V-230347, V-230348, V-230352, V-230353, V-230354, V-230356, V-230357, V-230358, V-230359, V-230359, V-230359, V-230354 360, V-230361, V-230362, V-230363, V-230365, V-230366, V-230368, V-230369, V-230370, V-230372, V-230373, V-230375, V-230376, V-230377, V-230378, V-230380, V-230382, V-230383, V-230385, V-230386, V-230387, V-230389, V-230390, V-230392, V-230392 230393, V230394, V230396, V230397, V230398, V230399, V230400, V230401, V230402, V230403, V-230404, V-230405, V-230406, V-230407, V-230408, V-230409, V-230410, V-230411, V-230412, V-230413, V-230418, V-230419, V-230421, V-230422, V-230423, V-230424, V-230425, V-230426, V-230427, V-230428, V-230429, V-230430, V-230431, V-230432, V-230433, V-230434, V-230435, V-230436, V-230437, V-230438, V-230439, V-230444, V-230446, V-230447, V-230448, V-230449, V-230455, V-230456, V-230462, V-230464, V-230465, V-230466, V-230467, V230470, V230471, V230472, V230473, V230474, V230475, V230478, V230480, V230481, V230482, V230483, V230488, V230489, V-230493, V-230502, V-230503, V-230505, V-230506, V-230507, V-230523, V-230524, V-230525, V-230526, V-230527, V-230532, V-230535, V-230536, V-230537, V-230538, V-230539, V-230540, V-230541, V-230542, V-230543, V-230544, V-230545, V-230546, V-230548, V-230549, V-230550, V-230555, V-230556, V-230557, V-230559, V-230560, V-230561, V-237640, V-237642, V-237643, V-244519, V-244523, V-242424, V-4525, V-244526, V-244528,

V-244531, V-244533, V-244535, V-244536, V-244538, V-244539, V-244542, V-244543, V-244544, V-244545, V-244547, V-244550, V-244551, V-244552, V-244553, V-244554, V-250315, V-250315, V-250316, V-250316, V-250317, V-251707, V-251708, V-251709, V-251710, V-251711, V-251713, V-251714, V-251715, V-251716, V-251717, V-251718, V-256974, V-257258, V-257258, V-274877, V-279929, V-279930 et V-279931

RHEL 9 STIG version 2, version 7

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

- RHEL 9/CentOS Stream 9

V-257780, V-257781, V-257783, V-257786, V-257788, V-257790, V-257791, V-257792, V-257793, V-257794, V-257797, V-257798, V-257799, V-257800, V-257801, V-257802, V-257803, V-257804, V-2577804 805, V-257806, V-257807, V-257808, V-257809, V-257810, V-257811, V-257812, V-257813, V-257814, V-257815, V-257816, V-257817, V-257825, V-257827, V-257828, V-257829, V-257830, V-257831, V-257832, V-257833, V-257834, V-257836, V-257838, V-257839, V-257840, V-257841, V-257842, V-257849, V-257882, V-257883, V-257884, V-257885, V-257886, V-257887, V-25788, V-257889, V-257890, V-257891, V-257892, V-257893, V-257894, V-257895, V-257896, V-257897, V-257899, V-257900, V-257901, V-257902, V-257903, V-257904, V-257905, V-257906, V-257907, V-257908, V-257909, V-257910, V-257911, V-257912, V-257913, V-257914, V-257915, V-257916, V-257917, V-257918, V-257919, V-257920, V-257921, V-257922, V-257923, V-257924, V-257925, V-257926, V-257927, V-257928, V-257929, V-257930, V-257933, V-257934, V-257935, V-257936, V-257939, V-257940, V-257942, V-257943, V-257944, V-257948, V-257951, V-257952, V-257953, V-257954, V-257957, V-257958, V-257959, V-257960, V-257961, V-257962, V-257963, V-257964, V-257965, V-257966, V-257968, V-257969, V-257970, V-257971, V-257972, V-257973, V-257974, V-257975, V-257976, V-257977, V-257978, V-257979, V-257980, V-257982, V-257983, V-257985, V-257987, V-257988, V-257992, V-257993, V-257994, V-257995, V-257996, V-257997, V-257999, V-258000, V-258001, V-258002, V-258002, V-258003, V-258004, V-258005, V-258006, V-258007, V-258008, V-258009, V-258011, V-258028, V-258034, V-258035, V-258038, V-258039, V-258041, V-258043, V-258046, V-258049, V-258049, V-258052, V-258054, V-258055, V-258056, V-258057, V-258060, V-258063, V-258064, V-258066, V-258068, V-258070, V-258071, V-258072, V-258073, V-258074, V-258077, V-258077, V-258079, V-258080, V-258081, V-258082, V-258083, V-258084, V-258085, V-258088, V-258089, V-258090, V-258091, V-258092, V-258093, V-258095, V-258097, V-258098, V-258100, V-258101, V-258102, V-258103, V-258104, V-258105, V-258107, V-258108, V-258109, V-258110, V-258111, V-258112, V-258113, V-258114, V-258115, V-258116,

V-258117, V-258118, V-258119, V-258121, V-258122, V-258123, V-258124, V-258125, V-258126, V-258128, V-258129, V-258130, V-258133, V-258137, V-258140, V-258141, V-258142, V-258144, V-258145, V-258147, V-258148, V-258150, V-258151, V-258152, V-258153, V-258154, V-258156, V-258157, V-258158, V-258159, V-258160, V-258161, V-258162, V-258163, V-258164, V-258165, V-258166, V-258168, V-258169, V-258170, V-258171, V-258172, V-258175, V-258176, V-258177, V-258178, V-258179, V-258180, V-258181, V-258182, V-258183, V-258184, V-258185, V-258186, V-258187, V-258188, V-258190, V-258191, V-258192, V-2583, V-258194, V-258195, V-258196, V-258197, V-258198, V-258199, V-258200, V-258201, V-258202, V-258203, V-258204, V-258205, V-258206, V-258207, V-258208, V-258210, V-258211, V-258212, V-258213, V-258214, V-258215, V-258216, V-258217, V-258218, V-258219, V-258220, V-258221, V-258222, V-258223, V-258224, V-258225, V-258227, V-258228, V-258229, V-258232, V-258233, V-258234, V-258237, V-258239, V-258240, V-270174, V-270175, V-270176, V-270177, V-272488 et V-279936

Amazon Linux 2023 STIG version 1, version 2

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

V-273995, V-274000, V-274001, V-274002, V-274003, V-274004, V-274005, V-274006, V-274008, V-274009, V-274010, V-274011, V-274012, V-274013, V-274014, V-274017, V-274018, V-274019, V-274020, V-274020, V-274021, V-274022, V-274023, V-274024, V-274026, V-274027, V-274028, V-274030, V-274031, V-274032, V-274033, V-274034, V-274035, V-274037, V-274040, V-274041, V-274042, V-274044, V-274044, V-274045, V-274047, V-274048, V-274049, V-274050, V-274051, V-274053, V-274054, V-274059, V-274061, V-274062, V-274069, V-274070, V-274071, V-274072, V-274073, V-274074, V-274075, V-274076, V-274077, V-274078, V-274079, V-274081, V-274082, V-274083, V-274084, V-274085, V-274086, V-274087, V-274088, V-274089, V-274090, V-274091, V-274092, V-274093, V-274094, V-274095, V-274096, V-274097, V-274098, V-274099, V-274100, V-274101, V-274102, V-274103, V-274104, V-274105, V-274106, V-274107, V-274108, V-274109, V-274111, V-274112, V-274113, V-274114, V-274115, V-274116, V-274117, V-274119, V-274120, V-274121, V-274122, V-274123, V-274124, V-274125, V-274126, V-274127, V-274128, V-274129, V-274130, V-274131, V-274132, V-274133, V-274134, V-274135, V-274136, V-274137, V-274138, V-274139, V-274140, V-274142, V-274143, V-274144, V-274145, V-274147, V-274149, V-274151, V-274152, V-274154, V-274155, V-274156, V-274157, V-274160, V-274161, V-274162, V-274163, V-274164, V-274165, V-274166, V-274167, V-274168, V-274169, V-274170, V-274173, V-274177, V-274181, V-274182, V-274185 et V-274187

SLES 12 STIG version 3, version 4

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

V-217102, V-217105, V-217105, V-217106, V-217106, V-217110, V-217116, V-217117, V-217118, V-217119, V-217120, V-217121, V-217122, V-217124, V-217125, V-217127, V-217128, V-217130, V-217134, V-217138, V-217143, V-217147, V-217152, V-217153, V-217154, V-217155, V-217156, V-217158, V-217161, V-217163, V-217166, V-217168, V-217169, V-217170, V-217171, V-217182, V-217183, V-217188, V-217190, V-217191, V-217194, V-217195, V-217196, V-217197, V-217200, V-217201, V-217202, V-217203, V-217204, V-217205, V-217206, V-217208, V-217210, V-217217221, V-217218, V-217223, V-217227, V-217230, V-217240, V-217241, V-217242, V-217243, V-217244, V-217245, V-217246, V-217247, V-217248, V-217249, V-217250, V-217252, V-217253, V-217254, V-217255, V-217257, V-217258, V-217260, V-217265, V-217266, V-217267, V-217269, V-217272, V-217273, V-217274, V-217275, V-217276, V-217277, V-217278, V-217280, V-217283, V-217284, V-217286, V-217287, V-217288, V-217289, V-217290, V-217291, V-217292, V-217293, V-217294, V-217295, V-217296, V-217299, V-217300, V-217301, V-217302, V-233308, V-237605, V-237606, V-237607767, V-237608, V-237609, V-237610, V-237611, V-237612, V-237613, V-237614, V-237615, V-237616, V-237617, V-237618, V-237619, V-237620, V-237621, V-237622, V-251720, V-251722, V-251722, V-255914, et V-256981

SLES 15 STIG version 2, version 6

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

V-234802, V-234807, V-234808, V-234809, V-234813, V-234815, V-234817, V-234821, V-234822, V-234823, V-234825, V-234827, V-234828, V-234829, V-234830, V-234832, V-234833, V-234834, V-234835, V-234836, V-234837, V-234838, V-234839, V-234840, V-234841, V-234842, V-234843, V-234844, V-234845, V-234848, V-234854, V-234855, V-234857, V-234858, V-234861, V-234862, V-234863, V-234869, V-234870, V-234875, V-234878, V-234880, V-234881, V-234882, V-234883, V-234884, V-234885, V-234886, V-234887, V-234888, V-234889, V-234891, V-234896, V-234897, V-234899, V-234900, V-234901, V-234902, V-234903, V-234904, V-234906, V-234910, V-234911, V-234912, V-234913, V-234914, V-234918, V-234924, V-234928, V-234932, V-234937, V-234938, V-234939, V-234940, V-234941, V-234942, V-234943, V-234944, V-234945, V-234946, V-234947, V-234948, V-234949, V-234950, V-234951, V-234952, V-234954, V-234956, V-234957, V-234958, V-234959, V-234961, V-234962, V-234964, V-234966, V-234969, V-234973, V-234975, V-234976, V-234977, V-234978, V-234979, V-234981, V-234982, V-234983, V-234991, V-235002, V-235003, V-235007, V-235008, V-23105009, V-23509, V-235013, V-235014, V-235015, V-235016, V-235017,

V-235018, V-235019, V-235020, V-235021, V-235022, V-235023, V-235024, V-235025, V-235026, V-235028, V-235029, V-235030, V-251724, V-255920, V-256983 et V-274879

Ubuntu 18.04 STIG version 2, version 15

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

V-219149, V-219155, V-219156, V-219160, V-219166, V-219168, V-219176, V-219181, V-219184, V-219186, V-219188, V-219189, V-219190, V-219191, V-219192, V-219193, V-219194, V-219195, V-219196, V-219197, V-219198, V-219199, V-219200, V-219201, V-219202, V-219203, V-219204, V-219205, V-219206, V-219207, V-219208, V-219209, V-219213, V-219214, V-219215, V-219216, V-219217, V-219218, V-219219, V-219220, V-219221, V-219222, V-219223, V-219224, V-219225, V-219226, V-219227, V-219228, V-219229, V-219230, V-219231, V-219232, V-219233, V-219234, V-219235, V-219236, V-219238, V-219239, V-219240, V-219241, V-219242, V-219243, V-219244, V-219250, V-219254, V-219257, V-219263, V-219264, V-219265, V-219266, V-219267, V-219268, V-219269, V-219270, V-219271, V-219272, V-219273, V-219274, V-219275, V-219276, V-219277, V-219279, V-219281, V-219287, V-219291, V-219296, V-219297, V-219298, V-219299, V-219300, V-219303, V-219304, V-219306, V-219309, V-219310, V-219311, V-219315, V-219318, V-219319, V-219323, V-219326, V-219328, V-219330, V-219331, V-219335, V-219336, V-219337, V-219338, V-219339, V-219342, V-219344, V-233779, V-233780 et V-255906

Ubuntu 20.04 STIG version 2 version 4

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

V-238200, V-238205, V-238207, V-238209, V-238210, V-238211, V-238212, V-238213, V-238220, V-238225, V-238227, V-238228, V-238229, V-238230, V-238231, V-238232, V-238236, V-238238, V-238239, V-238240, V-238241, V-238242, V-238244, V-238245, V-238246, V-238247, V-238248, V-238249, V-238250, V-238251, V-238252, V-238253, V-238254, V-238255, V-238256, V-238257, V-238258, V-238264, V-238268, V-238271, V-238277, V-238278, V-238279, V-238280, V-238281, V-238282, V-238283, V-238284, V-238285, V-238286, V-238287, V-238288, V-238289, V-238290, V-238291, V-238292, V-238293, V-238294, V-238295, V-238297, V-238298, V-238299, V-238300, V-238301, V-238302, V-238303, V-238304, V-238309, V-238310, V-238315, V-238316, V-238317, V-238318, V-238319, V-238320, V-238324, V-238325, V-238329, V-238330, V-238333, V-238334, V-238337, V-238338, V-238339, V-238340, V-238341, V-238342, V-238343, V-238344, V-238345, V-238346, V-238347, V-238348, V-238349, V-238350, V-238351, V-238352, V-238353, V-238355,

V-238356, V-238359, V-238360, V-238369, V-238370, V-238371, V-238376, V-238377, V-238378, V-251505, V-255912, V-274852 et V-274853

Ubuntu 22.04 STIG version 2, version 7

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

V-260471, V-260473, V-260474, V-260475, V-260477, V-260478, V-260485, V-260486, V-260487, V-260488, V-260489, V-260490, V-260491, V-260492, V-260493, V-260494, V-260495, V-260496, V-260497, V-260498, V-260499, V-260500, V-260505, V-260506, V-260507, V-260508, V-260509, V-260533, V-260534, V-260535, V-260537, V-260538, V-260540, V-260542, V-260542, V-260534, V-260535, V-260537, V-260538, V-260540, V-260542, V-260542, V-260542 543, V-260545, V-260546, V-260547, V-260553, V-260554, V-260555, V-260556, V-260557, V-260560, V-260561, V-260562, V-260563, V-260564, V-260565, V-260566, V-260567, V-260569, V-260572, V-260573, V-260574, V-260575, V-260576, V-260582, V-260584, V-260585, V-260586, V-260588, V-260589, V-260591, V-260594, V-260594, V-260596 7, V-260598, V-260599, V-260600, V-260601, V-260602, V-260603, V-260604, V-260605, V-260606, V-260607, V-260608, V-260609, V-260610, V-260611, V-260612, V-260613, V-260614, V-260616, V-260616, V-260617, V-260618, V-260619, V-260620, V-260621, V-260622, V-260623, V-260624, V-260625, V-260626, V-260627, V-260628, V-260629, V-260630, V-260631, V-260632, V-260633, V-260634, V-260635, V-260636, V-260637, V-260638, V-260639, V-260640, V-260641, V-260642, V-260643, V-260644, V-260645, V-260646, V-260647, V-260648, V-260649, V-274862, V-274864 et V-274866

Ubuntu 24.04 STIG version 1 version 4

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique aux vulnérabilités de catégorie III (faible) pour cette distribution Linux, ainsi que :

V-270649, V-270651, V-270652, V-270653, V-270654, V-270656, V-270657, V-270659, V-270660, V-270661, V-270662, V-270663, V-270669, V-270672, V-270673, V-270674, V-270676, V-270678, V-270679, V-270680, V-270681, V-270683, V-270684, V-270685, V-270686, V-270687, V-270688, V-270689, V-270692, V-270693, V-270696, V-270697, V-270698, V-270699, V-270700, V-270701, V-270702, V-270703, V-270704, V-270705, V-270709, V-270715, V-270716, V-270718, V-270720, V-270721, V-270722, V-270723, V-270724, V-270725, V-270726, V-270727, V-270728, V-270729, V-270730, V-270731, V-270732, V-270733, V-270737, V-270739, V-270740, V-270741, V-270742, V-270743, V-270746, V-270750, V-270753, V-270755, V-270756, V-270757, V-270758, V-270759, V-270760, V-270765, V-270766, V-270767, V-270768, V-270769, V-270770, V-270771, V-270772, V-270773, V-270775, V-270776, V-270777, V-270778, V-270779, V-270780, V-270781, V-270782,

V-270783, V-270784, V-270785, V-270786, V-270787, V-270788, V-270789, V-270790, V-270791, V-270792, V-270793, V-270794, V-270795, V-270796, V-270797, V-270798, V-270799, V-270800, V-270801, V-270802, V-270803, V-270804, V-270805, V-270806, V-270807, V-270808, V-270809, V-270810, V-270811, V-270812, V-270813, V-270814, V-270815, V-270821, V-270822, V-270823, V-270824, V-270825, V-270826, V-270827, V-270828, V-270829, V-270830, V-270831, V-270832, V-274870, V-274871, V-274872 et V-274873

Linux STIG Niveau élevé (catégorie I)

La liste suivante contient les paramètres STIG que le composant de renforcement applique à votre infrastructure. Si aucun paramètre pris en charge n'est applicable à votre infrastructure, le composant de renforcement ignore ce paramètre et passe à autre chose. Par exemple, certains paramètres STIG peuvent ne pas s'appliquer aux serveurs autonomes. Les politiques spécifiques à l'organisation peuvent également affecter les paramètres appliqués par le composant de renforcement, par exemple l'obligation pour les administrateurs de revoir les paramètres du document.

Pour une liste complète, consultez la bibliothèque de STIGs documents. Pour plus d'informations sur l'affichage de la liste complète, veuillez consulter la rubrique Outils d'affichage STIG.

> **Note**
>
> Les composants de renforcement élevé de Linux STIG incluent tous les paramètres STIG répertoriés qui AWSTOE s'appliquent aux composants de renforcement Linux STIG Low et Linux STIG Medium, en plus des paramètres STIG répertoriés qui s'appliquent spécifiquement aux vulnérabilités de catégorie I.

RHEL 7 STIG Version 3 Release 15

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

- RHEL 7/CentOS 7/ AL2

  V-204424, V-204425, V-204442, V-204443, V-204447, V-204448, V-204455, V-204462, V-204497, V-204497, V-204502, V-204594, V-204620 et V-204621

RHEL 8 STIG Version 2 Version 6

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

- RHEL 8/CentOS 8

  V-230223, V-230264, V-230283, V-230284, V-230487, V-230492, V-230533, V-230558, V-244540, V-279933, V-230265, V-230226, V-230530, V-268322, V-230529 et V-230531

RHEL 9 STIG version 2, version 7

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

- RHEL 9/CentOS Stream 9

  V-257820, V-257821, V-257826, V-257835, V-257955, V-257956, V-258059, V-258230, V-258238, V-257984, V-257986, V-258078, V-258094, V-258235, V-257784 et V-257785

Amazon Linux 2023 STIG version 1, version 2

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

V-273996, V-273997, V-273999, V-274007, V-274038, V-274039, V-274046, V-274052, V-274057 et V-274153

SLES 12 STIG version 3, version 4

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

V-217101, V-217139, V-217141, V-217142, V-217159, V-217160, V-217164, V-217264, V-217268, V-222386 et V-251721

SLES 15 STIG version 2, version 6

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

V-234800, V-234804, V-234818, V-234852, V-234859, V-234860, V-234898, V-234984, V-234985, V-234988, V-234989, V-234990, V-235031, V-235032 et V-251725

Ubuntu 18.04 STIG version 2, version 15

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

V-219157, V-219158, V-219177, V-219212, V-219308, V-219314, V-219316 et V-251507

Ubuntu 20.04 STIG version 2 version 4

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

V-238201, V-238218, V-238219, V-238326, V-238327, V-238380 et V-251504

Ubuntu 22.04 STIG version 2, version 7

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

V-260469, V-260482, V-260483, V-260523, V-260524, V-260526, V-260529, V-260539, V-260570, V-260571, V-260579 et V-279937

Ubuntu 24.04 STIG version 1 version 4

Inclut tous les paramètres STIG pris en charge que le composant de renforcement applique pour les vulnérabilités de catégories II et III (moyennes et faibles) de cette distribution Linux, ainsi que :

V-270647, V-270648, V-270665, V-270666, V-270708, V-270711, V-270712, V-270713, V-270714, V-270717, V-270736, V-270738 et V-279938

Journal d'historique des versions de STIG pour Linux

Cette section enregistre l'historique des versions des composants Linux. Pour voir les modifications et les versions publiées pendant un trimestre, choisissez le titre pour développer les informations.

Modifications apportées au premier trimestre 2026 - 10/03/2026 :

Mise à jour des versions STIG suivantes, appliquées STIGs à la version du premier trimestre 2026 pour tous les niveaux de conformité :

STIG-Build-Linux

- RHEL 8 STIG Version 2 Version 6

- RHEL 9 STIG version 2, version 7
- Amazon Linux 2023 STIG version 1, version 2
- SLES 12 STIG version 3, version 4
- SLES 15 STIG version 2, version 6
- Ubuntu 20.04 STIG version 2 version 4
- Ubuntu 22.04 STIG version 2 version 7
- Ubuntu 24.04 STIG version 1 version 4

Modifications apportées au troisième trimestre 2025 – 09/04/2025 :

Ajout de la prise en charge du système d'exploitation SUSE Linux Enterprise Server (SLES) et Amazon Linux 2023. Mise à jour des versions STIG suivantes et application des STIGS pour la version du troisième trimestre 2025 pour tous les niveaux de conformité () low/medium/high :

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 4
- RHEL 9 STIG Version 2 Release 5
- Amazon Linux 2023 STIG Version 1 Release 1
- SLES 12 STIG Version 3 Release 3
- SLES 15 STIG Version 2 Release 5
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 3
- Ubuntu 22.04 STIG Version 2 Release 5
- Ubuntu 24.04 STIG Version 1 Release 2

Modifications apportées au deuxième trimestre 2025 – 26/06/2025 :

Mise à jour des versions STIG suivantes, application des STIG pour la version du deuxième trimestre 2025 :

STIG-Build-Linux-Low version 2025.2.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 3

- RHEL 9 STIG Version 2 Release 4
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 2
- Ubuntu 22.04 STIG Version 2 Release 4
- Ubuntu 24.04 STIG Version 1 Release 1

STIG-Build-Linux-Medium version 2025.2.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 3
- RHEL 9 STIG Version 2 Release 4
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 2
- Ubuntu 22.04 STIG Version 2 Release 4
- Ubuntu 24.04 STIG Version 1 Release 1

STIG-Build-Linux-High version 2025.2.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 3
- RHEL 9 STIG Version 2 Release 4
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 2
- Ubuntu 22.04 STIG Version 2 Release 4
- Ubuntu 24.04 STIG Version 1 Release 1

Modifications apportées au premier trimestre 2025 – 04/11/2025 :

Mise à jour des versions STIG suivantes, application de STIG pour la version du premier trimestre 2025 et ajout du support pour Ubuntu 24.04 :

STIG-Build-Linux-Low version 2025.1.x

- RHEL 7 STIG Version 3 Release 15

- RHEL 8 STIG Version 2 Release 2
- RHEL 9 STIG Version 2 Release 3
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 2
- Ubuntu 22.04 STIG Version 2 Release 3
- Ubuntu 24.04 STIG Version 1 Release 1

STIG-Build-Linux-Medium version 2025.1.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 2
- RHEL 9 STIG Version 2 Release 3
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 2
- Ubuntu 22.04 STIG Version 2 Release 3
- Ubuntu 24.04 STIG Version 1 Release 1

STIG-Build-Linux-High version 2025.1.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 2
- RHEL 9 STIG Version 2 Release 3
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 2
- Ubuntu 22.04 STIG Version 2 Release 3
- Ubuntu 24.04 STIG Version 1 Release 1

Modifications apportées au quatrième trimestre 2024 – 10/12/2024 :

Mise à jour des versions STIG suivantes, application de STIG pour la version du quatrième trimestre 2024 et ajout d'informations sur deux nouveaux paramètres d'entrée pour les composants Linux :

STIG-Build-Linux-Low version 2024.4.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 1
- RHEL 9 STIG Version 2 Release 2
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 1
- Ubuntu 22.04 STIG Version 2 Release 2

STIG-Build-Linux-Medium version 2024.4.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 1
- RHEL 9 STIG Version 2 Release 2
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 1
- Ubuntu 22.04 STIG Version 2 Release 2

STIG-Build-Linux-High version 2024.4.x

- RHEL 7 STIG Version 3 Release 15
- RHEL 8 STIG Version 2 Release 1
- RHEL 9 STIG Version 2 Release 2
- Ubuntu 18.04 STIG Version 2 Release 15
- Ubuntu 20.04 STIG Version 2 Release 1
- Ubuntu 22.04 STIG Version 2 Release 2

Modifications apportées au troisième trimestre 2024 – 10/04/2024 (aucune modification) :

Aucune modification n'a été apportée au composant Linux STIG pour la version du troisième trimestre 2024.

Modifications apportées au deuxième trimestre 2024 – 10/05/2024 :

Versions STIG mises à jour et STIG appliqués pour la version du deuxième trimestre 2024. La prise en charge de RHEL 9, CentOS Stream 9 et Ubuntu 22.04 a également été ajoutée, comme suit :

STIG-Build-Linux-Low version 2024.2.x

- RHEL 7 STIG Version 3 Release 14
- RHEL 8 STIG Version 1 Release 14
- RHEL 9 STIG Version 1 Release 3
- Ubuntu 18.04 STIG Version 2 Release 14
- Ubuntu 20.04 STIG Version 1 Release 12
- Ubuntu 22.04 STIG Version 1 Release 1

STIG-Build-Linux-Medium version 2024.2.x

- RHEL 7 STIG Version 3 Release 14
- RHEL 8 STIG Version 1 Release 14
- RHEL 9 STIG Version 1 Release 3
- Ubuntu 18.04 STIG Version 2 Release 14
- Ubuntu 20.04 STIG Version 1 Release 12
- Ubuntu 22.04 STIG Version 1 Release 1

STIG-Build-Linux-High version 2024.2.x

- RHEL 7 STIG Version 3 Release 14
- RHEL 8 STIG Version 1 Release 14
- RHEL 9 STIG Version 1 Release 3
- Ubuntu 18.04 STIG Version 2 Release 14
- Ubuntu 20.04 STIG Version 1 Release 12
- Ubuntu 22.04 STIG Version 1 Release 1

Modifications apportées au premier trimestre 2024 – 02/06/2024 :

Versions STIG mises à jour et STIG appliqués pour la version du premier trimestre 2024 comme suit :

STIG-Build-Linux-Low version 2024.1.x

- RHEL 7 STIG Version 3 Release 14
- RHEL 8 STIG Version 1 Release 13
- Ubuntu 18.04 STIG Version 2 Release 13
- Ubuntu 20.04 STIG Version 1 Release 11

STIG-Build-Linux-Medium version 2024.1.x

- RHEL 7 STIG Version 3 Release 14
- RHEL 8 STIG Version 1 Release 13
- Ubuntu 18.04 STIG Version 2 Release 13
- Ubuntu 20.04 STIG Version 1 Release 11

STIG-Build-Linux-High version 2024.1.x

- RHEL 7 STIG Version 3 Release 14
- RHEL 8 STIG Version 1 Release 13
- Ubuntu 18.04 STIG Version 2 Release 13
- Ubuntu 20.04 STIG Version 1 Release 11

Modifications apportées au quatrième trimestre 2023 – 07/12/2023 :

Versions STIG mises à jour et STIG appliqués pour la version du quatrième trimestre 2023 comme suit :

STIG-Build-Linux-Low version 2023.4.x

- RHEL 7 STIG Version 3 Release 13
- RHEL 8 STIG Version 1 Release 12
- Ubuntu 18.04 STIG Version 2 Release 12
- Ubuntu 20.04 STIG Version 1 Release 10

STIG-Build-Linux-Medium version 2023.4.x

- RHEL 7 STIG Version 3 Release 13
- RHEL 8 STIG Version 1 Release 12
- Ubuntu 18.04 STIG Version 2 Release 12
- Ubuntu 20.04 STIG Version 1 Release 10

STIG-Build-Linux-High version 2023.4.x

- RHEL 7 STIG Version 3 Release 13
- RHEL 8 STIG Version 1 Release 12
- Ubuntu 18.04 STIG Version 2 Release 12
- Ubuntu 20.04 STIG Version 1 Release 10

Modifications apportées au troisième trimestre 2023 – 10/04/2023 :

Versions STIG mises à jour et STIG appliqués pour la version du troisième trimestre 2023 comme suit :

STIG-Build-Linux-Low version 2023.3.x

- RHEL 7 STIG Version 3 Release 12
- RHEL 8 STIG Version 1 Release 11
- Ubuntu 18.04 STIG Version 2 Release 11
- Ubuntu 20.04 STIG Version 1 Release 9

STIG-Build-Linux-Medium version 2023.3.x

- RHEL 7 STIG Version 3 Release 12
- RHEL 8 STIG Version 1 Release 11
- Ubuntu 18.04 STIG Version 2 Release 11
- Ubuntu 20.04 STIG Version 1 Release 9

STIG-Build-Linux-High version 2023.3.x

- RHEL 7 STIG Version 3 Release 12
- RHEL 8 STIG Version 1 Release 11
- Ubuntu 18.04 STIG Version 2 Release 11
- Ubuntu 20.04 STIG Version 1 Release 9

Modifications apportées au deuxième trimestre 2023 – 05/03/2023 :

Versions STIG mises à jour et STIG appliqués pour la version du deuxième trimestre 2023 comme suit :

STIG-Build-Linux-Low version 2023.2.x

- RHEL 7 STIG Version 3 Release 11
- RHEL 8 STIG Version 1 Release 10
- Ubuntu 18.04 STIG Version 2 Release 11
- Ubuntu 20.04 STIG Version 1 Release 8

STIG-Build-Linux-Medium version 2023.2.x

- RHEL 7 STIG Version 3 Release 11
- RHEL 8 STIG Version 1 Release 10
- Ubuntu 18.04 STIG Version 2 Release 11
- Ubuntu 20.04 STIG Version 1 Release 8

STIG-Build-Linux-High version 2023.2.x

- RHEL 7 STIG Version 3 Release 11
- RHEL 8 STIG Version 1 Release 10
- Ubuntu 18.04 STIG Version 2 Release 11
- Ubuntu 20.04 STIG Version 1 Release 8

Modifications apportées au premier trimestre 2023 – 27/03/2023 :

Versions STIG mises à jour et STIG appliqués pour la version du premier trimestre 2023 comme suit :

STIG-Build-Linux-Low version 2023.1.x

- RHEL 7 STIG Version 3 Release 10
- RHEL 8 STIG Version 1 Release 9
- Ubuntu 18.04 STIG Version 2 Release 10
- Ubuntu 20.04 STIG Version 1 Release 7

STIG-Build-Linux-Medium version 2023.1.x

- RHEL 7 STIG Version 3 Release 10
- RHEL 8 STIG Version 1 Release 9
- Ubuntu 18.04 STIG Version 2 Release 10
- Ubuntu 20.04 STIG Version 1 Release 7

STIG-Build-Linux-High version 2023.1.x

- RHEL 7 STIG Version 3 Release 10
- RHEL 8 STIG Version 1 Release 9
- Ubuntu 18.04 STIG Version 2 Release 10
- Ubuntu 20.04 STIG Version 1 Release 7

Modifications apportées au quatrième trimestre 2022 – 01/02/2023 :

Versions STIG mises à jour et STIG appliqués pour la version du quatrième trimestre 2022 comme suit :

STIG-Build-Linux-Low version 2022.4.x

- RHEL 7 STIG Version 3 Release 9
- RHEL 8 STIG Version 1 Release 8
- Ubuntu 18.04 STIG Version 2 Release 9

- Ubuntu 20.04 STIG Version 1 Release 6

STIG-Build-Linux-Medium version 2022.4.x

- RHEL 7 STIG Version 3 Release 9
- RHEL 8 STIG Version 1 Release 8
- Ubuntu 18.04 STIG Version 2 Release 9
- Ubuntu 20.04 STIG Version 1 Release 6

STIG-Build-Linux-High version 2022.4.x

- RHEL 7 STIG Version 3 Release 9
- RHEL 8 STIG Version 1 Release 8
- Ubuntu 18.04 STIG Version 2 Release 9
- Ubuntu 20.04 STIG Version 1 Release 6

Modifications apportées au troisième trimestre 2022 – 30/09/2022 (aucune modification) :

Aucune modification n'a été apportée au composant Linux STIG pour la version du troisième trimestre 2022.

Modifications apportées au deuxième trimestre 2022 – 08/02/2022 :

Introduction du support Ubuntu, mise à jour des versions STIG et application des STIG pour la version du deuxième trimestre 2022 comme suit :

STIG-Build-Linux-Low version 2022.2.x

- RHEL 7 STIG Version 3 Release 7
- RHEL 8 STIG Version 1 Release 6
- Ubuntu 18.04 STIG Version 2 Release 6 (nouveau)
- Ubuntu 20.04 STIG Version 1 Release 4 (nouveau)

STIG-Build-Linux-Medium version 2022.2.x

- RHEL 7 STIG Version 3 Release 7

- RHEL 8 STIG Version 1 Release 6
- Ubuntu 18.04 STIG Version 2 Release 6 (nouveau)
- Ubuntu 20.04 STIG Version 1 Release 4 (nouveau)

STIG-Build-Linux-High version 2022.2.x

- RHEL 7 STIG Version 3 Release 7
- RHEL 8 STIG Version 1 Release 6
- Ubuntu 18.04 STIG Version 2 Release 6 (nouveau)
- Ubuntu 20.04 STIG Version 1 Release 4 (nouveau)

Modifications apportées au premier trimestre 2022 – 26/04/2022 :

Refactorisé pour inclure un meilleur support pour les conteneurs. Combinaison du AL2 script précédent avec RHEL 7. Versions STIG mises à jour et STIG appliqués pour la version du premier trimestre 2022 comme suit :

STIG-Build-Linux-Low version 3.6.x

- RHEL 7 STIG Version 3 Release 6
- RHEL 8 STIG Version 1 Release 5

STIG-Build-Linux-Medium version 3.6.x

- RHEL 7 STIG Version 3 Release 6
- RHEL 8 STIG Version 1 Release 5

STIG-Build-Linux-High version 3.6.x

- RHEL 7 STIG Version 3 Release 6
- RHEL 8 STIG Version 1 Release 5

Modifications apportées au quatrième trimestre 2021 – 20/12/2021 :

Versions STIG mises à jour et STIG appliqués pour la version du quatrième trimestre 2021 comme suit :

STIG-Build-Linux-Low version 3.5.x

- RHEL 7 STIG Version 3 Release 5
- RHEL 8 STIG Version 1 Release 4

STIG-Build-Linux-Medium version 3.5.x

- RHEL 7 STIG Version 3 Release 5
- RHEL 8 STIG Version 1 Release 4

STIG-Build-Linux-High version 3.5.x

- RHEL 7 STIG Version 3 Release 5
- RHEL 8 STIG Version 1 Release 4

Modifications apportées au troisième trimestre 2021 au 30/09/2021 :

Versions STIG mises à jour et STIG appliqués pour la version du troisième trimestre 2021 comme suit :

STIG-Build-Linux-Low version 3.4.x

- RHEL 7 STIG Version 3 Release 4
- RHEL 8 STIG Version 1 Release 3

STIG-Build-Linux-Medium version 3.4.x

- RHEL 7 STIG Version 3 Release 4
- RHEL 8 STIG Version 1 Release 3

STIG-Build-Linux-High version 3.4.x

- RHEL 7 STIG Version 3 Release 4
- RHEL 8 STIG Version 1 Release 3

Composant de validation de conformité SCAP

Le protocole SCAP (Security Content Automation Protocol) est un ensemble de normes que les professionnels de l'informatique peuvent utiliser pour identifier les vulnérabilités de sécurité des applications à des fins de conformité. Le SCAP Compliance Checker (SCC) est un outil d'analyse validé par le SCAP, publié par le Naval Information Warfare Center (NIWC) Atlantic. Pour plus d'informations, voir Vérificateur de conformité au protocole SCAP (Security Content Automation Protocol) (SCC) sur le site Web de NIWC Atlantic.

Les `scap-compliance-checker-linux` composants AWSTOE `scap-compliance-checker-windows` et téléchargent et installent le scanner SCC sur les instances de construction et de test du pipeline. Lorsque le scanner s'exécute, il effectue des analyses de configuration authentifiées à l'aide des benchmarks DISA SCAP et fournit un rapport contenant les informations suivantes. AWSTOE écrit également les informations dans les journaux de vos applications.

- Paramètres STIG appliqués à l'instance.
- Un score de conformité global pour l'instance.

Nous vous recommandons d'exécuter la validation SCAP comme dernière étape de votre processus de création, afin de garantir l'exactitude des résultats de validation de conformité.

> **Note**
>
> Vous pouvez consulter les rapports à l'aide de l'un des outils de visualisation STIG. Ces outils sont disponibles en ligne via le DoD Cyber Exchange.

Les sections suivantes décrivent les benchmarks inclus dans les composants de validation SCAP.

scap-compliance-checker-windows version 2024.03.0

Le `scap-compliance-checker-windows` composant s'exécute sur les instances EC2 créées par Image Builder pour créer et tester l'image. AWSTOE enregistre à la fois le rapport et le score produits par l'application SCC.

Le composant exécute les étapes de flux de travail suivantes :

1. Télécharge et installe l'application SCC.

2. Importe les points de référence de conformité.
3. Exécute la validation à l'aide de l'application SCC.
4. Enregistre le rapport de conformité et le score localement sur le poste de travail de l'instance de build.
5. Enregistre le score de conformité du rapport local dans les fichiers journaux de AWSTOE l'application.

> **Note**
>
> AWSTOE prend actuellement en charge la validation de conformité SCAP pour Windows Server 2012 R2 MS, 2016, 2019 et 2022.

Le composant du vérificateur de conformité SCAP pour Windows inclut les tests de performance suivants :

Version SCC : 5.10

Points de référence du quatrième trimestre 2023 :

- U_MS_Defender_Antivirus_V2R5_Stig_Scap_1-2_Benchmark
- U_MS_ _Framework_4-0_V2R2_STIG_SCAP_1-2_Benchmark DotNet
- U_MS_ _V2R6_Stig_Scap_1-2_Benchmark IE11
- U_MS_Windows_2012_et_2012_R2_DC_V3R5_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_2012_et_2012_R2_MS_V3R5_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Defender_Firewall_v2r3_Stig_Scap_1-2_Benchmark
- U_MS_Windows_Server_2016_V2R7_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2019_V3R2_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2022_V2R2_STIG_SCAP_1-2_Benchmark
- U_CAN_Ubuntu_20-04_LTS_V1R10_STIG_SCAP_1-2_Benchmark
- U_RHEL_7_V3R15_Stig_Scap_1-3_Benchmark
- U_RHEL_8_V1R13_Stig_Scap_1-3_Benchmark
- U_RHEL_9_V2R1_STIG_SCAP_1-3_Benchmark

scap-compliance-checker-linux version 2021.04.0

Le `scap-compliance-checker-linux` composant s'exécute sur les instances EC2 créées par Image Builder pour créer et tester l'image. AWSTOE enregistre à la fois le rapport et le score produits par l'application SCC.

Le composant exécute les étapes de flux de travail suivantes :

1. Télécharge et installe l'application SCC.
2. Importe les points de référence de conformité.
3. Exécute la validation à l'aide de l'application SCC.
4. Enregistre le rapport de conformité et le score localement, à l'emplacement suivant sur l'instance de build :`/opt/scc/SCCResults`.
5. Enregistre le score de conformité du rapport local dans les fichiers journaux de AWSTOE l'application.

> **Note**
>
> AWSTOE prend actuellement en charge la validation de conformité SCAP pour RHEL 7/8 et Ubuntu 18.04/20.04. L'application SCC prend actuellement en charge l'architecture x86 pour la validation.

Le composant du vérificateur de conformité SCAP pour Linux inclut les tests de performance suivants :

Version SCC : 5.10

Points de référence du quatrième trimestre 2023 :

- U_CAN_Ubuntu_20-04_LTS_V1R10_STIG_SCAP_1-2_Benchmark
- U_RHEL_7_V3R15_Stig_Scap_1-3_Benchmark
- U_RHEL_8_V1R13_Stig_Scap_1-3_Benchmark
- U_RHEL_9_V2R1_STIG_SCAP_1-3_Benchmark
- U_MS_Defender_Antivirus_V2R5_Stig_Scap_1-2_Benchmark
- U_MS_ _Framework_4-0_V2R2_STIG_SCAP_1-2_Benchmark DotNet

- U_MS_ _V2R6_Stig_Scap_1-2_Benchmark IE11
- U_MS_Windows_2012_et_2012_R2_DC_V3R5_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_2012_et_2012_R2_MS_V3R5_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Defender_Firewall_v2r3_Stig_Scap_1-2_Benchmark
- U_MS_Windows_Server_2016_V2R7_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2019_V3R2_STIG_SCAP_1-2_Benchmark
- U_MS_Windows_Server_2022_V2R2_STIG_SCAP_1-2_Benchmark

Historique des versions de SCAP

Le tableau suivant décrit les modifications importantes apportées à l'environnement et aux paramètres SCAP décrits dans ce document.

Modifier	Description	Date
Mises à jour du SCAP du premier trimestre 2025	• scap-compliance-checker-windows version 2024.03.0 (version SCC : 5.10) • scap-compliance-checker-windows Version 2025.01.0 (Version SCC : 5.10)	11 avril 2025
Mises à jour du SCAP du quatrième trimestre 2023	• scap-compliance-checker-windows version 2023.04.0 (version SCC : 5.8) • scap-compliance-checker-windows version 2023.04.0 (version SCC : 5.8)	20 décembre 2021
Mises à jour du SCAP du troisième trimestre 2023	• scap-compliance-checker-windows version 2023.03.0 (version SCC : 5.7.2) • scap-compliance-checker-linux version 2023.03.0 (version SCC : 5.7.2)	13 novembre 2023

Modifier	Description	Date
Composants SCAP ajoutés	Les composants SCAP suivants ont été introduits : • scap-compliance-checker-linuxVersion créée 2021.04.0 (Version SCC : 5.4.2) • scap-compliance-checker-linuxVersion créée 2021.04.0 (Version SCC : 5.4.2)	20 décembre 2021

Développez des composants personnalisés pour votre image Image Builder

Vous pouvez créer vos propres composants pour personnaliser vos images Image Builder en fonction de vos spécifications exactes. Suivez les étapes ci-dessous pour développer des composants personnalisés pour votre image Image Builder ou vos recettes de conteneurs.

1. Si vous souhaitez développer votre document de composant et le valider localement, vous pouvez installer l'application AWS Task Orchestrator and Executor (AWSTOE) et la configurer sur votre machine locale. Pour de plus amples informations, veuillez consulter Configuration manuelle pour développer des composants personnalisés avec AWSTOE.
2. Créez un document de composant qui utilise la structure du document de AWSTOE composant. Pour plus d'informations sur le cadre documentaire, consultezUtilisez la commande AWSTOE cadre de documentation sur les composants pour les composants personnalisés.
3. Spécifiez le document de votre composant lorsque vous créez un composant personnalisé. Pour de plus amples informations, veuillez consulter Création d'un composant personnalisé avec Image Builder.

Rubriques

- Création d'un document de composant YAML pour les composants personnalisés dans Image Builder
- Création d'un composant personnalisé avec Image Builder

Création d'un document de composant YAML pour les composants personnalisés dans Image Builder

Pour créer un composant, vous devez fournir un document de composant d'application YAML ou JSON. Le document contient le code qui s'exécute pendant les phases et les étapes que vous définissez pour personnaliser votre image.

Certains des exemples présentés dans cette section créent un composant de génération qui appelle le module `UpdateOS` d'action dans l'application de gestion des AWSTOE composants. Le module met à jour le système d'exploitation. Pour plus d'informations sur le module `UpdateOS` d'action, consultezMettre à jour le système d'exploitation.

L'exemple du système d'exploitation macOS utilise le module `ExecuteBash` d'action pour installer et vérifier l'`wget`utilitaire. Le module `UpdateOS` d'action n'est pas compatible avec macOS. Pour plus d'informations sur le module `ExecuteBash` d'action, consultezExecuteBash. Pour plus d'informations sur les phases, les étapes et la syntaxe des documents des composants de AWSTOE l'application, voir Utiliser des documents dans AWSTOE.

> **Note**
>
> Image Builder détermine le type de composant à partir des phases définies dans le document du composant comme suit :
>
> - Construire — Il s'agit du type de composant par défaut. Tout ce qui n'est pas classé comme composant de test est un composant de construction. Ce type de composant s'exécute pendant la phase de création de l'image. Si une `test` phase est définie pour ce composant de construction, celle-ci s'exécute pendant la phase de test.
> - Test — Pour être considéré comme un composant de test, le document du composant ne doit inclure qu'une seule phase, nommée`test`. Pour les tests liés à la configuration des composants de génération, nous vous recommandons de ne pas utiliser de composant de test autonome. Utilisez plutôt la `test` phase dans le composant de construction associé.
>
> Pour plus d'informations sur la manière dont Image Builder utilise les étapes et les phases pour gérer le flux de travail des composants dans son processus de création, consultezUtilisez des composants pour personnaliser l'image de votre Image Builder.

Pour créer un document de composant d'application YAML pour un exemple d'application, suivez les étapes de l'onglet correspondant à votre système d'exploitation d'image.

Linux

Création d'un fichier de composant YAML

Utilisez un outil d'édition de fichiers pour créer votre document de composant. Les exemples de documentation utilisent un fichier nommé*update-linux-os.yaml*, dont le contenu est le suivant :

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-linux-os
description: Updates Linux with the latest security updates.
schemaVersion: 1
phases:
  - name: build
    steps:
    - name: UpdateOS
      action: UpdateOS
# Document End
```

> **Tip**
>
> Utilisez un outil tel que ce validateur YAML en ligne ou une extension YAML lint dans votre environnement de code pour vérifier que votre code YAML est bien formé.

Windows

Création d'un fichier de composant YAML

Utilisez un outil d'édition de fichiers pour créer votre document de composant. Les exemples de documentation utilisent un fichier nommé*update-windows-os.yaml*, dont le contenu est le suivant :

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-windows-os
description: Updates Windows with the latest security updates.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: UpdateOS
        action: UpdateOS
# Document End
```

> **Tip**
>
> Utilisez un outil tel que ce validateur YAML en ligne ou une extension YAML lint dans votre environnement de code pour vérifier que votre code YAML est bien formé.

macOS

Création d'un fichier de composant YAML

Utilisez un outil d'édition de fichiers pour créer votre document de composant. Les exemples de documentation utilisent un fichier nommé*wget-macos.yaml*, dont le contenu est le suivant :

```
name: WgetInstallDocument
description: This is wget installation document.
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: WgetBuildStep
        action: ExecuteBash
        inputs:
          commands:
            - |
              PATH=/usr/local/bin:$PATH
              sudo -u ec2-user brew install wget


  - name: validate
    steps:
      - name: WgetValidateStep
        action: ExecuteBash
        inputs:
          commands:
            - |
              function error_exit {
                echo $1
                echo "{\"failureMessage\":\"$2\"}"
                exit 1
              }
```

```
            type wget
            if [ $? -ne 0 ]; then
              error_exit "$stderr" "Wget installation failed!"
            fi

- name: test
  steps:
    - name: WgetTestStep
      action: ExecuteBash
      inputs:
        commands:
          - wget -h
```

> **Tip**
>
> Utilisez un outil tel que ce validateur YAML en ligne ou une extension YAML lint dans votre environnement de code pour vérifier que votre code YAML est bien formé.

Création d'un composant personnalisé avec Image Builder

Une fois que vous avez terminé le document de composant, vous pouvez l'utiliser pour créer un composant personnalisé que vos recettes Image Builder peuvent utiliser. Vous pouvez créer un composant personnalisé à partir de la console Image Builder, de l'API ou SDKs de la ligne de commande. Pour plus d'informations sur la façon de créer un composant personnalisé avec des paramètres d'entrée et de l'utiliser dans vos recettes, consultezTutoriel : Création d'un composant personnalisé avec des paramètres d'entrée.

Les sections suivantes expliquent comment créer des composants à partir de la console ou du AWS CLI.

Table des matières

- Création d'un composant personnalisé depuis la console
- Créez un composant personnalisé à partir du AWS CLI
- Importez un script pour créer un composant à partir du AWS CLI
- Gestion automatique des versions de build
- Utilisation des références de version

Création d'un composant personnalisé depuis la console

Pour créer un composant d' AWSTOE application à partir de la console Image Builder, procédez comme suit :

1. Ouvrez la console https://console.aws.amazon.com/imagebuilder/EC2 Image Builder à l'adresse.
2. Sélectionnez Composants dans le volet de navigation. Sélectionnez ensuite Créer un composant.
3. Sur la page Créer un composant, sous Détails du composant, entrez ce qui suit :

   a. Système d'exploitation d'image (OS). Spécifiez le système d'exploitation avec lequel le composant est compatible.

   b. Catégorie de composant. Dans le menu déroulant, sélectionnez le type de composant de construction ou de test que vous créez.

   c. Nom du composant. Entrez un nom pour le composant.

   d. Version du composant. Entrez le numéro de version du composant.

   e. Description. Fournissez une description facultative pour vous aider à identifier le composant.

   f. Modifier la description. Fournissez une description facultative pour vous aider à comprendre les modifications apportées à cette version du composant.
4. Dans la section Définition du document, l'option par défaut est Définir le contenu du document. Le document du composant définit les actions qu'Image Builder exécute sur les instances de génération et de test pour créer votre image.

   Dans le champ Contenu, entrez le contenu du document de votre composant YAML. Pour commencer par un exemple de Hello World pour Linux, choisissez l'option Utiliser un exemple. Pour en savoir plus sur la création d'un document de composant YAML ou sur le copier-coller de l'exemple UpdateOS à partir de cette page, consultez. Création d'un document de composant YAML pour les composants personnalisés dans Image Builder
5. Après avoir saisi les détails du composant, sélectionnez Créer un composant.

> **Note**
>
> Pour voir votre nouveau composant lorsque vous créez ou mettez à jour une recette, appliquez le filtre Owned by me à la liste des composants de construction ou de test. Le filtre se trouve en haut de la liste des composants, à côté du champ de recherche.

6. Pour supprimer un composant, sur la page Composants, cochez la case à côté du composant que vous souhaitez supprimer. Dans le menu déroulant Actions, sélectionnez Supprimer le composant.

Mettre à jour un composant

Pour créer une nouvelle version de composant, procédez comme suit :

1. En fonction de votre point de départ :
   - Sur la page de liste des composants : cochez la case à côté du nom du composant, puis sélectionnez Créer une nouvelle version dans le menu Actions.
   - Sur la page détaillée du composant : cliquez sur le bouton Créer une nouvelle version dans le coin supérieur droit de l'en-tête.
2. Les informations du composant sont déjà renseignées avec les valeurs actuelles lorsque la page Créer un composant s'affiche. Suivez les étapes de création d'un composant pour le mettre à jour. Cela garantit que vous entrez une version sémantique unique dans la version du composant. Pour en savoir plus sur le versionnement sémantique des ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

Créez un composant personnalisé à partir du AWS CLI

Dans cette section, vous allez apprendre à configurer et à utiliser les commandes Image Builder AWS CLI pour créer un composant d' AWSTOE application, comme suit.

- Téléchargez le document de votre composant YAML dans un compartiment S3 auquel vous pouvez faire référence depuis la ligne de commande.
- Créez le composant AWSTOE d'application à l'aide de la create-component commande.
- Répertoriez les versions des composants à l'aide de la list-components commande et d'un filtre de nom pour voir quelles versions existent déjà. Vous pouvez utiliser le résultat pour déterminer quelle sera la prochaine version pour les mises à jour.

Pour créer un composant d' AWSTOE application à partir d'un document YAML d'entrée, suivez les étapes correspondant à la plate-forme de votre système d'exploitation d'image.

Linux

Stockez le document relatif aux composants de votre application dans Amazon S3

Vous pouvez utiliser un compartiment S3 comme référentiel pour le document source des composants de votre AWSTOE application. Pour enregistrer le document de votre composant, procédez comme suit :

- Téléchargez le document sur Amazon S3

  Si la taille de votre document est inférieure à 64 Ko, vous pouvez ignorer cette étape. Les documents d'une taille supérieure ou égale à 64 Ko doivent être stockés dans Amazon S3.

  ```
  aws s3 cp update-linux-os.yaml s3://amzn-s3-demo-destination-bucket/my-path/update-linux-os.yaml
  ```

Création d'un composant à partir du document YAML

Pour rationaliser la create-component commande que vous utilisez dans le AWS CLI, créez un fichier JSON contenant tous les paramètres de composant que vous souhaitez transmettre à la commande. Indiquez l'emplacement du *update-linux-os.yaml* document que vous avez créé précédemment. La paire `uri` clé-valeur contient la référence du fichier.

> **Note**
>
> La convention de dénomination des valeurs de données dans le fichier JSON suit le modèle spécifié pour les paramètres de demande d'opération de l'API Image Builder. Pour consulter les paramètres de demande de commande d'API, consultez la CreateComponentcommande dans le manuel de référence de l'API EC2 Image Builder. Pour fournir les valeurs de données sous forme de paramètres de ligne de commande, reportez-vous aux noms de paramètres spécifiés dans la référence des AWS CLI commandes.

1. Créer un fichier JSON d'entrée CLI

   Utilisez un outil d'édition de fichiers pour créer un fichier nommé*create-update-linux-os-component.json*. Incluez le contenu suivant :

```
{
 "name": "update-linux-os",
 "semanticVersion": "1.1.2",
 "description": "An example component that updates the Linux operating system",
 "changeDescription": "Initial version.",
 "platform": "Linux",
 "uri": "s3://amzn-s3-demo-destination-bucket/my-path/update-linux-os.yaml",
 "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
b123-456b-7f89-0123456f789c",
 "tags": {
  "MyTagKey-purpose": "security-updates"
 }
}
```

2. Création du composant

   Utilisez la commande suivante pour créer le composant, en faisant référence au nom du fichier JSON que vous avez créé à l'étape précédente :

   ```
   aws imagebuilder create-component --cli-input-json file://create-update-linux-
   os-component.json
   ```

   > **Note**
   >
   > - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
   > - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Windows

Stockez le document relatif aux composants de votre application dans Amazon S3

Vous pouvez utiliser un compartiment S3 comme référentiel pour le document source des composants de votre AWSTOE application. Pour enregistrer le document de votre composant, procédez comme suit :

- Téléchargez le document sur Amazon S3

  Si la taille de votre document est inférieure à 64 Ko, vous pouvez ignorer cette étape. Les documents d'une taille supérieure ou égale à 64 Ko doivent être stockés dans Amazon S3.

  ```
  aws s3 cp update-windows-os.yaml s3://amzn-s3-demo-destination-bucket/my-path/update-windows-os.yaml
  ```

Création d'un composant à partir du document YAML

Pour rationaliser la create-component commande que vous utilisez dans le AWS CLI, créez un fichier JSON contenant tous les paramètres de composant que vous souhaitez transmettre à la commande. Indiquez l'emplacement du *update-windows-os.yaml* document que vous avez créé précédemment. La paire `uri` clé-valeur contient la référence du fichier.

> **Note**
>
> La convention de dénomination des valeurs de données dans le fichier JSON suit le modèle spécifié pour les paramètres de demande d'opération de l'API Image Builder. Pour consulter les paramètres de demande de commande d'API, consultez la CreateComponentcommande dans le manuel de référence de l'API EC2 Image Builder. Pour fournir les valeurs de données sous forme de paramètres de ligne de commande, reportez-vous aux noms de paramètres spécifiés dans la référence des AWS CLI commandes.

1. Créer un fichier JSON d'entrée CLI

   Utilisez un outil d'édition de fichiers pour créer un fichier nommé*create-update-windows-os-component.json*. Incluez le contenu suivant :

   ```
   {
    "name": "update-windows-os",
    "semanticVersion": "1.1.2",
    "description": "An example component that updates the Windows operating system.",
    "changeDescription": "Initial version.",
    "platform": "Windows",
    "uri": "s3://amzn-s3-demo-destination-bucket/my-path/update-windows-os.yaml",
   ```

```
 "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
b123-456b-7f89-0123456f789c",
 "tags": {
  "MyTagKey-purpose": "security-updates"
 }
}
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

2. Création du composant

Utilisez la commande suivante pour créer le composant, en faisant référence au nom du fichier JSON que vous avez créé à l'étape précédente :

```
aws imagebuilder create-component --cli-input-json file://create-update-windows-
os-component.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

macOS

Stockez le document relatif aux composants de votre application dans Amazon S3

Vous pouvez utiliser un compartiment S3 comme référentiel pour le document source des composants de votre AWSTOE application. Pour enregistrer le document de votre composant, procédez comme suit :

- Téléchargez le document sur Amazon S3

  Si la taille de votre document est inférieure à 64 Ko, vous pouvez ignorer cette étape. Les documents d'une taille supérieure ou égale à 64 Ko doivent être stockés dans Amazon S3.

  ```
  aws s3 cp wget-macos.yaml s3://amzn-s3-demo-destination-bucket/my-path/wget-macos.yaml
  ```

Création d'un composant à partir du document YAML

Pour rationaliser la create-component commande que vous utilisez dans le AWS CLI, créez un fichier JSON contenant tous les paramètres de composant que vous souhaitez transmettre à la commande. Indiquez l'emplacement du *wget-macos.yaml* document que vous avez créé précédemment. La paire `uri` clé-valeur contient la référence du fichier.

> **Note**
>
> La convention de dénomination des valeurs de données dans le fichier JSON suit le modèle spécifié pour les paramètres de demande d'opération de l'API Image Builder. Pour consulter les paramètres de demande de commande d'API, consultez la CreateComponentcommande dans le manuel de référence de l'API EC2 Image Builder. Pour fournir les valeurs de données sous forme de paramètres de ligne de commande, reportez-vous aux noms de paramètres spécifiés dans la référence des AWS CLI commandes.

1. Créer un fichier JSON d'entrée CLI

   Utilisez un outil d'édition de fichiers pour créer un fichier nommé*install-wget-macos-component.json*. Incluez le contenu suivant :

```
{
 "name": "install install-wget-macos-component",
 "semanticVersion": "1.1.2",
 "description": "An example component that installs and verifies the wget
 utility on macOS.",
 "changeDescription": "Initial version.",
 "platform": "macOS",
 "uri": "s3://amzn-s3-demo-destination-bucket/my-path/wget-macos.yaml",
 "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
b123-456b-7f89-0123456f789c",
 "tags": {
  "MyTagKey-purpose": "install-software"
 }
}
```

2. Création du composant

   Utilisez la commande suivante pour créer le composant, en faisant référence au nom du fichier JSON que vous avez créé à l'étape précédente :

   ```
   aws imagebuilder create-component --cli-input-json file://install-wget-macos-
   component.json
   ```

   > **Note**
   >
   > - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
   > - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

AWSTOE gestion des versions des composants pour les mises à jour depuis AWS CLI

AWSTOE les noms et versions des composants sont intégrés dans le nom de ressource Amazon (ARN) du composant, après le préfixe du composant. Chaque nouvelle version d'un composant possède son propre ARN unique. Les étapes de création d'une nouvelle version sont exactement les mêmes que celles de création d'un nouveau composant, à condition que la version sémantique

soit unique pour le nom de ce composant. Pour en savoir plus sur le versionnement sémantique des ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

Pour être sûr d'attribuer la version logique suivante, obtenez d'abord une liste des versions existantes du composant que vous souhaitez modifier. Utilisez la list-components commande avec le AWS CLI, et filtrez sur le nom.

Dans cet exemple, vous filtrez sur le nom du composant que vous avez créé dans les exemples Linux précédents. Pour répertorier le composant que vous avez créé, utilisez la valeur du `name` paramètre du fichier JSON que vous avez utilisé dans la create-component commande.

```
aws imagebuilder list-components --filters name="name",values="update-linux-os"
{
    "requestId": "123a4567-b890-123c-45d6-ef789ab0cd1e",
    "componentVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-
linux-os/1.0.0",
            "name": "update-linux-os",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-
linux-os/1.0.1",
            "name": "update-linux-os",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

Sur la base de vos résultats, vous pouvez déterminer quelle devrait être la prochaine version.

Importez un script pour créer un composant à partir du AWS CLI

Dans certains scénarios, il peut être plus facile de démarrer avec un script préexistant. Pour ce scénario, vous pouvez utiliser l'exemple suivant.

Cet exemple suppose que vous disposez d'un fichier appelé *import-component.json* (comme indiqué). Notez que le fichier fait directement référence à un PowerShell script appelé `AdminConfig.ps1` qui a déjà été téléchargé vers*amzn-s3-demo-source-bucket*. Actuellement, `SHELL` est pris en charge pour le composant`format`.

```
{
"name": "MyImportedComponent",
"semanticVersion": "1.0.0",
"description": "An example of how to import a component",
"changeDescription": "First commit message.",
"format": "SHELL",
"platform": "Windows",
"type": "BUILD",
"uri": "s3://amzn-s3-demo-source-bucket/AdminConfig.ps1",
"kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/60763706-
b131-418b-8f85-3420912f020c"
}
```

Pour créer le composant à partir d'un script importé, exécutez la commande suivante.

```
aws imagebuilder import-component --cli-input-json file://import-component.json
```

Gestion automatique des versions de build

Lorsque vous créez un composant portant le même nom et la même version sémantique qu'un composant existant, Image Builder incrémente automatiquement la version de construction (par exemple, de `/1` à`/2`, à`/3`, etc.). Cela vous permet d'effectuer des mises à jour itératives de vos composants sans gérer manuellement les numéros de version, ce qui est particulièrement utile dans les CI/CD pipelines et les infrastructure-as-code déploiements. Si le contenu du composant est identique à celui de la version précédente, Image Builder revient `ResourceAlreadyExistsException` pour empêcher les composants dupliqués d'utiliser votre quota de service.

Utilisation des références de version

Lorsque vous créez ou récupérez un composant, Image Builder fournit automatiquement des modèles de version préconstruits ARNs avec des caractères génériques dans l' latestVersionReferences objet. Ces références facilitent l'utilisation des dernières versions de vos composants dans les recettes et les pipelines sans analyse ARNs manuelle.

Choisir la bonne référence de version

- latestVersionArn (x.x.x) - Utilisez toujours la dernière version du composant.
- atestMajorVersionArn (1.x.x) - Utilisez les dernières versions mineures et patchs dans une version majeure.
- latestMinorVersionArn (1.2.x) - Utilisez uniquement la dernière version du correctif.
- latestPatchVersionArn (1.2.3) - Référencez une version sémantique spécifique, mais obtenez la dernière version de compilation.

> **Note**
>
> Pour éviter des frais imprévus, veillez à nettoyer les ressources et les pipelines que vous avez créés à partir des exemples de ce guide. Pour plus d'informations sur la suppression de ressources dans Image Builder, consultezSupprimer les ressources d'Image Builder obsolètes ou inutilisées.

Comment Image Builder utilise AWS Task Orchestrator and Executor application pour gérer les composants

EC2 Image Builder utilise l'application AWS Task Orchestrator and Executor (AWSTOE) pour orchestrer des flux de travail complexes, modifier les configurations système et tester vos images sans avoir besoin de scripts ou de code DevOps supplémentaires. Cette application gère et exécute les composants qui implémentent son schéma de document déclaratif.

AWSTOE est une application autonome qu'Image Builder installe sur ses instances de génération et de test lorsque vous créez une image. Vous pouvez également l'installer manuellement sur les instances EC2 pour créer vos propres composants personnalisés. Il ne nécessite aucune configuration supplémentaire et peut également fonctionner sur site.

> **Tip**
>
> Pour plus d'informations sur les modèles de composants et sur un script qui valide et exécute les documents des composants localement, consultez les AWSTOE exemples sur GitHub. Pour de plus amples informations, veuillez consulter Découvrez des exemples de projets Image Builder sur GitHub.

Table des matières

- AWSTOE downloads
- Régions prises en charge
- AWSTOE référence des commandes
- Configuration manuelle pour développer des composants personnalisés avec AWSTOE
- Utilisez la commande AWSTOE cadre de documentation sur les composants pour les composants personnalisés
- Modules d'action pris en charge par AWSTOE gestionnaire de composants
- Configurez l'entrée pour AWSTOE exécuter la commande

AWSTOE downloads

Pour l'installer AWSTOE, choisissez le lien de téléchargement correspondant à votre architecture et à votre plateforme. Si vous vous connectez à un point de terminaison VPC pour votre service (Image Builder, par exemple), celui-ci doit être associé à une politique de point de terminaison personnalisée qui inclut l'accès au compartiment S3 pour les AWSTOE téléchargements. Sinon, vos instances de compilation et de test ne pourront pas télécharger le script bootstrap (`bootstrap.sh`) et installer l' AWSTOE application. Pour de plus amples informations, veuillez consulter Création d'une politique de point de terminaison VPC pour Image Builder.

> **Important**
>
> AWS supprime progressivement la prise en charge des versions 1.0 et 1.1 de TLS. Pour accéder au compartiment S3 à des fins de AWSTOE téléchargement, votre logiciel client doit utiliser la version 1.2 ou ultérieure du protocole TLS. Pour plus d'informations, consultez ce billet AWS de blog sur la sécurité.

Architecture	Plateforme	Lien de téléchargement	Exemple
386	AL 2 et 2023 RHEL 7, 8 et 9 Ubuntu 16.04, 18.04, 20.04, 22.04 et 24.04 CentOS 7 et 8 SUSE 12 et 15	`https://awstoe-<region>.s3.com/latest/linux/386/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/386/awstoe
AMD64	AL 2 et 2023 RHEL 7, 8 et 9 Ubuntu 16.04, 18.04, 20.04, 22.04 et 24.04 CentOS 7 et 8 CentOS Stream 8 SUSE 12 et 15	`https://awstoe-<region>.s3.com/latest/linux/amd64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/awstoe
AMD64	macOS 10.14.x (Mojave), 10.15.x (Catalina), 11.x (Big Sur), 12.x (Monterey)	`https://awstoe-region.s3.com/latest/darwin/amd64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/darwin/amd64/awstoe
AMD64	Windows Server 2012 R2, 2016, 2019 et 2022	`https://awstoe-<region>.s3.com/latest/windows/amd64/awstoe.exe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/windows/amd64/awstoe.exe

Architecture	Plateforme	Lien de téléchargement	Exemple
ARM64	AL 2 et 2023 RHEL 7, 8 et 9 Ubuntu 16.04, 18.04, 20.04, 22.04 et 24.04 CentOS 7 et 8 CentOS Stream 8 SUSE 12 et 15	`https://awstoe- <region>.s3.com/latest/linux/arm64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/arm64/awstoe

Régions prises en charge

AWSTOE est pris en charge en tant qu'application autonome dans les régions suivantes.

Région AWS nom	Région AWS
USA Est (Ohio)	us-east-2
USA Est (Virginie du Nord)	us-east-1
AWS GovCloud (US-East)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1
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 (Osaka)	ap-northeast-3

Région AWS nom	Région AWS
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Mumbai)	ap-south-1
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 (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Francfort)	eu-central-1
Europe (Zurich)	eu-central-2
Europe (Stockholm)	eu-north-1
Europe (Milan)	eu-south-1
Europe (Espagne)	eu-south-2
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Paris)	eu-west-3
Israël (Tel Aviv)	il-central-1
Moyen-Orient (EAU)	me-central-1
Moyen-Orient (Bahreïn)	me-south-1
Amérique du Sud (São Paulo)	sa-east-1

Région AWS nom	Région AWS
Chine (Beijing)	cn-north-1
China (Ningxia)	cn-northwest-1

AWSTOE référence des commandes

AWSTOE est une application de gestion de composants en ligne de commande qui s'exécute sur les instances Amazon EC2. Lorsqu'Image Builder lance une instance de génération ou de test EC2, il s'installe AWSTOE sur l'instance. Ensuite, il exécute AWSTOE les commandes AWS CLI pour installer ou valider les composants spécifiés dans l'image ou la recette du conteneur.

> **Note**
>
> Certains modules AWSTOE d'action nécessitent des autorisations élevées pour s'exécuter sur un serveur Linux. Pour utiliser des autorisations élevées, préfixez la syntaxe de la commande avec sudo ou exécutez la sudo su commande une fois lorsque vous vous connectez avant d'exécuter les commandes liées ci-dessous. Pour plus d'informations sur les modules AWSTOE d'action, consultezModules d'action pris en charge par AWSTOE gestionnaire de composants.

run

Utilisez la run commande pour exécuter les scripts de document YAML pour un ou plusieurs documents de composants.

valider

Exécutez la validate commande pour valider la syntaxe du document YAML pour un ou plusieurs documents de composants.

commande awstoe run

Cette commande exécute les scripts des documents de composants YAML dans l'ordre dans lequel ils sont inclus dans le fichier de configuration spécifié par le `--config` paramètre ou dans la liste des documents de composants spécifiée par le `--documents` paramètre.

> **Note**
>
> Vous devez spécifier exactement l'un des paramètres suivants, jamais les deux :
> --configuration
> --documents

Syntaxe

```
awstoe run [--config <file path>] [--cw-ignore-failures <?>]
      [--cw-log-group <?>] [--cw-log-region us-west-2] [--cw-log-stream <?>]
      [--document-s3-bucket-owner <owner>] [--documents <file path,file path,...>]
      [--execution-id <?>] [--log-directory <file path>]
      [--log-s3-bucket-name <name>] [--log-s3-bucket-owner <owner>]
      [--log-s3-key-prefix <?>] [--parameters name1=value1,name2=value2...]
      [--phases <phase name>] [--blocked-action-modules <module,module,...>]
      [--state-directory <directory path>] [--version <?>]
      [--help] [--trace]
```

Paramètres et options

Parameters

--configuration ***./config-example.json***

Forme abrégée : -c *./config-example.json*

Le fichier de configuration (conditionnel). Ce paramètre contient l'emplacement du fichier JSON qui contient les paramètres de configuration des composants que cette commande exécute. Si vous spécifiez des paramètres de run commande dans un fichier de configuration, vous ne devez pas les `--documents` spécifier. Pour plus d'informations sur la configuration des entrées, consultezConfigurez l'entrée pour AWSTOE exécuter la commande.

Les emplacements valides incluent :

- Un chemin de fichier local (*./config-example.json*)
- Un URI S3 (`s3://`*bucket/key*)

--cw-ignore-échecs

Forme abrégée : N/A

Ignorez les échecs de journalisation dans les CloudWatch journaux.

--cw-log-group

Forme abrégée : N/A

Le `LogGroup` nom des CloudWatch journaux.

--cw-log-region

Forme abrégée : N/A

La AWS région qui s'applique aux CloudWatch journaux.

--cw-log-stream

Forme abrégée : N/A

Le `LogStream` nom des CloudWatch journaux, qui indique AWSTOE où diffuser le `console.log` fichier.

--document-s3-bucket-owner

Forme abrégée : N/A

L'ID de compte du propriétaire du compartiment pour les URI-based documents S3.

--documents ***./doc-1.yaml,./doc-n.yaml***

Forme abrégée : -d, *./doc-1.yaml ./doc-n*

Les documents du composant (conditionnel). Ce paramètre contient une liste d'emplacements de fichiers séparés par des virgules pour les documents du composant YAML à exécuter. Si vous spécifiez des documents YAML pour la run commande à l'aide du `--documents` paramètre, vous ne devez pas spécifier le `--config` paramètre. N'incluez pas d'espaces entre les éléments de la liste.

Les emplacements valides incluent :

- chemins de fichiers locaux (*./component-doc-example.yaml*).
- URI S3 (`s3://`*bucket/key*).
- Version de compilation du composant Image Builder ARNs (arn:aws:imagebuilder:us-west-:component/ /2021.12. *2:123456789012 my-example-component* 02/1).

--identifiant d'exécution

Forme abrégée : -i

Il s'agit de l'identifiant unique qui s'applique à l'exécution de la run commande en cours. Cet ID est inclus dans les noms des fichiers de sortie et des fichiers journaux, afin d'identifier ces fichiers de manière unique et de les lier à l'exécution de la commande en cours. Si ce paramètre est omis, AWSTOE génère un GUID.

--log-directory

Forme abrégée : -l

Le répertoire de destination où sont AWSTOE stockés tous les fichiers journaux de l'exécution de cette commande. Par défaut, ce répertoire se trouve dans le répertoire parent suivant :`TOE_<DATETIME>_<EXECUTIONID>`. Si vous ne spécifiez pas le répertoire des journaux, AWSTOE utilise le répertoire de travail actuel (`.`).

--log-s3-bucket-name

Forme abrégée : -b

Si les journaux des composants sont stockés dans Amazon S3 (recommandé), AWSTOE télécharge les journaux de l'application des composants dans le compartiment S3 nommé dans ce paramètre.

--log-s3-bucket-owner

Forme abrégée : N/A

Si les journaux des composants sont stockés dans Amazon S3 (recommandé), il s'agit de l'ID du compte propriétaire du compartiment dans lequel AWSTOE les fichiers journaux sont écrits.

--log-s3-clé-préfixe

Forme abrégée : -k

Si les journaux des composants sont stockés dans Amazon S3 (recommandé), il s'agit du préfixe de la clé d'objet S3 pour l'emplacement du journal dans le compartiment.

--paramètres ***name1*** =***value1***, ***name2*** =***value2***...

Forme abrégée : N/A

Les paramètres sont des variables modifiables définies dans le document du composant, avec des paramètres que l'application appelante peut fournir lors de l'exécution.

--les phases

Forme abrégée : -p

Liste séparée par des virgules qui spécifie les phases à exécuter à partir des documents des composants YAML. Si un document de composant inclut des phases supplémentaires, celles-ci ne seront pas exécutées.

--annuaire-état

Forme abrégée : -s

Le chemin du fichier dans lequel les fichiers de suivi de l'état sont stockés.

--modules d'action bloqués

Forme abrégée : N/A

Liste de modules d'action séparés par des virgules qui ne peuvent pas être exécutés dans le document de composants. Si un document contient un module d'action bloqué, la validation échoue avant le début de l'exécution.

--version

Forme abrégée : -v

Spécifie la version de l'application du composant.

Options

--help

Forme abrégée : -h

Affiche un manuel d'aide pour utiliser les options de l'application de gestion des composants.

--trace

Forme abrégée : -t

Active une journalisation détaillée sur la console.

Output

La run commande produit un résumé de l'exécution JSON. L'exemple suivant montre le résultat d'une exécution réussie.

```
{
```

```
    "executionId": "12345678-abcd-efgh-1234-example12345",
    "status": "success",
    "failedStepCount": 0,
    "executedStepCount": 3,
    "ignoredFailedStepCount": 0,
    "skippedStepCount": 0,
    "failureMessage": "",
    "logUrl": "s3://my-bucket/my-prefix/12345678-abcd-efgh-1234-example12345"
}
```

Le `status` champ indique le résultat global de l'exécution :

- `success`— Toutes les phases et étapes ont été achevées avec succès.
- `failed`— Une ou plusieurs étapes ont échoué car l'on`Failure`action l'est`Abort`.
- `timedOut`— Une ou plusieurs étapes ont dépassé leur `timeoutSeconds` valeur.

commande awstoe validate

Lorsque vous exécutez cette commande, elle valide la syntaxe du document YAML pour chacun des documents de composants spécifiés par le `--documents` paramètre.

Syntaxe

```
awstoe validate [--blocked-action-modules <module,module,...>]
      [--document-s3-bucket-owner <owner>]
      --documents <file path,file path,...> [--help] [--trace]
```

Paramètres et options

Parameters

--modules d'action bloqués

Forme abrégée : N/A

Liste de modules d'action séparés par des virgules qui ne peuvent pas être exécutés dans le document de composants. Si un document contient un module d'action bloqué, la validation échoue.

--document-s3-bucket-owner

Forme abrégée : N/A

ID de compte source des URI-based documents S3 fournis.

--documents *./doc-1.yaml,./doc-n.yaml*

Forme abrégée : -d, *./doc-1.yaml ./doc-n*

Les documents des composants (obligatoire). Ce paramètre contient une liste d'emplacements de fichiers séparés par des virgules pour les documents des composants YAML à exécuter, sans espace entre les éléments. Les emplacements valides incluent :

- chemins de fichiers locaux (*./component-doc-example.yaml*)
- URI S3 () `s3://`*bucket/key*
- Version de compilation du composant Image Builder ARNs (arn:aws:imagebuilder:us-west-:component/ /2021.12. *2:123456789012 my-example-component* 02/1)

Options

--help

Forme abrégée : -h

Affiche un manuel d'aide pour utiliser les options de l'application de gestion des composants.

--trace

Forme abrégée : -t

Active une journalisation détaillée sur la console.

Output

La validate commande produit un résultat de validation JSON. L'exemple suivant montre le résultat d'une validation réussie.

```
{
    "validationStatus": "success",
    "message": "Validation successful."
}
```

L'exemple suivant montre la sortie lorsque la validation échoue.

```
{
    "validationStatus": "failed",
```

```
    "message": "Document has 0 phases. Phases list must not be empty."
}
```

Configuration manuelle pour développer des composants personnalisés avec AWSTOE

L'application AWS Task Orchestrator and Executor (AWSTOE) est une application autonome qui crée, valide et exécute des commandes dans un cadre de définition de composants. AWS les services peuvent être utilisés AWSTOE pour orchestrer les flux de travail, installer des logiciels, modifier les configurations du système et tester les versions d'images.

Procédez comme suit pour installer manuellement l' AWSTOE application et l'utiliser en tant qu'application autonome pour développer des composants personnalisés. Image Builder s'occupe de ces étapes pour vous, si vous utilisez la console ou les AWS CLI commandes Image Builder pour créer des composants personnalisés. Pour de plus amples informations, veuillez consulter Créez des composants personnalisés avec Image Builder.

Vérifiez la signature du téléchargement de AWSTOE l'installation

Cette section décrit le processus recommandé pour vérifier la validité du téléchargement d'installation pour AWSTOE les systèmes d'exploitation Linux, macOS et Windows.

Rubriques

- Vérifiez la signature du téléchargement de l' AWSTOE installation sous Linux ou macOS
- Vérifiez la signature du téléchargement de l' AWSTOE installation sous Windows

Vérifiez la signature du téléchargement de l' AWSTOE installation sous Linux ou macOS

Cette rubrique décrit le processus recommandé pour vérifier la validité du téléchargement d'installation pour les systèmes AWSTOE d'exploitation Linux ou macOS.

Chaque fois que vous téléchargez une application sur Internet, nous vous recommandons d'authentifier l'identité de l'éditeur du logiciel. Vérifiez également que l'application n'a pas été modifiée ou corrompue depuis sa publication. Cela vous évitera d'installer une version de l'application contenant un virus ou tout autre code malveillant.

Si, après avoir exécuté les étapes décrites dans cette rubrique, vous déterminez que le logiciel de l' AWSTOE application est modifié ou endommagé, n'exécutez pas le fichier d'installation. Contactez plutôt Support Pour plus d'informations sur vos options de support, consultez Support.

AWSTOE les fichiers pour les systèmes d'exploitation Linux et macOS sont signés à l'aide GnuPG d'une implémentation open source de la norme Pretty Good Privacy (OpenPGP) pour les signatures numériques sécurisées. GnuPG(également connu sous le nom deGPG) fournit une authentification et un contrôle d'intégrité par le biais d'une signature numérique. Amazon EC2 publie une clé publique et des signatures que vous pouvez utiliser pour vérifier les outils de la CLI Amazon EC2 téléchargés. Pour plus d'informations sur PGP et GnuPG (GPG), consultez http://www.gnupg.org.

La première étape consiste à établir une approbation avec l'éditeur du logiciel. Téléchargez la clé publique de l'éditeur du logiciel, vérifiez que le propriétaire de cette clé publique est bien celui qu'il prétend être, puis ajoutez la clé publique à votre porte-clés. Votre porte-clés est un ensemble de clés publiques connues. Après avoir établi l'authenticité de la clé publique, vous pouvez l'utiliser pour vérifier la signature de l'application.

Rubriques

- Installation des outils GPG
- Authentification et importation de la clé publique
- Vérification de la signature du package

Installation des outils GPG

Si votre système d'exploitation est Linux, Unix ou macOS, les outils GPG sont probablement déjà installés. Pour tester si les outils sont installés sur votre système, tapez gpg à partir d'une invite de commande. Si les outils GPG sont installés, une invite de commande GPG s'affiche. Si les outils GPG ne sont pas installés, un message d'erreur indiquant que la commande est introuvable s'affiche. Vous pouvez installer le package GnuPG à partir d'un référentiel.

Pour installer les outils GPG, sélectionnez le système d'exploitation correspondant à votre instance.

Debian-based Linux

Depuis un terminal, exécutez la commande suivante :

```
apt-get install gnupg
```

Red Hat–based Linux

Depuis un terminal, exécutez la commande suivante :

```
yum install gnupg
```

macOS

Depuis un terminal, exécutez la commande suivante :

```
brew install gnupg
```

Authentification et importation de la clé publique

L'étape suivante du processus consiste à authentifier la clé AWSTOE publique et à l'ajouter en tant que clé fiable dans votre GPG trousseau de clés.

Pour authentifier et importer la clé AWSTOE publique

1. Obtenez une copie de notre clé publique GPG en effectuant l'une des actions suivantes :

   - Téléchargez la clé sur

     https://awstoe - **<region>** .s3. **<region>**.amazonaws. com/assets/awstoe.gpg. Par exemple, https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/assets/awstoe.gpg.
   - Copiez la clé à partir du texte suivant et collez-la dans un fichier nommé `awstoe.gpg`. Veillez à inclure tout ce qui suit :

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

     mQENBF8UqwsBCACdiRF2bkZYaFSDPFC+LIkWLwFvtUCRwAHtD8KIwTJ6LVn3fHAU
     GhuK0ZH9mRrqRT2bq/xJjGsnF9VqTj2AJqndGJdDjz75YCZYM+ocZ+r5HSJaeW9i
     S5dykHj7Txti2zHe0G5+W0v7v5bPi2sPHsN7XWQ7+G2AMEPTz8PjxY//I0DvMQns
     Sle3l9hz6wCClz1l9LbBzTyHfSm5ucTXvNe88XX5Gmt37OCDM7vfli0Ctv8WFoLN
     6jbxuA/sV71yIkPm9IYp3+GvaKeT870+sn8/JOOKE/U4sJV1ppbqmuUzDfhrZUaw
     8eW8IN9A1FTIuWiZED/5L83UZuQs1S7s2PjlABEBAAG0GkFXU1RPRSA8YXdzdG9l
     QGFtYXpvbi5jb20+iQE5BBMBCAAjBQJfFKsLAhsDBwsJCAcDAgEGFQgCCQoLBBYC
     AwECHgECF4AACgkQ3r3BVvWuvFJGiwf9EVmrBR77+Qe/DUeXZJYoaFr7If/fVDZl
     6V3TC6p0J0Veme7uXleRUTFOjzbh+7e5sDX19HrnPquzCnzfMiqbp4lSoeUuNdOf
     FcpuTCQH+M+sIEIgPno4PLl0Uj2uE1o++mxmonBl/Krk+hly8hB2L/9n/vW3L7BN
     OMb1Ll9PmgGPbWipcT8KRdz4SUex9TXGYzjlWb3jU3uXetdaQY1M3kVKE1siRsRN
     YYDtpcjmwbhjpu4xm19aFqNoAHCDctEsXJA/mkU3erwIRocPyjAZE2dnlkL9ZkFZ
     z9DQkcIarbCnybDM5lemBbdhXJ6hezJE/b17VA0t1fY04MoEkn6oJg==
     =oyze
     -----END PGP PUBLIC KEY BLOCK-----
     ```

2. À l'invite de commande du répertoire dans lequel vous avez enregistré awstoe.gpg, utilisez la commande suivante pour importer la clé AWSTOE publique dans votre trousseau de clés.

```
gpg --import awstoe.gpg
```

La commande renvoie des résultats semblables à ce qui suit :

```
gpg: key F5AEBC52: public key "AWSTOE <awstoe@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

Notez la valeur de la clé ; vous en aurez besoin lors de l'étape suivante. Dans l'exemple précédent, la valeur de la clé est F5AEBC52.

3. Vérifiez l'empreinte en exécutant la commande suivante, en remplaçant key-value (valeur clé) par la valeur de l'étape précédente :

```
gpg --fingerprint key-value
```

Cette commande renvoie un résultat semblable à ce qui suit :

```
pub   2048R/F5AEBC52 2020-07-19
      Key fingerprint = F6DD E01C 869F D639 15E5  5742 DEBD C156 F5AE BC52
uid       [ unknown] AWSTOE <awstoe@amazon.com>
```

De plus, la chaîne de l'empreinte doit être identique à F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52, comme illustré dans l'exemple précédent. Comparez l'empreinte de la clé renvoyée à celle publiée sur cette page. Elles doivent correspondre. S'ils ne correspondent pas, n'installez pas le script AWSTOE d'installation et contactez Support.

Vérification de la signature du package

Après avoir installé les outils GPG, authentifié et importé la clé publique de AWSTOE et vérifié que la clé publique est de confiance, vous êtes prêt à vérifier la signature du script d'installation.

Pour vérifier la signature du script d'installation

1. À l'invite de commande, exécutez la commande suivante pour télécharger le fichier binaire de l'application :

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe
```

Par exemple :

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe
```

Les valeurs prises en charge pour **`architecture`** peuvent être `amd64` `386`, et `arm64`.

2. À l'invite de commande, exécutez la commande suivante pour télécharger le fichier de signature du binaire d'application correspondant à partir du même chemin de préfixe de clé S3 :

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe.sig
```

Par exemple :

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe.sig
```

Les valeurs prises en charge pour **`architecture`** peuvent être `amd64` `386`, et `arm64`.

3. Vérifiez la signature en exécutant la commande suivante à l'invite de commande dans le répertoire dans lequel vous avez enregistré `awstoe.sig` et dans le fichier AWSTOE d'installation. Ces deux fichiers doivent être présents.

```
gpg --verify ./awstoe.sig ~/awstoe
```

Le résultat doit ressembler à ce qui suit :

```
gpg: Signature made Mon 20 Jul 2020 08:54:55 AM IST using RSA key ID F5AEBC52
gpg: Good signature from "AWSTOE awstoe@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: F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52
```

Si le résultat contient l'expression `Good signature from "AWSTOE <awstoe@amazon.com>"`, cela signifie que la signature a été vérifiée et vous pouvez continuer à exécuter le script d'installation d' AWSTOE .

Si le résultat inclut l'expression `BAD signature`, vérifiez si vous avez effectué la procédure correctement. Si cette réponse persiste, n'exécutez pas le fichier d'installation que vous avez téléchargé précédemment et contactez Support.

Voici les informations détaillées sur les avertissements qui peuvent s'afficher :

- AVERTISSEMENT : Cette clé n'est pas certifiée par une signature fiable ! Rien n'indique que la signature appartient au propriétaire. Idéalement, vous devriez vous rendre dans un AWS bureau et recevoir la clé en personne. Cependant, vous le téléchargeriez probablement à partir d'un site Web. Dans ce cas, le site Web est un AWS site Web.
- gpg: no ultimately trusted keys found. Cela signifie que la clé en question n'est pas « définitivement approuvée » par vous ou par d'autres personnes en qui vous avez confiance.

Pour plus d'informations, consultez http://www.gnupg.org.

Vérifiez la signature du téléchargement de l' AWSTOE installation sous Windows

Cette rubrique décrit le processus recommandé pour vérifier la validité du fichier d'installation de l' AWS Task Orchestrator and Executor application sur les systèmes d'exploitation Windows.

Lorsque vous téléchargez une application à partir d'internet, nous vous recommandons d'authentifier l'identité de l'éditeur du logiciel et de vérifier que l'application n'a pas été modifiée ou corrompue depuis sa publication. Cela vous évitera d'installer une version de l'application contenant un virus ou tout autre code malveillant.

Si, après avoir exécuté les étapes décrites dans cette rubrique, vous déterminez que le logiciel de l' AWSTOE application est modifié ou endommagé, n'exécutez pas le fichier d'installation. Au lieu de cela, contactez Support.

Pour vérifier la validité du fichier binaire awstoe téléchargé sur les systèmes d'exploitation Windows, assurez-vous que l'empreinte numérique du certificat de signature Amazon Services LLC est égale à cette valeur :

9D CA 72 17 DA FF B8 2F E4 C4 67 77 36 2F A4 AA C9 08 82 15

> **Note**
>
> Pendant la période de déploiement d'un nouveau binaire, il est possible que votre certificat de signataire ne corresponde pas à la nouvelle empreinte numérique. Si votre certificat de signataire ne correspond pas, vérifiez que la valeur de l'empreinte numérique est la suivante : BA 81 25 EE AC 64 2E A9 F3 C5 93 CA 6D 3E B7 93 7D 68 75 74

Pour vérifier cette valeur, exécutez la procédure suivante :

1. Cliquez avec le bouton droit sur le fichier `awstoe.exe` téléchargé et ouvrez la fenêtre Properties (Propriétés).
2. Choisissez l'onglet Signatures numériques.
3. Dans Signature List, choisissez Amazon Services LLC, puis Details.
4. Choisissez l'onglet General (Général), s'il n'est pas déjà sélectionné, puis View Certificate (Afficher le certificat).
5. Sélectionnez l'onglet Détails, puis sélectionnez All (Tous) dans la liste déroulante Show (Afficher), si cette option n'est pas déjà sélectionnée.
6. Faites défiler l'écran vers le bas jusqu'au champ Thumbprint (Empreinte), puis choisissez Thumbprint (Empreinte). Cela affichera la valeur complète de l'empreinte dans la fenêtre inférieure.

   - Si la valeur de l'empreinte affichée dans la fenêtre inférieure est identique à la valeur suivante :

     9D CA 72 17 DA FF B8 2F E4 C4 67 77 36 2F A4 AA C9 08 82 15

     alors votre AWSTOE binaire téléchargé est authentique et peut être installé en toute sécurité.
   - Si la valeur de l'empreinte numérique dans la fenêtre de détails inférieure n'est pas identique à la valeur précédente, ne l'exécutez pas. `awstoe.exe`

Étapes de démarrage

- Étape 1 : Installation AWSTOE
- Étape 2 : définir les AWS informations d'identification
- Étape 3 : développer les documents des composants localement
- Étape 4 : Valider AWSTOE les composants

- Étape 5 : Exécuter AWSTOE les composants

Étape 1 : Installation AWSTOE

Pour développer des composants localement, téléchargez et installez l' AWSTOE application.

1. Téléchargez l' AWSTOE application

   Pour procéder à l'installation AWSTOE, choisissez le lien de téléchargement correspondant à votre architecture et à votre plate-forme. Pour obtenir la liste complète des liens de téléchargement des applications, voir AWSTOE downloads

   > **Important**
   >
   > AWS supprime progressivement le support pour les versions 1.0 et 1.1 du protocole TLS. Pour accéder au compartiment S3 à des fins de AWSTOE téléchargement, votre logiciel client doit utiliser la version 1.2 ou ultérieure du protocole TLS. Pour plus d'informations, consultez ce billet AWS de blog sur la sécurité.

2. Vérifiez la signature

   Les étapes de vérification de votre téléchargement dépendent de la plate-forme du serveur sur lequel vous exécutez l' AWSTOE application après l'avoir installée. Pour vérifier votre téléchargement sur un serveur Linux, consultezVérifiez la signature sous Linux ou macOS. Pour vérifier votre téléchargement sur un serveur Windows, consultezVérifiez la signature sous Windows.

> **Note**
>
> AWSTOE est invoqué directement depuis son emplacement de téléchargement. Il n'est pas nécessaire d'effectuer une étape d'installation séparée. Cela signifie également que cela AWSTOE peut apporter des modifications à l'environnement local.
> Pour vous assurer d'isoler les modifications lors du développement des composants, nous vous recommandons d'utiliser une EC2 instance pour développer et tester AWSTOE les composants.

Étape 2 : définir les AWS informations d'identification

AWSTOE nécessite des AWS informations d'identification pour se connecter à d'autres sites Services AWS, tels qu'Amazon S3 et Amazon CloudWatch, lors de l'exécution de tâches telles que :

- Téléchargement de AWSTOE documents depuis un chemin Amazon S3 fourni par l'utilisateur.
- Modules `S3Download` d'exécution ou `S3Upload` d'action.
- Le streaming se connecte à CloudWatch, lorsqu'il est activé.

Si vous exécutez AWSTOE sur une EC2 instance, l'exécution AWSTOE utilise les mêmes autorisations que le rôle IAM attaché à l' EC2 instance.

Pour plus d'informations sur les rôles IAM pour EC2, consultez la section Rôles IAM pour Amazon. EC2

Les exemples suivants montrent comment définir les AWS informations d'identification à l'aide des variables d'`AWS_SECRET_ACCESS_KEY`environnement `AWS_ACCESS_KEY_ID` et.

Pour définir ces variables sous Linux, macOS ou Unix, utilisez`export`.

```
export AWS_ACCESS_KEY_ID=your_access_key_id
```

```
export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Pour définir ces variables sous Windows à l'aide de PowerShell, utilisez`$env`.

```
$env:AWS_ACCESS_KEY_ID=your_access_key_id
```

```
$env:AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Pour définir ces variables sous Windows à l'aide de l'invite de commande, utilisez`set`.

```
set AWS_ACCESS_KEY_ID=your_access_key_id
```

```
set AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Étape 3 : développer les documents des composants localement

Les composants sont créés à partir de documents YAML en texte brut. Pour plus d'informations sur la syntaxe des documents, consultezUtilisez la commande AWSTOE cadre de documentation sur les composants pour les composants personnalisés.

Vous trouverez ci-dessous des exemples de documents relatifs aux composants Hello World pour vous aider à démarrer.

Linux

Certains des exemples de composants Linux présentés dans ce guide font référence à un fichier de document de composant nomméhello-world-linux.yml. Vous pouvez utiliser le document suivant pour commencer à utiliser ces exemples.

```
name: Hello World
description: This is hello world testing document for Linux.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the test phase.'
```

Windows

Certains des exemples de composants Windows présentés dans ce guide font référence à un fichier de document de composant nomméhello-world-windows.yml. Vous pouvez utiliser le document suivant pour commencer à utiliser ces exemples.

```
name: Hello World
description: This is Hello World testing document for Windows.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the test phase.'
```

macOS

Certains des exemples de composants macOS présentés dans ce guide font référence à un fichier de document de composant nomméhello-world-macos.yml. Vous pouvez utiliser le document suivant pour commencer à utiliser ces exemples.

```
name: Hello World
description: This is hello world testing document for macOS.
schemaVersion: 1.0
phases:
  - name: build
```

```
  steps:
    - name: HelloWorldStep
      action: ExecuteBash
      inputs:
        commands:
          - echo 'Hello World from the build phase.'
- name: validate
  steps:
    - name: HelloWorldStep
      action: ExecuteBash
      inputs:
        commands:
          - echo 'Hello World from the validate phase.'
- name: test
  steps:
    - name: HelloWorldStep
      action: ExecuteBash
      inputs:
        commands:
          - echo 'Hello World from the test phase.'
```

Étape 4 : Valider AWSTOE les composants

Vous pouvez valider la syntaxe des AWSTOE composants localement avec l' AWSTOE application. Les exemples suivants montrent la `validate` commande AWSTOE d'application permettant de valider la syntaxe d'un composant sans l'exécuter.

> **Note**
>
> L' AWSTOE application ne peut valider que la syntaxe des composants pour le système d'exploitation actuel. Par exemple, lors de l'exécution `awstoe.exe` sous Windows, vous ne pouvez pas valider la syntaxe d'un document Linux qui utilise le module `ExecuteBash` d'action.

Linux ou macOS

```
awstoe validate --documents /home/user/hello-world.yml
```

Windows

```
awstoe.exe validate --documents C:\Users\user\Documents\hello-world.yml
```

Étape 5 : Exécuter AWSTOE les composants

L' AWSTOE application peut exécuter une ou plusieurs phases de documents spécifiés à l'aide de l'argument de ligne de `--phases` commande. Les valeurs prises en charge pour `--phases` sont `buildvalidate`, et`test`. Plusieurs valeurs de phase peuvent être saisies sous forme de valeurs séparées par des virgules.

Lorsque vous fournissez une liste de phases, l' AWSTOE application exécute séquentiellement les phases spécifiées pour chaque document. Par exemple, AWSTOE exécute les `validate` phases `build` et de`document1.yaml`, puis les `validate` phases `build` et de`document2.yaml`.

Pour garantir que vos journaux sont stockés en toute sécurité et conservés à des fins de dépannage, nous vous recommandons de configurer le stockage des journaux dans Amazon S3. Dans Image Builder, l'emplacement Amazon S3 pour la publication des journaux est spécifié dans la configuration de l'infrastructure. Pour plus d'informations sur la configuration de l'infrastructure, voir Gérer la configuration de l'infrastructure d'Image Builder

Si aucune liste de phases n'est fournie, l' AWSTOE application exécute toutes les phases dans l'ordre indiqué dans le document YAML.

Pour exécuter des phases spécifiques dans un ou plusieurs documents, utilisez les commandes suivantes.

Monophasé

```
awstoe run --documents hello-world.yml --phases build
```

Phases multiples

```
awstoe run --documents hello-world.yml --phases build,test
```

Document exécuté

Exécuter toutes les phases dans un seul document

```
awstoe run --documents documentName.yaml
```

Exécuter toutes les phases dans plusieurs documents

```
awstoe run --documents documentName1.yaml,documentName2.yaml
```

Entrez les informations Amazon S3 pour télécharger les AWSTOE journaux à partir d'un chemin local défini par l'utilisateur (recommandé)

```
awstoe run --documents documentName.yaml --log-s3-bucket-name amzn-s3-demo-destination-bucket --log-s3-key-prefix S3KeyPrefix --log-s3-bucket-owner S3BucketOwner --log-directory local_path
```

Exécutez toutes les phases dans un seul document et affichez tous les journaux sur la console

```
awstoe run --documents documentName.yaml --trace
```

Exemple de commande

```
awstoe run --documents s3://bucket/key/doc.yaml --phases build,validate
```

Exécuter un document avec un identifiant unique

```
awstoe run --documents documentName.yaml --execution-id user-provided-id --phases build,test
```

Obtenez de l'aide pour AWSTOE

```
awstoe --help
```

Utilisez la commande AWSTOE cadre de documentation sur les composants pour les composants personnalisés

Pour créer un composant à l'aide de la structure de composants AWS Task Orchestrator and Executor (AWSTOE), vous devez fournir un YAML-based document qui représente les phases et les étapes applicables au composant que vous créez. Services AWS utilisent votre composant lorsqu'ils créent une nouvelle image de machine Amazon (AMI) ou une nouvelle image de conteneur.

Rubriques

- Flux de travail documentaire sur les composants
- Journalisation des composants
- Chainage des entrées et des sorties

- Schéma du document et définitions
- Exemples de documents
- Utiliser des variables dans votre document de composant personnalisé
- Utiliser des constructions conditionnelles dans AWSTOE
- Utilisez les opérateurs de comparaison dans AWSTOE documents relatifs aux composants
- Utiliser des opérateurs logiques dans les documents relatifs aux AWSTOE composants
- Utiliser des constructions en boucle dans AWSTOE

Flux de travail documentaire sur les composants

Le document du AWSTOE composant utilise des phases et des étapes pour regrouper les tâches associées et organiser ces tâches dans un flux de travail logique pour le composant.

> **Tip**
>
> Le service qui utilise votre composant pour créer une image peut implémenter des règles concernant les phases à utiliser pour son processus de génération et le moment où ces phases sont autorisées à s'exécuter. Il est important d'en tenir compte lors de la conception de votre composant.

Phases

Les phases représentent la progression de votre flux de travail tout au long du processus de création de l'image. Par exemple, le service Image Builder utilise `build` et met en `validate` phase les images qu'il produit au cours de sa phase de création. Il utilise les `container-host-test` phases `test` et au cours de sa phase de test pour s'assurer que l'instantané d'image ou l'image du conteneur produit les résultats attendus avant de créer l'AMI finale ou de distribuer l'image du conteneur.

Lorsque le composant s'exécute, les commandes associées à chaque phase sont appliquées dans l'ordre dans lequel elles apparaissent dans le document du composant.

Règles pour les phases

- Chaque nom de phase doit être unique dans un document.
- Vous pouvez définir de nombreuses phases dans votre document.

- Vous devez inclure au moins l'une des phases suivantes dans votre document :
  - build — pour Image Builder, cette phase est généralement utilisée pendant la phase de construction.
  - valider — pour Image Builder, cette phase est généralement utilisée pendant la phase de construction.
  - test — pour Image Builder, cette phase est généralement utilisée pendant la phase de test.
- Les phases s'exécutent toujours dans l'ordre dans lequel elles sont définies dans le document. L'ordre dans lequel elles sont spécifiées pour AWSTOE les commandes du n' AWS CLI a aucun effet.

Étapes

Les étapes sont des unités de travail individuelles qui définissent le flux de travail au sein de chaque phase. Les étapes sont exécutées par ordre séquentiel. Cependant, l'entrée ou la sortie d'une étape peuvent également alimenter une étape suivante en tant qu'entrée. C'est ce qu'on appelle le « chaînage ».

Règles relatives aux étapes

- Le nom de l'étape doit être unique pour la phase.
- L'étape doit utiliser une action prise en charge (module d'action) qui renvoie un code de sortie.

  Pour obtenir la liste complète des modules d'action pris en charge, leur fonctionnement, leurs input/output valeurs et des exemples, voirModules d'action pris en charge par AWSTOE gestionnaire de composants.

Journalisation des composants

AWSTOE crée un nouveau dossier journal sur les instances EC2 utilisées pour créer et tester une nouvelle image, chaque fois que votre composant s'exécute. Pour les images de conteneur, le dossier journal est stocké dans le conteneur.

Pour faciliter le dépannage en cas de problème lors du processus de création de l'image, le document d'entrée et tous les fichiers de sortie AWSTOE créés lors de l'exécution du composant sont stockés dans le dossier journal.

Le nom du dossier journal comprend les éléments suivants :

1. Répertoire des journaux : lorsqu'un service exécute un AWSTOE composant, il est transmis dans le répertoire des journaux, ainsi que les autres paramètres de la commande. Dans les exemples suivants, nous montrons le format de fichier journal utilisé par Image Builder.
   - Linux et macOS : `/var/lib/amazon/toe/`
   - Windows : `$env:ProgramFiles\Amazon\TaskOrchestratorAndExecutor\`
2. Préfixe de fichier — Il s'agit d'un préfixe standard utilisé pour tous les composants : « »`TOE_`.
3. Durée d'exécution — Il s'agit d'un horodatage au format YYYY-MM-DD _ HH-MM-SS _UTC-0 .
4. ID d'exécution : il s'agit du GUID attribué lors de l' AWSTOE exécution d'un ou de plusieurs composants.

Exemple : *`/var/lib/amazon/toe/`*`TOE_`*`2021-07-01_12-34-56_UTC-0_a1bcd2e3-45f6-789a-bcde-0fa1b2c3def4`*

AWSTOE stocke les fichiers principaux suivants dans le dossier journal :

Fichiers d'entrée

- document.yaml — Document utilisé comme entrée pour la commande. Une fois le composant exécuté, ce fichier est stocké sous forme d'artefact.

Fichiers de sortie

- application.log — Le journal de l'application contient des informations horodatées AWSTOE sur le niveau de débogage indiquant ce qui se passe pendant l'exécution du composant.
- detailedoutput.json — Ce fichier JSON contient des informations détaillées sur l'état d'exécution, les entrées, les sorties et les échecs pour tous les documents, phases et étapes applicables au composant lors de son exécution.
- console.log — Le journal de la console contient toutes les informations de sortie standard (stdout) et d'erreur standard (stderr) écrites sur la console pendant l'exécution du composant. AWSTOE
- chaining.json — Ce fichier JSON représente les optimisations AWSTOE appliquées pour résoudre les expressions de chaînage.

> **Note**
>
> Le dossier journal peut également contenir d'autres fichiers temporaires qui ne sont pas abordés ici.

Chainage des entrées et des sorties

L'application AWSTOE de gestion de configuration fournit une fonctionnalité permettant de chaîner les entrées et les sorties en écrivant des références dans les formats suivants :

```
{{ phase_name.step_name.inputs/outputs.variable }}
```

or

```
{{ phase_name.step_name.inputs/outputs[index].variable }}
```

La fonction de chaînage vous permet de recycler le code et d'améliorer la maintenabilité du document.

Règles pour le chaînage

- Les expressions de chaînage ne peuvent être utilisées que dans la section des entrées de chaque étape.
- Les instructions contenant des expressions de chaînage doivent être placées entre guillemets. Par exemple :
  - Expression non valide : `echo {{ phase.step.inputs.variable }}`
  - Expression valide : `"echo {{ phase.step.inputs.variable }}"`
  - Expression valide : `'echo {{ phase.step.inputs.variable }}'`
- Les expressions de chaînage peuvent faire référence à des variables issues d'autres étapes et phases du même document. Cependant, le service d'appel peut avoir des règles qui exigent que le chaînage des expressions ne fonctionne que dans le contexte d'une seule étape. Par exemple, Image Builder ne prend pas en charge le chaînage entre la phase de création et la phase de test, car il exécute chaque étape indépendamment.
- Les index des expressions de chaînage suivent une indexation basée sur zéro. L'indice commence par zéro (0) pour référencer le premier élément.

Exemples

Pour faire référence à la variable source dans la deuxième entrée de l'étape d'exemple suivante, le modèle de chaînage est`{{ build.SampleS3Download.inputs[1].source }}`.

```
phases:
  - name: 'build'
    steps:
      - name: SampleS3Download
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://sample-bucket/sample1.ps1'
            destination: 'C:\sample1.ps1'
          - source: 's3://sample-bucket/sample2.ps1'
            destination: 'C:\sample2.ps1'
```

Pour faire référence à la variable de sortie (égale à « Hello ») de l'étape d'exemple suivante, le modèle de chaînage est`{{ build.SamplePowerShellStep.outputs.stdout }}`.

```
phases:
  - name: 'build'
    steps:
      - name: SamplePowerShellStep
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          commands:
            - 'Write-Host "Hello"'
```

Schéma du document et définitions

Le schéma YAML d'un document est le suivant.

```
name: (optional)
description: (optional)
schemaVersion: "string"

phases:
  - name: "string"
```

```
    steps:
      - name: "string"
        action: "string"
        timeoutSeconds: integer
        onFailure: "Abort|Continue|Ignore"
        maxAttempts: integer
        inputs:
```

Les définitions de schéma d'un document sont les suivantes.

Champ	Description	Type	Obligatoire
name	Nom du document.	String	Non
description	Description du document	String	Non
schemaVersion	Version du schéma du document, actuellement 1.0.	String	Oui
phases	Une liste des phases avec leurs étapes.	List	Oui

Le `schemaVersion` champ accepte `"1.0"` ou `"1"` (qui se normalise à`"1.0"`). AWSTOE ne prend pas en charge les autres versions de schéma.

Vous pouvez définir le `disableLogging` champ facultatif au `true` niveau du document, de la phase ou de l'étape. Ce paramètre supprime les résultats de journalisation pour cette étendue. Utilisez ce paramètre pour les étapes qui traitent des données sensibles.

Les définitions du schéma d'une phase sont les suivantes.

Champ	Description	Type	Obligatoire
name	Nom de la phase. Les noms de phase doivent être	String	Oui

Champ	Description	Type	Obligatoire
	conformes aux règles suivantes : • Les noms de phase ne doivent contenir que des lettres, des chiffres, des traits d'union et des traits de soulignement. • Les noms de phase doivent comporter entre 3 et 128 caractères. • Les noms de phase doivent être uniques dans un document. • Lorsqu'elle est utilisée avec EC2 Image Builder, au moins une phase doit être `build` nommée`validate`, `test` ou.		
steps	Liste des étapes de la phase.	List	Oui

Les définitions du schéma d'une étape sont les suivantes.

Champ	Description	Type	Obligatoire	Valeur par défaut
name	User-defined nom de l'étape.	String		

Champ	Description	Type	Obligatoire	Valeur par défaut
	Les noms des étapes doivent être conformes aux règles suivantes : • Les noms des étapes ne doivent contenir que des lettres, des chiffres, des traits d'union et des traits de soulignement. • Les noms des étapes doivent comporter entre 3 et 128 caractères. • Les noms des étapes doivent être uniques au sein d'une phase.			
action	Mot-clé relatif au module qui exécute l'étape.	String		

Champ	Description	Type	Obligatoire	Valeur par défaut
timeoutSeconds	Nombre de secondes pendant lesquelles l'étape s'exécute avant d'échouer ou de réessayer. Supporte également la valeur -1, ce qui indique un délai d'expirat ion infini. 0 et les autres valeurs négatives ne sont pas autorisées.	Entier	Non	7 200 secondes (120 minutes)

Champ	Description	Type	Obligatoire	Valeur par défaut
onFailure	Spécifie ce que l'étape doit faire en cas d'échec. Les valeurs valides sont les suivantes : • Abandonner — Échoue l'étape après le nombre maximum de tentatives et arrête l'exécution. Définit le statut de la phase et du document à`Failed`. • Continuer — Échoue l'étape après le nombre maximum de tentatives et continue d'exécuter les étapes restantes. Définit le statut de la phase et du document à`Failed`.	String	Non	Interruption

Champ	Description	Type	Obligatoire	Valeur par défaut
	• Ignorer : définit l'étape `IgnoredFailure` après le nombre maximum de tentatives infructueuses et continue à exécuter les étapes restantes. Définit le statut de la phase et du document à`SuccessWithIgnoredFailure`.			
maxAttempts	Nombre maximum de tentatives autorisées avant d'échouer à l'étape.	Entier	Non	1
inputs	Contient les paramètres requis par le module d'action pour exécuter l'étape.	Dict	Oui	

Exemples de documents

Les exemples suivants présentent les documents des composants AWSTOE qui exécutent des tâches pour le système d'exploitation cible.

Linux

Exemple 1 : Exécuter un fichier binaire personnalisé

Voici un exemple de document qui télécharge et exécute un fichier binaire personnalisé sur une instance Linux.

```
name: LinuxBin
description: Download and run a custom Linux binary file.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://<replaceable>amzn-s3-demo-source-bucket</replaceable>/
<replaceable>myapplication</replaceable>
            destination: /tmp/<replaceable>myapplication</replaceable>
      - name: Enable
        action: ExecuteBash
        onFailure: Continue
        inputs:
          commands:
            - 'chmod u+x {{ build.Download.inputs[0].destination }}'
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '--install'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Windows

Exemple 1 : Installation des mises à jour Windows

Voici un exemple de document qui installe toutes les mises à jour Windows disponibles, exécute un script de configuration, valide les modifications avant la création de l'AMI et teste les modifications après la création de l'AMI.

```
name: RunConfig_UpdateWindows
description: 'This document will install all available Windows updates and run a
 config script. It will then validate the changes before an AMI is created. Then
 after AMI creation, it will test all the changes.'
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: DownloadConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/config.ps1'
            destination: 'C:\config.ps1'

      - name: RunConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          file: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: Cleanup
        action: DeleteFile
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - path: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: RebootAfterConfigApplied
        action: Reboot
        inputs:
```

```
        delaySeconds: 60

    - name: InstallWindowsUpdates
      action: UpdateOS

- name: validate
  steps:
    - name: DownloadTestConfigScript
      action: S3Download
      timeoutSeconds: 60
      onFailure: Abort
      maxAttempts: 3
      inputs:
        - source: 's3://customer-bucket/testConfig.ps1'
          destination: 'C:\testConfig.ps1'

    - name: ValidateConfigScript
      action: ExecutePowerShell
      timeoutSeconds: 120
      onFailure: Abort
      maxAttempts: 3
      inputs:
        file: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

    - name: Cleanup
      action: DeleteFile
      onFailure: Abort
      maxAttempts: 3
      inputs:
        - path: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

- name: test
  steps:
    - name: DownloadTestConfigScript
      action: S3Download
      timeoutSeconds: 60
      onFailure: Abort
      maxAttempts: 3
      inputs:
        - source: 's3://customer-bucket/testConfig.ps1'
          destination: 'C:\testConfig.ps1'

    - name: ValidateConfigScript
      action: ExecutePowerShell
```

```
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          file: '{{test.DownloadTestConfigScript.inputs[0].destination}}'
```

Exemple 2 : installer le AWS CLI sur une instance Windows

Voici un exemple de document qui installe le AWS CLI sur une instance Windows à l'aide du fichier d'installation.

```
name: InstallCLISetUp
description: Install &CLI; using the setup file
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLISetup.exe
            destination: C:\Windows\temp\AWSCLISetup.exe
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '/install'
            - '/quiet'
            - '/norestart'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Exemple 3 : installer le AWS CLI avec le programme d'installation MSI

Voici un exemple de document qui installe le AWS CLI avec le programme d'installation MSI.

```
name: InstallCLIMSI
description: Install &CLI; using the MSI installer
```

```
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLI64PY3.msi
            destination: C:\Windows\temp\AWSCLI64PY3.msi
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: 'C:\Windows\System32\msiexec.exe'
          arguments:
            - '/i'
            - '{{ build.Download.inputs[0].destination }}'
            - '/quiet'
            - '/norestart'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

macOS

Exemple 1 : Exécuter un fichier binaire macOS personnalisé

Voici un exemple de document qui télécharge et exécute un fichier binaire personnalisé sur une instance de macOS.

```
name: macOSBin
description: Download and run a binary file on macOS.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://<replaceable>amzn-s3-demo-source-bucket</replaceable>/
<replaceable>myapplication</replaceable>
            destination: /tmp/<replaceable>myapplication</replaceable>
```

```
      - name: Enable
        action: ExecuteBash
        onFailure: Continue
        inputs:
          commands:
            - 'chmod u+x {{ build.Download.inputs[0].destination }}'
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '--install'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Utiliser des variables dans votre document de composant personnalisé

Les variables permettent d'étiqueter les données avec des noms significatifs qui peuvent être utilisés dans l'ensemble d'une application. Vous pouvez définir des variables personnalisées avec des formats simples et lisibles pour les flux de travail complexes, et les référencer dans le document du composant d'application YAML correspondant à un AWSTOE composant.

Cette section fournit des informations pour vous aider à définir les variables de votre AWSTOE composant dans le document du composant d'application YAML, notamment la syntaxe, les contraintes de nom et des exemples.

Constantes

Les constantes sont des variables immuables qui ne peuvent pas être modifiées ou remplacées une fois définies. Les constantes peuvent être définies à l'aide des valeurs figurant dans la `constants` section d'un AWSTOE document.

Règles pour les noms de constantes

- Le nom doit comporter entre 3 et 128 caractères.
- Le nom ne peut contenir que des caractères alphanumériques (a-z, A-Z, 0-9), des tirets (-) ou des traits de soulignement (_).

- Le nom doit être unique dans le document.
- Le nom doit être spécifié sous forme de chaîne YAML.

Syntaxe

```
constants:
  - <name>:
      type: <constant type>
      value: <constant value>
```

Nom de la touche	Obligatoire	Description
`name`	Oui	Nom de la constante. Doit être unique pour le document (il ne doit pas être identique aux autres noms de paramètres ou constantes).
`value`	Oui	Valeur de la constante.
`type`	Oui	Type de constante. Le type pris en charge est`string`.

Valeurs constantes de référence dans un document

Vous pouvez référencer des constantes dans les entrées d'étape ou de boucle à l'intérieur de votre document YAML, comme suit :

- Les références constantes distinguent les majuscules et minuscules, et le nom doit correspondre exactement.
- Le nom doit être placé entre accolades `{{ MyConstant }}` doubles.
- Les espaces sont autorisés à l'intérieur des bretelles et sont automatiquement découpés. Par exemple, toutes les références suivantes sont valides :

  `{{ MyConstant }}`, `{{ MyConstant}}`, `{{MyConstant }}`, `{{MyConstant}}`
- La référence dans le document YAML doit être spécifiée sous forme de chaîne (entre guillemets simples ou doubles).

Par exemple : n'- {{ *MyConstant* }}est pas valide, car il n'est pas identifié comme une chaîne.

Toutefois, les références suivantes sont toutes deux valides : - '{{ *MyConstant* }}' et- "{{ *MyConstant* }}".

Exemples

Constante référencée dans les entrées d'étape

```
name: Download AWS CLI version 2
schemaVersion: 1.0
constants:
  - Source:
      type: string
      value: https://awscli.amazonaws.com/AWSCLIV2.msi
phases:
  - name: build
    steps:
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

Constante référencée dans les entrées de boucle

```
name: PingHosts
schemaVersion: 1.0
constants:
  - Hosts:
      type: string
      value: 127.0.0.1,amazon.com
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
            list: '{{ Hosts }}'
            delimiter: ','
```

```
        inputs:
          commands:
            - ping -c 4 {{ loop.value }}
```

Parameters

Les paramètres sont des variables mutables, avec des paramètres que l'application appelante peut fournir au moment de l'exécution. Vous pouvez définir des paramètres dans la `Parameters` section du document YAML.

Règles relatives aux noms de paramètres

- Le nom doit comporter entre 3 et 128 caractères.
- Le nom ne peut contenir que des caractères alphanumériques (a-z, A-Z, 0-9), des tirets (-) ou des traits de soulignement (_).
- Le nom doit être unique dans le document.
- Le nom doit être spécifié sous forme de chaîne YAML.

Syntaxe

```
parameters:
  - <name>:
      type: <parameter type>
      default: <parameter value>
      description: <parameter description>
```

Nom de la touche	Obligatoire	Description
`name`	Oui	Nom du paramètre. Doit être unique pour le document (il ne doit pas être identique aux autres noms de paramètres ou constantes).
`type`	Oui	Type de données du paramètre. Les types pris en charge incluent :`string`.

Nom de la touche	Obligatoire	Description
`default`	Non	La valeur par défaut du paramètre.
`description`	Non	Décrit le paramètre.

Valeurs des paramètres de référence dans un document

Vous pouvez référencer des paramètres dans des entrées par étapes ou en boucle dans votre document YAML, comme suit :

- Les références aux paramètres distinguent les majuscules et minuscules, et le nom doit correspondre exactement.
- Le nom doit être placé entre accolades `{{ MyParameter }}` doubles.
- Les espaces sont autorisés à l'intérieur des bretelles et sont automatiquement découpés. Par exemple, toutes les références suivantes sont valides :

  `{{ MyParameter }}`, `{{ MyParameter}}`, `{{MyParameter }}`, `{{MyParameter}}`
- La référence dans le document YAML doit être spécifiée sous forme de chaîne (entre guillemets simples ou doubles).

  Par exemple : n'`- {{ MyParameter }}`est pas valide, car il n'est pas identifié comme une chaîne.

  Toutefois, les références suivantes sont toutes deux valides : `- '{{ MyParameter }}'` et `- "{{ MyParameter }}"`.

Exemples

Les exemples suivants montrent comment utiliser des paramètres dans votre document YAML :

- Référez-vous à un paramètre dans les entrées d'étape :

```
name: Download AWS CLI version 2
schemaVersion: 1.0
parameters:
  - Source:
```

```
      type: string
      default: 'https://awscli.amazonaws.com/AWSCLIV2.msi'
      description: The AWS CLI installer source URL.
phases:
  - name: build
    steps:
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

- Référez-vous à un paramètre dans les entrées de boucle :

```
name: PingHosts
schemaVersion: 1.0
parameters:
  - Hosts:
      type: string
      default: 127.0.0.1,amazon.com
      description: A comma separated list of hosts to ping.
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
            list: '{{ Hosts }}'
            delimiter: ','
        inputs:
          commands:
            - ping -c 4 {{ loop.value }}
```

Remplacer les paramètres lors de l'exécution

Vous pouvez utiliser l'`--parameters`option AWS CLI avec une paire clé-valeur pour définir une valeur de paramètre lors de l'exécution.

- <value>Spécifiez la paire clé-valeur du paramètre sous forme de nom et de valeur, séparés par un signe égal (<name>=).
- Les paramètres multiples doivent être séparés par une virgule.

- Les noms de paramètres introuvables dans le document du composant YAML sont ignorés.
- Le nom et la valeur du paramètre sont tous deux obligatoires.

> **Important**
>
> Les paramètres des composants sont des valeurs en texte brut et sont connectés AWS CloudTrail. Nous vous recommandons d'utiliser AWS Secrets Manager le AWS Systems Manager Parameter Store pour stocker vos secrets. Pour plus d'informations sur Secrets Manager, voir Qu'est-ce que Secrets Manager ? dans le guide de AWS Secrets Manager l'utilisateur. Pour plus d'informations sur AWS Systems Manager Parameter Store, voir AWS Systems Manager Parameter Store dans le guide de AWS Systems Manager l'utilisateur.

Syntaxe

```
--parameters name1=value1,name2=value2...
```

Option CLI	Obligatoire	Description
--paramètres *name* =*value*,...	Non	Cette option prend une liste de paires clé-valeur, avec le nom du paramètre comme clé.

Exemples

Les exemples suivants montrent comment utiliser des paramètres dans votre document YAML :

- La paire clé-valeur du paramètre spécifiée dans cette `--parameter` option n'est pas valide :

```
--parameters ntp-server=
```

- Définissez une paire clé-valeur de paramètre avec l'`--parameter`option dans le champ : AWS CLI

```
--parameters ntp-server=ntp-server-windows-qe.us-east1.amazon.com
```

- Définissez plusieurs paires clé-valeur de paramètres à l'aide de l'`--parameter`option suivante : AWS CLI

```
--parameters ntp-server=ntp-server.amazon.com,http-url=https://internal-us-
east1.amazon.com
```

Utiliser les paramètres du magasin de paramètres de Systems Manager

Vous pouvez référencer les AWS Systems Manager paramètres du magasin de paramètres (paramètres SSM) dans les documents de vos composants en préfixant les variables par. `aws:ssm` Par exemple,

`{{ aws:ssm:/my/param }}`se résout à la valeur du paramètre `/my/param` SSM.

Cette fonctionnalité prend en charge les types de paramètres SSM suivants :

- Chaîne — Correspond au type de AWSTOE chaîne.
- StringList — Correspond au AWSTOE `stringList` type.
- SecureString — Correspond au type de AWSTOE chaîne.

Pour plus d'informations sur le magasin de paramètres, voir AWS Systems Manager Parameter Store dans le guide de AWS Systems Manager l'utilisateur.

Vous pouvez également référencer des AWS Secrets Manager secrets à l'aide d'un paramètre `SecureString` SSM. Par exemple : `{{ aws:ssm:/aws/reference/secretsmanager/test/test-secret }}`. Pour plus d'informations, consultez la section Référencement de AWS Secrets Manager secrets à partir des paramètres du Parameter Store.

> **Important**
>
> Image Builder exclut la résolution des `SecureString` paramètres de ses journaux. Cependant, vous devez également vous assurer que les informations sensibles ne sont pas enregistrées par le biais des commandes émises dans le document du composant. Par exemple, si vous utilisez la `echo` commande avec une chaîne sécurisée, la commande écrit une valeur en texte brut dans le journal.

Autorisations IAM requises

Pour utiliser les paramètres de Systems Manager dans vos composants, votre rôle d'instance doit être `ssm:GetParameter` autorisé à accéder à l'ARN de la ressource de paramètres. Par exemple :

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": "ssm:GetParameter",
   "Resource": "arn:aws:ssm:*:111122223333:parameter/ImageBuilder-*"
  }
 ]
}
```

Pour accéder aux valeurs chiffrées, vous devez également disposer des autorisations suivantes :

- Ajoutez `kms:Decrypt` des `SecureString` paramètres ou AWS Secrets Manager des valeurs chiffrés gérés par un client AWS KMS key.
- Ajoutez `secretsmanager:GetSecretValue` si vous faites référence à un secret du Secrets Manager.

Référencer un paramètre SSM dans un document de composant

L'exemple suivant montre comment référencer un paramètre Systems Manager Parameter Store parmi les paramètres Systems Manager d'un composant :

```
name: UseSSMParameterVariable
description: This is a sample component document that prints out the value of an SSM
 Parameter. Never do this for a SecureString parameter.
schemaVersion: 1.0

phases:
  - name: verify
    steps:
      - name: EchoParameterValue
```

```
        action: ExecuteBash
        inputs:
          commands:
            - echo "Log SSM parameter name: /my/test/param, value {{ aws:ssm:/my/test/
param }}."
```

Résolution des variables d'exécution dynamiques pour les paramètres SSM

AWSTOE fournit la fonction intégrée suivante que vous pouvez utiliser dans les références de variables pour manipuler ou transformer des valeurs lors de l'exécution.

fonction de résolution

La `resolve` fonction résout une référence de variable à l'intérieur d'une autre référence de variable, permettant un référencement dynamique des noms de variables. Cela est utile lorsque vous travaillez avec des paramètres SSM où une partie du chemin du paramètre peut être variable et transmise en tant que paramètre de document.

La `resolve` fonction prend uniquement en charge la résolution dynamique de la partie du nom d'un paramètre SSM.

Syntaxe

`dynamic_variable`Dans l'exemple suivant, le nom d'un paramètre SSM doit être l'un des suivants :

- Une référence de paramètre SSM (par exemple,`aws:ssm:`*`/my/param`*)
- Une référence aux paramètres d'un document de composant (par exemple,*`parameter-name`*)

```
{{ aws:ssm:resolve(dynamic_variable) }}
```

Exemple : résolution d'un paramètre SSM lors de l'exécution

L'exemple suivant montre comment utiliser la `resolve` fonction dans un document de composant YAML :

```
name: SsmParameterTest
description: This component verifies an SSM parameter variable reference with the echo
 command.
schemaVersion: 1.0

parameters:
```

```
  - parameter-name:
      type: string
      description: "test"

phases:
  - name: validate
    steps:
      - name: PrintDynamicVariable
        action: ExecuteBash
        inputs:
          commands:
            - echo "{{ aws:ssm:resolve(parameter-name) }}"
```

Utiliser des constructions conditionnelles dans AWSTOE

Les constructions conditionnelles exécutent différentes actions dans le document de votre composant selon que l'expression conditionnelle spécifiée est évaluée à `true` ou. `false` Vous pouvez utiliser la `if` construction pour contrôler le flux d'exécution dans le document de votre composant.

si Construct

Vous pouvez utiliser la `if` construction pour évaluer si une étape doit être exécutée ou non. Par défaut, lorsque l'expression `if` conditionnelle est évaluée à`true`, AWSTOE exécute l'étape, et lorsque la condition est évaluée à`false`, AWSTOE ignore l'étape. Si une étape est ignorée, elle est considérée comme une étape réussie lorsqu' AWSTOE il est déterminé si la phase et le document ont été exécutés correctement.

> **Note**
>
> Une `if` instruction n'est évaluée qu'une seule fois, même si l'étape déclenche un redémarrage. Si une étape redémarre, elle reconnaît que l'`if`instruction a déjà été évaluée et continue là où elle s'est arrêtée.

> **Note**
>
> Les expressions conditionnelles prennent en charge une profondeur d'imbrication maximale de 4 niveaux. Si vous imbriquez des opérateurs logiques (`and`,`or`,`not`) au-delà de 4 niveaux de profondeur, la validation échoue.

Syntaxe

```
if:
  - <conditional expression>:
      [then: <step action>]
      [else: <step action>]
```

Nom de la touche	Obligatoire	Description
expression conditionnelle	Oui	L'expression conditionnelle peut contenir exactement l'un des types d'opérateurs suivants au niveau supérieur. • Opérateur de comparaison : pour obtenir la liste des opérateurs de comparaison et des informations sur leur fonctionnement dans les documents relatifs aux AWSTOE composants, voirOpérateurs de comparaison. • Opérateur logique : les opérateurs logiques incluent `and` `or` `not`, et et opèrent sur un ou plusieurs opérateurs de comparaison. Pour plus d'informations sur le fonctionnement des opérateurs logiques dans les documents relatifs aux AWSTOE composants, consultezOpérateurs logiques.

Nom de la touche	Obligatoire	Description
		Si votre expression doit satisfaire plusieurs condition s, utilisez un opérateur logique pour spécifier vos conditions.
then	Non	Définit l'action à effectuer si l'expression conditionnelle est évaluée à`true`.
else	Non	Définit l'action à effectuer si l'expression conditionnelle est évaluée à`false`.
action par étapes	Conditionnel	Lorsque vous utilisez `then` ou`else`, vous devez spécifier l'une des étapes suivantes : • Abandonner : AWSTOE marque l'étape comme ayant échoué. • Exécuter : AWSTOE exécute l'étape. • Ignorer : permet de AWSTOE sauter l'étape.

Exemple 1 : installer le package

Les exemples d'étapes suivants, tirés d'un document de AWSTOE composant, utilisent des opérateurs logiques pour tester une valeur de paramètre et exécuter les commandes du gestionnaire de packages appropriées pour installer une application si le package est décompressé.

```
  - name: InstallUnzipAptGet
    action: ExecuteBash
    if:
      and:
          - binaryExists: 'apt-get'
          - not:
              binaryExists: 'unzip'
    inputs:
      commands:
          - sudo apt-get update
          - sudo apt-get install -y unzip

  - name: InstallUnzipYum
    action: ExecuteBash
    if:
      and:
          - binaryExists: 'yum'
          - not:
              binaryExists: 'unzip'
    inputs:
      commands:
          - sudo yum install -y unzip

  - name: InstallUnzipZypper
    action: ExecuteBash
    if:
      and:
          - binaryExists: 'zypper'
          - not:
              binaryExists: 'unzip'
    inputs:
      commands:
          - sudo zypper refresh
          - sudo zypper install -y unzip
```

Exemple 2 : sauter une étape

L'exemple suivant montre deux manières de sauter une étape. L'un utilise un opérateur logique et l'autre un opérateur de comparaison avec l'action `Skip` step.

```
# Creates a file if it does not exist using not
- name: CreateMyConfigFile-1
```

```
  action: ExecuteBash
  if:
    not:
      fileExists: '/etc/my_config'
  inputs:
    commands:
      - echo "Hello world" > '/etc/my_config'

# Creates a file if it does not exist using then and else
- name: CreateMyConfigFile-2
  action: ExecuteBash
  if:
    fileExists: '/etc/my_config'
    then: Skip
    else: Execute
  inputs:
    commands:
      - echo "Hello world" > '/etc/my_config'
```

Utilisez les opérateurs de comparaison dans AWSTOE documents relatifs aux composants

Vous pouvez utiliser les opérateurs de comparaison suivants avec le module Affirmer d'action et avec les expressions conditionnelles qui utilisent lesi Construct. Un opérateur de comparaison peut agir sur une seule valeur, par exemple`stringIsEmpty`, ou il peut comparer une valeur de référence à une deuxième valeur (valeur variable) pour déterminer si l'expression conditionnelle est évaluée à `true` ou`false`.

Si la comparaison fonctionne sur deux valeurs, la seconde valeur peut être une variable de chaînage.

Lorsque vous comparez des valeurs d'un type différent, les conversions de valeurs suivantes peuvent avoir lieu avant la comparaison :

- Pour les comparaisons numériques, si la valeur de la variable est une chaîne, AWSTOE convertit la chaîne en nombre avant l'évaluation. Si la conversion n'est pas possible, la comparaison est renvoyée`false`. Par exemple, si la valeur de la variable est`"1.0"`, la conversion fonctionne, mais si c'est la valeur de la variable, `"a10"` la conversion échoue.
- Pour les comparaisons de chaînes, si la valeur de la variable est un nombre, elle est AWSTOE convertie en chaîne avant l'évaluation.

Comparez les chaînes

Les opérateurs de comparaison suivants fonctionnent avec des chaînes pour comparer des valeurs, pour tester la présence d'espaces ou d'une chaîne vide, ou pour comparer une valeur d'entrée à un modèle d'expression régulière. Les comparaisons de chaînes ne font pas la distinction majuscules/majuscules et ne réduisent pas les espaces entre le début et la fin des entrées de chaîne.

Opérateurs de comparaison de chaînes

- stringIsEmpty
- stringIsWhitespace
- stringEquals
- stringLessThan
- stringLessThanEquals
- stringGreaterThan
- stringGreaterThanEquals
- patternMatches

stringIsEmpty

L'`stringIsEmpty`opérateur revient `true` si la chaîne spécifiée ne contient aucun caractère. Par exemple :

```
# Evaluates to true
stringIsEmpty: ""

# Evaluates to false
stringIsEmpty: " "

# Evaluates to false
stringIsEmpty: "Hello."
```

stringIsWhitespace

Teste si la chaîne spécifiée pour ne `stringIsWhitespace` contient que des espaces. Par exemple :

```
# Evaluates to true
```

```
stringIsWhitespace: "   "

# Evaluates to false
stringIsWhitespace: ""

# Evaluates to false
stringIsWhitespace: " Hello?"
```

String est égal

Teste si la chaîne spécifiée pour `stringEquals` correspond exactement à la chaîne spécifiée dans le `value` paramètre. Par exemple :

```
# Evaluates to true
stringEquals: 'Testing, testing...'
value: 'Testing, testing...'

# Evaluates to false
stringEquals: 'Testing, testing...'
value: 'Hello again.'

# Evaluates to false
stringEquals: 'Testing, testing...'
value: 'TESTING, TESTING....'

# Evaluates to false
stringEquals: 'Testing, testing...'
value: '   Testing, testing...'

# Evaluates to false
stringEquals: 'Testing, testing...'
value: 'Testing, testing...   '
```

stringLessThan

Teste si la chaîne spécifiée pour `stringLessThan` est inférieure à la chaîne spécifiée dans le `value` paramètre. Par exemple :

```
# Evaluates to true
# This comparison operator isn't case sensitive
stringlessThan: 'A'
value: 'a'
```

```
# Evaluates to true - 'a' is less than 'b'
stringlessThan: 'b'
value: 'a'

# Evaluates to true
# Numeric strings compare as less than alphabetic strings
stringlessThan: 'a'
value: '0'

# Evaluates to false
stringlessThan: '0'
value: 'a'
```

stringLessThanEquals

Teste si la chaîne spécifiée pour `stringLessThanEquals` est inférieure ou égale à la chaîne spécifiée dans le `value` paramètre. Par exemple :

```
# Evaluates to true - 'a' is equal to 'a'
stringLessThanEquals: 'a'
value: 'a'

# Evaluates to true - since the comparison isn't case sensitive, 'a' is equal to 'A'
stringLessThanEquals: 'A'
value: 'a'

# Evaluates to true - 'a' is less than 'b'
stringLessThanEquals: 'b'
value: 'a'

# Evaluates to true - '0' is less than 'a'
stringLessThanEquals: 'a'
value: '0'

# Evaluates to false - 'a' is greater than '0'
stringLessThanEquals: '0'
value: 'a'
```

stringGreaterThan

Teste si la chaîne spécifiée pour `stringGreaterThan` est supérieure à la chaîne spécifiée dans le `value` paramètre. Par exemple :

```
# Evaluates to false - since the comparison isn't case sensitive, 'A' is equal to
 'a'
stringGreaterThan: 'a'
value: 'A'

# Evaluates to true - 'b' is greater than 'a'
stringGreaterThan: 'a'
value: 'b'

# Evaluates to true - 'a' is greater than '0'
stringGreaterThan: '0'
value: 'a'

# Evaluates to false - '0' is less than 'a'
stringGreaterThan: 'a'
value: '0'
```

stringGreaterThanEquals

Teste si la chaîne spécifiée pour `stringGreaterThanEquals` est supérieure ou égale à la chaîne spécifiée dans le `value` paramètre. Par exemple :

```
# Evaluates to true - 'a' is equal to 'A'
stringGreaterThanEquals: 'A'
value: 'a'

# Evaluates to true - 'b' is greater than 'a'
stringGreaterThanEquals: 'a'
value: 'b'

# Evaluates to true - 'a' is greater than '0'
stringGreaterThanEquals: '0'
value: 'a'

# Evaluates to false - '0' is less than 'a'
stringGreaterThanEquals: 'a'
value: '0'
```

Correspondances à motifs

Teste si la chaîne spécifiée dans le `value` paramètre correspond au modèle d'expression régulière spécifié pour`patternMatches`. La comparaison utilise le package Golang regexp,

qui est conforme à la syntaxe RE2. Pour plus d'informations sur les règles RE2, consultez le référentiel google/ re2 dans. GitHub

L'exemple suivant montre une correspondance de modèle qui renvoie `true` :

```
patternMatches: '^[a-z]+$'
value: 'ThisIsValue'
```

Comparez les chiffres

Les opérateurs de comparaison suivants fonctionnent avec des nombres. Les valeurs fournies pour ces opérateurs doivent être de l'un des types suivants, conformément à la spécification YAML. Support pour les comparaisons numériques utilisant l'opérateur de comparaison de gros paquets Golang, par exemple : func (*Float) Cmp.

- Entier
- Float (basé sur float64, qui prend en charge les nombres de -1,7e+308 à +1,7e+308)
- Une chaîne qui correspond au modèle d'expression régulière suivant : `^[-+]?([0-9]+[.])?[0-9]+$`

Opérateurs de comparaison de nombres

- numberEquals
- numberLessThan
- numberLessThanEquals
- numberGreaterThan
- numberGreaterThanEquals

Le nombre est égal

Teste si le nombre spécifié pour `numberEquals` est égal au nombre spécifié dans le `value` paramètre. Tous les exemples de comparaison suivants renvoient `true` :

```
# Values provided as a positive number
numberEquals: 1
value: 1
```

```
# Comparison value provided as a string
numberEquals: '1'
value: 1

# Value provided as a string
numberEquals: 1
value: '1'

# Values provided as floats
numberEquals: 5.0
value: 5.0

# Values provided as a negative number
numberEquals: -1
value: -1
```

numberLessThan

Teste si le nombre spécifié pour `numberLessThan` est inférieur au nombre spécifié dans le `value` paramètre. Par exemple :

```
# Evaluates to true
numberLessThan: 2
value: 1

# Evaluates to true
numberLessThan: 2
value: 1.9

# Evaluates to false
numberLessThan: 2
value: '2'
```

numberLessThanEquals

Teste si le nombre spécifié pour `numberLessThanEquals` est inférieur ou égal au nombre spécifié dans le `value` paramètre. Par exemple :

```
# Evaluates to true
numberLessThanEquals: 2
value: 1
```

```
# Evaluates to true
numberLessThanEquals: 2
value: 1.9

# Evaluates to true
numberLessThanEquals: 2
value: '2'

# Evaluates to false
numberLessThanEquals: 2
value: 2.1
```

numberGreaterThan

Teste si le nombre spécifié pour `numberGreaterThan` est supérieur au nombre spécifié dans le `value` paramètre. Par exemple :

```
# Evaluates to true
numberGreaterThan: 1
value: 2

# Evaluates to true
numberGreaterThan: 1
value: 1.1

# Evaluates to false
numberGreaterThan: 1
value: '1'
```

numberGreaterThanEquals

Teste si le nombre spécifié pour `numberGreaterThanEquals` est supérieur ou égal au nombre spécifié dans le `value` paramètre. Par exemple :

```
# Evaluates to true
numberGreaterThanEquals: 1
value: 2

# Evaluates to true
numberGreaterThanEquals: 1
value: 1.1

# Evaluates to true
```

```
numberGreaterThanEquals: 1
value: '1'

# Evaluates to false
numberGreaterThanEquals: 1
value: 0.8
```

Vérifier les fichiers

Les opérateurs de comparaison suivants vérifient le hachage du fichier ou vérifient si un fichier ou un dossier existe.

Opérateurs de fichiers et de dossiers

- binaryExists
- fileExists
- folderExists
- fileMD5Equals
- fileSHA1Equals
- fileSHA256Equals
- fileSHA512Equals

Le binaire existe

Teste si une application est disponible dans le chemin actuel. Par exemple :

```
binaryExists: 'foo'
```

> **Note**
>
> Sur les systèmes Linux et macOS, pour une application nommée*foo*, cela fonctionne de la même manière que la commande bash suivante :type ***foo*** >/dev/null 2>&1, où $? == 0 indique une comparaison réussie.
> Sur les systèmes Windows, pour une application nommée*foo*, cela fonctionne de la même manière que la PowerShell commande & C:\Windows\System32\where.exe /Q ***foo*** où $LASTEXITCODE = 0 indique une comparaison réussie.

Le fichier existe

Teste si un fichier existe au chemin spécifié. Vous pouvez fournir un chemin absolu ou relatif. Si l'emplacement que vous spécifiez existe et est un fichier, la comparaison est évaluée à`true`. Par exemple :

```
fileExists: '/path/to/file'
```

> **Note**
>
> Sur les systèmes Linux et macOS, cela fonctionne de la même manière que la commande bash suivante :-d ***/path/to/file***, où $? == 0 indique une comparaison réussie.
> Sur les systèmes Windows, cela fonctionne de la même manière que la PowerShell commandeTest-Path -Path '***C:\path\to\file***' -PathType 'Leaf'.

Le dossier existe

Teste si un dossier existe sur le chemin spécifié. Vous pouvez fournir un chemin absolu ou relatif. Si l'emplacement que vous spécifiez existe et qu'il s'agit d'un dossier, la comparaison est évaluée à`true`. Par exemple :

```
folderExists: '/path/to/folder'
```

> **Note**
>
> Sur les systèmes Linux et macOS, cela fonctionne de la même manière que la commande bash suivante :-d ***/path/to/folder***, où $? == 0 indique une comparaison réussie.
> Sur les systèmes Windows, cela fonctionne de la même manière que la PowerShell commandeTest-Path -Path '***C:\path\to\folder***' -PathType 'Container'.

Le fichier MD5 est égal

Teste si le hachage MD5 d'un fichier est égal à une valeur spécifiée. Par exemple :

```
fileMD5Equals: '<MD5Hash>'
path: '/path/to/file'
```

Le fichier HA1 est égal

Teste si le hachage SHA1 d'un fichier est égal à une valeur spécifiée. Par exemple :

```
fileSHA1Equals: '<SHA1Hash>'
path: '/path/to/file'
```

Le fichier HA256 est égal à

Teste si le hachage SHA256 d'un fichier est égal à une valeur spécifiée. Par exemple :

```
fileSHA256Equals: '<SHA256Hash>'
path: '/path/to/file'
```

Fichier HA512 est égal à

Teste si le hachage SHA512 d'un fichier est égal à une valeur spécifiée. Par exemple :

```
fileSHA512Equals: '<SHA512Hash>'
path: '/path/to/file'
```

Utiliser des opérateurs logiques dans les documents relatifs aux AWSTOE composants

Vous pouvez utiliser les opérateurs logiques suivants pour ajouter ou modifier des expressions conditionnelles dans votre document de composant. AWSTOE évalue les expressions conditionnelles dans l'ordre dans lequel les conditions sont spécifiées. Pour plus d'informations sur les opérateurs de comparaison pour les documents de composants, consultezUtilisez les opérateurs de comparaison dans AWSTOE documents relatifs aux composants.

et

Avec l'andopérateur, vous pouvez évaluer deux ou plusieurs comparaisons sous la forme d'une seule expression. L'expression est évaluée `true` lorsque toutes les conditions de la liste sont vraies. Dans le cas contraire, l'expression est évaluée à. `false`

Exemples :

L'exemple suivant effectue deux comparaisons : une chaîne et un nombre. Les deux comparaisons étant vraies, l'expression est évaluée comme vraie.

```
and:
  - stringEquals: 'test_string'
    value: 'test_string'
  - numberEquals: 1
    value: 1
```

L'exemple suivant effectue également deux comparaisons. La première comparaison est fausse, c'est à ce moment que l'évaluation s'arrête et que la seconde comparaison est ignorée. L'expression est évaluée à. `false`

```
and:
  - stringEquals: 'test_string'
    value: 'Hello world!'
  - numberEquals: 1
    value: 1
```

ou

Avec l'`or`opérateur, vous pouvez évaluer deux ou plusieurs comparaisons sous la forme d'une seule expression. L'expression indique `true` si l'une des comparaisons spécifiées est vraie. Si aucune des comparaisons spécifiées n'est évaluée à`true`, l'expression est évaluée à`false`.

Exemples :

L'exemple suivant effectue deux comparaisons : une chaîne et un nombre. La première comparaison étant vraie, l'expression est évaluée à `true` et la seconde comparaison est ignorée.

```
or:
  - stringEquals: 'test_string'
    value: 'test_string'
  - numberEquals: 1
    value: 3
```

L'exemple suivant effectue également deux comparaisons. La première comparaison est fausse, et l'évaluation se poursuit. La deuxième comparaison étant vraie, l'expression est évaluée à`true`.

```
or:
  - stringEquals: 'test_string'
    value: 'Hello world!'
  - numberEquals: 1
```

```
    value: 1
```

Dans le dernier exemple, les deux comparaisons étant fausses, l'expression est évaluée à`false`.

```
or:
  - stringEquals: 'test_string'
    value: 'Hello world!'
  - numberEquals: 1
    value: 3
```

pas

Avec l'`not`opérateur, vous pouvez annuler une seule comparaison. L'expression indique `true` si la comparaison est fausse. Si la comparaison est vraie, l'expression est évaluée à`false`.

Exemples :

L'exemple suivant effectue une comparaison de chaînes. La comparaison étant fausse, l'expression est évaluée à`true`.

```
not:
  - stringEquals: 'test_string'
    value: 'Hello world!'
```

L'exemple suivant effectue également une comparaison de chaînes. La comparaison étant vraie, l'expression est évaluée à`false`.

```
not:
  - stringEquals: 'test_string'
    value: 'test_string'
```

Utiliser des constructions en boucle dans AWSTOE

Cette section fournit des informations pour vous aider à créer des constructions en boucle dans le. AWSTOE Les constructions en boucle définissent une séquence répétée d'instructions. Vous pouvez utiliser les types de constructions en boucle suivants dans : AWSTOE

- `for`constructs — Itérer sur une séquence bornée d'entiers.
- `forEach`construit

- `forEach`boucle avec liste d'entrée : itère sur une collection limitée de chaînes.
- `forEach`boucle avec liste délimitée : itère sur une collection limitée de chaînes jointes par un délimiteur.

> **Note**
>
> Les constructions en boucle ne prennent en charge que les types de données sous forme de chaîne.

Sujets de construction en boucle

- Variables d'itération de référence
- Types de constructions en boucle
- Champs d'étape
- Sorties d'étape et d'itération

Variables d'itération de référence

Pour faire référence à l'index et à la valeur de la variable d'itération en cours, l'expression de référence `{{ loop.* }}` doit être utilisée dans le corps d'entrée d'une étape contenant une construction en boucle. Cette expression ne peut pas être utilisée pour faire référence aux variables d'itération de la construction en boucle d'une autre étape.

L'expression de référence comprend les membres suivants :

- `{{ loop.index }}`— Position ordinale de l'itération en cours, indexée à. `0`
- `{{ loop.value }}`— La valeur associée à la variable d'itération en cours.

Noms des boucles

Toutes les constructions en boucle ont un champ de nom facultatif pour l'identification. Si un nom de boucle est fourni, il peut être utilisé pour faire référence à des variables d'itération dans le corps d'entrée de l'étape. Pour faire référence aux indices d'itération et aux valeurs d'une boucle nommée, utilisez `{{ <loop_name>.* }}` with `{{ loop.* }}` dans le corps de saisie de l'étape. Cette expression ne peut pas être utilisée pour faire référence à la construction en boucle nommée d'une autre étape.

L'expression de référence comprend les membres suivants :

- `{{ <loop_name>.index }}`— Position ordinale de l'itération en cours de la boucle nommée, qui est indexée à. `0`
- `{{ <loop_name>.value }}`— La valeur associée à la variable d'itération actuelle de la boucle nommée.

Résoudre les expressions de référence

AWSTOE Résout les expressions de référence comme suit :

- `{{ <loop_name>.* }}`— AWSTOE résout cette expression selon la logique suivante :
  - Si la boucle de l'étape en cours d'exécution correspond à la `<loop_name>` valeur, l'expression de référence est résolue en fonction de la structure en boucle de l'étape en cours d'exécution.
  - `<loop_name>`aboutit à la construction en boucle nommée si elle apparaît dans l'étape en cours d'exécution.
- `{{ loop.* }}`— AWSTOE résout l'expression à l'aide de la construction en boucle définie dans l'étape en cours d'exécution.

Si des expressions de référence sont utilisées dans une étape ne contenant pas de boucle, les expressions AWSTOE ne sont pas résolues et elles apparaissent dans l'étape sans être remplacées.

> **Note**
>
> Les expressions de référence doivent être placées entre guillemets pour être correctement interprétées par le compilateur YAML.

Types de constructions en boucle

Cette section fournit des informations et des exemples sur les types de construction en boucle qui peuvent être utilisés dans le AWSTOE.

Types de constructions en boucle

- pour boucle
- Boucle ForEach avec liste d'entrées
- Boucle ForEach avec liste délimitée

`pour` boucle

La `for` boucle itère sur une plage d'entiers spécifiée dans une limite définie par le début et la fin des variables. Les valeurs itératives font partie de l'ensemble `[start, end]` et incluent les valeurs limites.

AWSTOE vérifie les `updateBy` valeurs `start``end`, et pour s'assurer que la combinaison ne donne pas lieu à une boucle infinie.

`for`schéma de boucle

```
  - name: "StepName"
    action: "ActionModule"
    loop:
      name: "string"
      for:
        start: int
        end: int
        updateBy: int
inputs:
  ...
```

`pour` entrée en boucle

Champ	Description	Type	Obligatoire	Par défaut
`name`	Nom unique de la boucle. Il doit être unique par rapport aux autres noms de boucle de la même phase.	String	Non	""
`start`	Valeur de départ de l'itération. N'accepte pas le chaînage d'expressions.	Entier	Oui	n/a

Champ	Description	Type	Obligatoire	Par défaut
`end`	Valeur finale de l'itération. N'accepte pas le chaînage d'expressions.	Entier	Oui	n/a
`updateBy`	Différence selon laquelle une valeur itérative est mise à jour par addition. Il doit s'agir d'une valeur négative ou positive différente de zéro. N'accepte pas le chaînage d'expressions.	Entier	Oui	n/a

`for`exemple d'entrée en boucle

```
  - name: "CalculateFileUploadLatencies"
    action: "ExecutePowerShell"
    loop:
      for:
        start: 100000
        end: 1000000
        updateBy: 100000
    inputs:
      commands:
        - |
          $f = new-object System.IO.FileStream c:\temp\test{{ loop.index }}.txt,
 Create, ReadWrite
          $f.SetLength({{ loop.value }}MB)
          $f.Close()
        - c:\users\administrator\downloads\latencyTest.exe --file c:\temp
\test{{ loop.index }}.txt
```

```
        - AWS s3 cp c:\users\administrator\downloads\latencyMetrics.json s3://bucket/
latencyMetrics.json
        - |
          Remove-Item -Path c:\temp\test{{ loop.index }}.txt
          Remove-Item -Path c:\users\administrator\downloads\latencyMetrics.json
```

Boucle **`ForEach`** avec liste d'entrées

La `forEach` boucle itère sur une liste explicite de valeurs, qui peuvent être des chaînes ou des expressions enchaînées.

`forEach`boucle avec schéma de liste d'entrées

```
  - name: "StepName"
    action: "ActionModule"
    loop:
      name: "string"
      forEach:
        - "string"
    inputs:
  ...
```

Boucle **`ForEach`** avec entrée de liste d'entrées

Champ	Description	Type	Obligatoire	Par défaut
`name`	Nom unique de la boucle. Il doit être unique par rapport aux autres noms de boucle de la même phase.	String	Non	""
Liste des chaînes de `forEach` boucles	Liste des chaînes pour l'itération. Accepte les expressio ns chaînées	Liste de chaînes	Oui	n/a

Champ	Description	Type	Obligatoire	Par défaut
	sous forme de chaînes dans la liste. Les expressions chaînées doivent être placées entre guillemets pour que le compilateur YAML puisse les interpréter correctement.			

`forEach`boucle avec liste d'entrées, exemple 1

```
- name: "ExecuteCustomScripts"
  action: "ExecuteBash"
  loop:
    name: BatchExecLoop
    forEach:
      - /tmp/script1.sh
      - /tmp/script2.sh
      - /tmp/script3.sh
  inputs:
    commands:
      - echo "Count {{ BatchExecLoop.index }}"
      - sh "{{ loop.value }}"
      - |
        retVal=$?
        if [ $retVal -ne 0 ]; then
          echo "Failed"
        else
          echo "Passed"
       fi
```

`forEach`boucle avec liste d'entrées, exemple 2

```
- name: "RunMSIWithDifferentArgs"
```

```
  action: "ExecuteBinary"
  loop:
    name: MultiArgLoop
    forEach:
      - "ARG1=C:\Users ARG2=1"
      - "ARG1=C:\Users"
      - "ARG1=C:\Users ARG3=C:\Users\Administrator\Documents\f1.txt"
  inputs:
    commands:
      path: "c:\users\administrator\downloads\runner.exe"
      args:
        - "{{ MultiArgLoop.value }}"
```

`forEach`exemple de boucle avec liste d'entrées 3

```
- name: "DownloadAllBinaries"
  action: "S3Download"
  loop:
    name: MultiArgLoop
    forEach:
      - "bin1.exe"
      - "bin10.exe"
      - "bin5.exe"
  inputs:
    - source: "s3://bucket/{{ loop.value }}"
      destination: "c:\temp\{{ loop.value }}"
```

Boucle **`ForEach`** avec liste délimitée

La boucle itère sur une chaîne contenant des valeurs séparées par un délimiteur. Pour itérer sur les constituants de la chaîne, utilisez AWSTOE le délimiteur pour diviser la chaîne en un tableau adapté à l'itération.

`forEach`boucle avec schéma de liste délimité

```
- name: "StepName"
  action: "ActionModule"
  loop:
    name: "string"
    forEach:
      list: "string"
      delimiter: ".,;:\n\t -_"
```

```
    inputs:
  ...
```

Boucle **`ForEach`** avec entrée de liste délimitée

Champ	Description	Type	Obligatoire	Par défaut
`name`	Nom unique attribué à la boucle. Il doit être unique par rapport aux autres noms de boucles de la même phase.	String	Non	""
`list`	Chaîne composée de chaînes constitutives reliées par un caractère délimiteur commun. Accepte également les expressions chaînées. Dans le cas d'expressions chaînées, assurez-vous qu'elles sont placées entre guillemets pour une interprétation correcte par le compilateur YAML.	String	Oui	n/a

Champ	Description	Type	Obligatoire	Par défaut
`delimiter`	Caractère utilisé pour séparer les chaînes d'un bloc. La virgule est la virgule par défaut. Un seul caractère délimiteur est autorisé dans la liste donnée : • Point : `"."` • Virgule : `","` • Point-virgule : `";"` • Colon : `":"` • Nouvelle gamme : `"\n"` • Onglet : `"\t"` • Espace : `" "` • Trait d'union : `"-"` • Souligner : `"_"`	String	Non	Virgule : `","`

Champ	Description	Type	Obligatoire	Par défaut
	Les expressions de chaînage ne peuvent pas être utilisées.			

> **Note**
>
> La valeur de `list` est traitée comme une chaîne immuable. Si la source de `list` est modifiée pendant l'exécution, elle ne sera pas reflétée pendant l'exécution.

`forEach`boucle avec liste délimitée, exemple 1

Cet exemple utilise le modèle d'expression de chaînage suivant pour faire référence à la sortie d'une autre étape :`<phase_name>.<step_name>.[inputs | outputs].<var_name>`.

```
- name: "RunMSIs"
  action: "ExecuteBinary"
  loop:
    forEach:
      list: "{{ build.GetAllMSIPathsForInstallation.outputs.stdout }}"
      delimiter: "\n"
  inputs:
    commands:
      path: "{{ loop.value }}"
```

`forEach`boucle avec liste délimitée, exemple 2

```
- name: "UploadMetricFiles"
  action: "S3Upload"
  loop:
    forEach:
      list: "/tmp/m1.txt,/tmp/m2.txt,/tmp/m3.txt,..."
  inputs:
    commands:
      - source: "{{ loop.value }}"
        destination: "s3://bucket/key/{{ loop.value }}"
```

Champs d'étape

Les boucles font partie d'une étape. Aucun champ lié à l'exécution d'une étape n'est appliqué aux itérations individuelles. Les champs d'étape s'appliquent uniquement au niveau de l'étape, comme suit :

- TimeoutSeconds — Toutes les itérations de la boucle doivent être exécutées dans le délai spécifié par ce champ. Si le délai d'exécution de la boucle expire, AWSTOE exécute la politique de nouvelle tentative de l'étape et réinitialise le paramètre de délai d'expiration pour chaque nouvelle tentative. Si le délai d'exécution de la boucle dépasse le délai d'expiration après avoir atteint le nombre maximal de tentatives, le message d'échec de l'étape indique que le délai d'exécution de la boucle a expiré.
- OnFailure — La gestion des défaillances est appliquée à l'étape comme suit :
  - Si OnFailure est défini sur`Abort`, AWSTOE quitte la boucle et réessaie l'étape conformément à la politique de nouvelles tentatives. Une fois le nombre maximal de tentatives atteint, AWSTOE marque l'étape en cours comme ayant échoué et arrête l'exécution du processus.

    AWSTOE définit le code d'état de la phase parent et du document sur`Failed`.

    > **Note**
    >
    > Aucune autre étape n'est exécutée après l'échec de l'étape.

  - Si OnFailure est défini sur`Continue`, AWSTOE quitte la boucle et réessaie l'étape conformément à la politique de nouvelles tentatives. Après le nombre maximal de tentatives, AWSTOE marque l'étape en cours comme ayant échoué et passe à l'étape suivante.

    AWSTOE définit le code d'état de la phase parent et du document sur`Failed`.

  - Si OnFailure est défini sur`Ignore`, AWSTOE quitte la boucle et réessaie l'étape conformément à la politique de nouvelles tentatives. Une fois le nombre maximal de tentatives atteint, AWSTOE marque l'étape en cours comme telle `IgnoredFailure` et passe à l'étape suivante.

    AWSTOE définit le code d'état de la phase parent et du document sur`SuccessWithIgnoredFailure`.

> **Note**
>
> Ceci est toujours considéré comme une exécution réussie, mais inclut des informations vous indiquant qu'une ou plusieurs étapes ont échoué et ont été ignorées.

- MaxAttempts — À chaque nouvelle tentative, l'étape complète et toutes les itérations sont exécutées depuis le début.
- status — État général de l'exécution d'une étape. `status`ne représente pas le statut des itérations individuelles. Le statut d'une étape comportant des boucles est déterminé comme suit :
  - Si une seule itération échoue, le statut d'une étape indique un échec.
  - Si toutes les itérations aboutissent, le statut d'une étape indique un succès.
- StartTime : heure de début globale de l'exécution d'une étape. Ne représente pas l'heure de début des itérations individuelles.
- EndTime — Heure de fin globale de l'exécution d'une étape. Ne représente pas l'heure de fin des itérations individuelles.
- FailureMessage — Inclut les indices d'itération qui ont échoué en cas d'erreurs autres que le délai d'expiration. En cas d'erreur de temporisation, le message indique que l'exécution de la boucle a échoué. Aucun message d'erreur individuel n'est fourni pour chaque itération afin de minimiser la taille des messages d'échec.

Sorties d'étape et d'itération

Chaque itération contient une sortie. À la fin d'une exécution en boucle, AWSTOE consolide toutes les sorties d'itération réussies dans. `detailedOutput.json` Les sorties consolidées sont un assemblage de valeurs appartenant aux clés de sortie correspondantes telles que définies dans le schéma de sortie du module d'action. L'exemple suivant montre comment les sorties sont consolidées :

Sortie de **`ExecuteBash`** pour l'itération 1

```
{
 "stdout":"Hello"
}
```

Sortie de **`ExecuteBash`** pour l'itération 2

```
{
 "stdout":"World"
}
```

Sortie de **`ExecuteBash`** for Step

```
{
 "stdout":"Hello\nWorld"
}
```

Par exemple, `ExecuteBash`, `ExecutePowerShell`, et `ExecuteBinary` sont des modules d'action qui renvoient `STDOUT` en tant que sortie du module d'action. `STDOUT`les messages sont joints au nouveau caractère de ligne pour produire le résultat global de l'étape`detailedOutput.json`.

AWSTOE ne consolidera pas les résultats des itérations infructueuses.

Modules d'action pris en charge par AWSTOE gestionnaire de composants

Les services de création d'images, tels que EC2 Image Builder, AWSTOE utilisent des modules d'action pour configurer les instances EC2 utilisées pour créer et tester des images de machine personnalisées. Cette section décrit les fonctionnalités des modules d' AWSTOE action couramment utilisés et explique comment les configurer, y compris des exemples.

Les composants sont créés à partir de documents YAML en texte brut. Pour plus d'informations sur la syntaxe des documents, consultezUtilisez la commande AWSTOE cadre de documentation sur les composants pour les composants personnalisés.

> **Note**
>
> Tous les modules d'action utilisent le même compte que l'agent Systems Manager lorsqu'ils s'exécutent, sous Linux et `NT Authority\SYSTEM` sous Windows. `root`

La référence croisée suivante classe les modules d'action en fonction du type d'actions qu'ils exécutent.

Exécution générale

- Assert (Linux, Windows, macOS)
- ExecuteBash (Linux, macOS)
- ExecuteBinary (Linux, Windows, macOS)
- ExecuteDocument (Linux, Windows, macOS)
- ExecutePowerShell (Fenêtres)

Téléchargement et envoi de fichiers

- Téléchargement de S3 (Linux, Windows, macOS)
- Téléchargement en mode S3 (Linux, Windows, macOS)
- WebDownload (Linux, Windows, macOS)

Opérations du système de fichiers

- AppendFile (Linux, Windows, macOS)
- CopyFile (Linux, Windows, macOS)
- CopyFolder (Linux, Windows, macOS)
- CreateFile (Linux, Windows, macOS)
- CreateFolder (Linux, Windows, macOS)
- CreateSymlink (Linux, Windows, macOS)
- DeleteFile (Linux, Windows, macOS)
- DeleteFolder (Linux, Windows, macOS)
- ListFiles (Linux, Windows, macOS)
- MoveFile (Linux, Windows, macOS)
- MoveFolder (Linux, Windows, macOS)
- ReadFile (Linux, Windows, macOS)
- SetFileEncoding (Linux, Windows, macOS)
- SetFileOwner (Linux, Windows, macOS)

- SetFolderOwner (Linux, Windows, macOS)
- SetFilePermissions (Linux, Windows, macOS)
- SetFolderPermissions (Linux, Windows, macOS)

Actions d'installation du logiciel

- Installez MSI (Windows)
- Désinstallez MSI (Windows)

Actions du système

- Redémarrer (Linux, Windows)
- SetRegistry (Fenêtres)
- Mettre à jour le système d'exploitation (Linux, Windows)

Modules d'exécution généraux

La section suivante contient des détails sur les modules d'action qui exécutent des commandes et contrôlent le flux de travail d'exécution.

Modules d'action d'exécution générale

- Assert (Linux, Windows, macOS)
- ExecuteBash (Linux, macOS)
- ExecuteBinary (Linux, Windows, macOS)
- ExecuteDocument (Linux, Windows, macOS)
- ExecutePowerShell (Fenêtres)

Assert (Linux, Windows, macOS)

Le module d'action Assert effectue des comparaisons de valeurs en utilisant Opérateurs de comparaison ou en Opérateurs logiques tant qu'entrée. Le résultat de l'expression de l'opérateur (vrai ou faux) indique le statut global de réussite ou d'échec de l'étape.

Si la comparaison ou l'expression de l'opérateur logique est évaluée à`true`, l'étape est marquée comme`Success`. Dans le cas contraire, l'étape est marquée comme`Failed`. Si l'étape échoue, le `onFailure` paramètre détermine le résultat de l'étape.

Input

Nom de la touche	Description	Type	Obligatoire
`input`	Contient une comparaison ou un opérateur logique unique. Notez que les opérateurs logiques peuvent contenir plusieurs opérateurs de comparaison.	Ceci est variable en fonction de l'opérateur	Oui

Exemple de saisie : comparaison simple à l'aide de l'opérateur de **`stringEquals`** comparaison

Cet exemple évalue à. `true`

```
- name: StringComparison
  action: Assert
  inputs:
    stringEquals: '2.1.1'
    value: '{{ validate.ApplicationVersion.outputs.stdout }}'
```

Exemple de saisie : comparaisons Regex à l'aide de l'opérateur de **`patternMatches`** comparaison

Ces exemples sont tous évalués à`true`.

```
- name: Letters only
  action: Assert
  inputs:
    patternMatches: '^[a-zA-Z]+$'
    value: 'ThisIsOnlyLetters'

- name: Letters and spaces only
  action: Assert
  inputs:
    patternMatches: '^[a-zA-Z\s]+$'
```

```
    value: 'This text contains spaces'

- name: Numbers only
  action: Assert
  inputs:
    patternMatches: '^[0-9]+$'
    value: '1234567890'
```

Exemple d'entrée : comparaisons imbriquées avec des opérateurs logiques et des variables chaînées

L'exemple suivant illustre des comparaisons imbriquées avec des opérateurs logiques qui utilisent des comparaisons avec des variables chaînées. `Assert`Évalue `true` si l'une des conditions suivantes est vraie :

- Le `ApplicationVersion` est supérieur à `2.0` et `CPUArchitecture` égal`arm64`.
- Les `CPUArchitecture` égaux`x86_64`.

```
- name: NestedComparisons
  action: Assert
  inputs:
    or: # <- first level deep
      - and: # <- second level deep
          - numberGreaterThan: 2.0 # <- third level deep
            value: '{{ validate.ApplicationVersion.outputs.stdout }}'
          - stringEquals: 'arm64'
            value: '{{ validate.CPUArchitecture.outputs.stdout }}'
      - stringEquals: 'x86_64'
        value: '{{ validate.CPUArchitecture.outputs.stdout }}'
```

Sortie :

Le résultat d'une `Assert` est le succès ou l'échec de l'étape.

ExecuteBash (Linux, macOS)

Le module ExecuteBashd'action vous permet d'exécuter des scripts bash avec un shell intégré. code/commands Ce module est compatible avec Linux.

Toutes les commandes et instructions que vous spécifiez dans le bloc de commandes sont converties dans un fichier (par exemple`input.sh`) et exécutées avec le shell bash. Le résultat de l'exécution du fichier shell est le code de sortie de l'étape.

Le ExecuteBashmodule gère les redémarrages du système si le script se termine avec un code de sortie de. 194 Lorsqu'elle est lancée, l'application exécute l'une des actions suivantes :

- L'application transmet le code de sortie à l'appelant s'il est exécuté par l'agent Systems Manager. L'agent Systems Manager gère le redémarrage du système et exécute la même étape que celle à l'origine du redémarrage, comme décrit dans la section Redémarrage d'une instance gérée à partir de scripts.
- L'application enregistre le courant`executionstate`, configure un déclencheur de redémarrage pour réexécuter l'application et redémarre le système.

Après le redémarrage du système, l'application exécute la même étape que celle à l'origine du redémarrage. Si vous avez besoin de cette fonctionnalité, vous devez écrire des scripts idempotents capables de gérer plusieurs invocations de la même commande shell.

Input

Nom de la touche	Description	Type	Obligatoire
`commands`	Contient une liste d'instructions ou de commandes à exécuter conformément à la syntaxe bash. Multi-line Le YAML est autorisé.	List	Oui

Exemple de saisie : avant et après un redémarrage

```
name: ExitCode194Example
description: This shows how the exit code can be used to restart a system with
 ExecuteBash
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: RestartTrigger
        action: ExecuteBash
        inputs:
          commands:
```

```
            - |
              REBOOT_INDICATOR=/var/tmp/reboot-indicator
              if [ -f "${REBOOT_INDICATOR}" ]; then
                echo 'The reboot file exists. Deleting it and exiting with success.'
                rm "${REBOOT_INDICATOR}"
                exit 0
              fi
              echo 'The reboot file does not exist. Creating it and triggering a
 restart.'
              touch "${REBOOT_INDICATOR}"
              exit 194
```

Output

Champ	Description	Type
`stdout`	Sortie standard de l'exécution des commandes.	chaîne

Si vous lancez un redémarrage et que vous renvoyez le code de sortie dans le 194 cadre du module d'action, la compilation reprendra à l'étape du module d'action qui a initié le redémarrage. Si vous démarrez un redémarrage sans le code de sortie, le processus de génération risque d'échouer.

Exemple de sortie : avant le redémarrage (première fois via le document)

```
{
 “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

Exemple de sortie : après le redémarrage, (deuxième fois dans le document)

```
{
 “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

ExecuteBinary (Linux, Windows, macOS)

Le module ExecuteBinaryd'action vous permet d'exécuter des fichiers binaires avec une liste d'arguments de ligne de commande.

Le ExecuteBinarymodule gère les redémarrages du système si le fichier binaire se termine avec un code de sortie `194` (Linux) ou `3010` (Windows). Dans ce cas, l'application exécute l'une des actions suivantes :

- L'application transmet le code de sortie à l'appelant s'il est exécuté par l'agent Systems Manager. L'agent Systems Manager gère le redémarrage du système et exécute la même étape que celle qui a initié le redémarrage, comme décrit dans Redémarrage d'une instance gérée à partir de scripts.
- L'application enregistre le courant`executionstate`, configure un déclencheur de redémarrage pour réexécuter l'application et redémarre le système.

Après le redémarrage du système, l'application exécute la même étape que celle à l'origine du redémarrage. Si vous avez besoin de cette fonctionnalité, vous devez écrire des scripts idempotents capables de gérer plusieurs invocations de la même commande shell.

Input

Nom de la touche	Description	Type	Obligatoire
`path`	Le chemin d'accès au fichier binaire à exécuter.	String	Oui
`arguments`	Contient une liste d'arguments de ligne de commande à utiliser lors de l'exécution du binaire.	Liste de chaînes	Non

Exemple de saisie : installer .NET

```
- name: "InstallDotnet"
  action: ExecuteBinary
  inputs:
    path: C:\PathTo\dotnet_installer.exe
    arguments:
      - /qb
      - /norestart
```

Output

Champ	Description	Type
`stdout`	Sortie standard de l'exécution des commandes.	chaîne

Exemple de sortie

```
{
 "stdout": "success"
}
```

ExecuteDocument (Linux, Windows, macOS)

Le module ExecuteDocumentd'action ajoute la prise en charge des documents de composants imbriqués, en exécutant plusieurs documents de composants à partir d'un seul document. AWSTOE valide le document transmis dans le paramètre d'entrée lors de l'exécution.

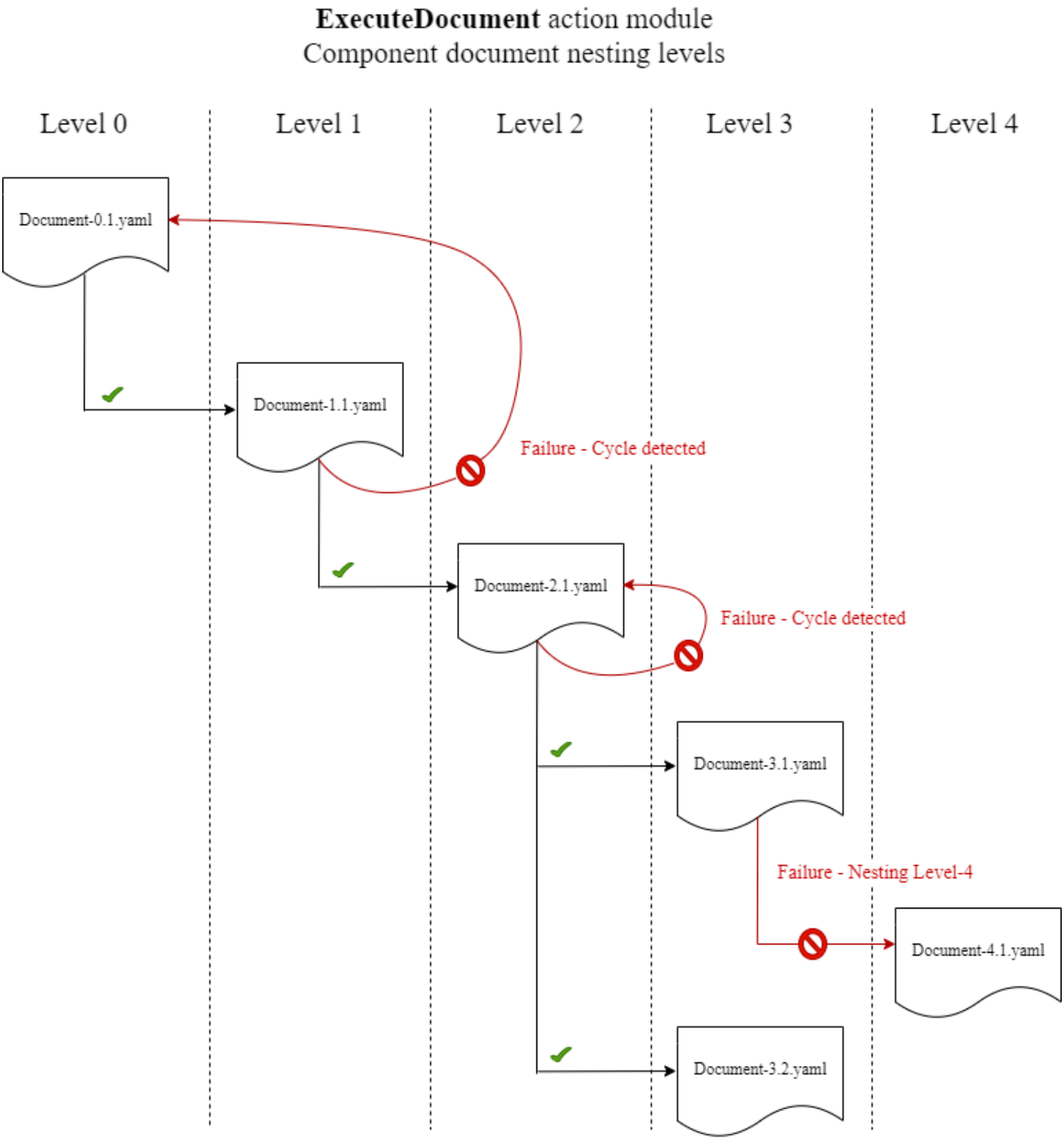
Restrictions

- Ce module d'action ne s'exécute qu'une seule fois, sans qu'aucune nouvelle tentative ne soit autorisée et qu'aucune option ne permet de définir des limites de délai d'expiration. ExecuteDocumentdéfinit les valeurs par défaut suivantes et renvoie une erreur si vous essayez de les modifier.
  - `timeoutSeconds`: -1
  - `maxAttempts` : 1

  > **Note**
  >
  > Vous pouvez laisser ces valeurs vides et AWSTOE utiliser les valeurs par défaut.

- L'imbrication de documents est autorisée, jusqu'à trois niveaux de profondeur, mais pas plus. Trois niveaux d'imbrication se traduisent par quatre niveaux de document, le niveau supérieur n'étant pas imbriqué. Dans ce scénario, le document de niveau le plus bas ne doit appeler aucun autre document.
- L'exécution cyclique des documents des composants n'est pas autorisée. Tout document qui s'appelle lui-même en dehors d'une construction en boucle, ou qui appelle un autre document situé

plus haut dans la chaîne d'exécution actuelle, lance un cycle qui peut entraîner une boucle sans fin. Lorsqu'il AWSTOE détecte une exécution cyclique, il arrête l'exécution et enregistre l'échec.



Si un document de composant essaie de s'exécuter lui-même ou d'exécuter l'un des documents de composant situés plus haut dans la chaîne d'exécution en cours, l'exécution échoue.

Entrée

Nom de la touche	Description	Type	Obligatoire
`document`	Chemin du document du composant. Les options valides sont les suivantes : • Chemins de fichiers locaux • URI S3 • ARN de la version de build du composant EC2 Image Builder	String	Oui
`document-s3-bucket-owner`	L'ID de compte du propriétaire du compartiment S3 pour le compartiment S3 dans lequel les documents des composants sont stockés. (Recommandé si vous utilisez des URI S3 dans le document de votre composant.)	String	Non
`phases`		String	Non

Nom de la touche	Description	Type	Obligatoire
	Phases à exécuter dans le document du composant, exprimées sous forme de liste séparée par des virgules. Si aucune phase n'est spécifiée, toutes les phases sont exécutées.		
`parameters`	Paramètres d'entrée transmis au document du composant lors de l'exécution sous forme de paires clé-valeur.	Liste des cartes de paramètres	Non

Entrée de mappage de paramètres

Nom de la touche	Description	Type	Obligatoire
`name`	Nom du paramètre d'entrée à transmett re au document du composant que le module ExecuteDo cumentd'action exécute.	String	Oui
`value`	La valeur du paramètre d'entrée.	String	Oui

Exemples de saisie

Les exemples suivants montrent les variations des entrées pour le document de votre composant, en fonction de votre chemin d'installation.

Exemple de saisie : chemin du document local

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: Sample-1.yaml
          phases: build
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Exemple de saisie : URI S3 en tant que chemin de document

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: s3://my-bucket/Sample-1.yaml
          document-s3-bucket-owner: 123456789012
          phases: build,validate
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Exemple de saisie : ARN du composant EC2 Image Builder en tant que chemin de document

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: arn:aws:imagebuilder:us-west-2:aws:component/Sample-Test/1.0.0
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Utiliser une ForEach boucle pour exécuter des documents

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForEachLoop'
          forEach:
            - Sample-1.yaml
            - Sample-2.yaml
        inputs:
          document: "{{myForEachLoop.value}}"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Utilisation d'une boucle For pour exécuter des documents

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForLoop'
          for:
            start: 1
            end: 2
            updateBy: 1
        inputs:
          document: "Sample-{{myForLoop.value}}.yaml"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Output

AWSTOE crée un fichier de sortie appelé à `detailedoutput.json` chaque fois qu'il s'exécute. Le fichier contient des détails sur chaque phase et étape de chaque document composant invoqué pendant son exécution. Pour le module ExecuteDocumentd'action, vous trouverez un bref résumé de l'exécution dans le `outputs` champ, ainsi que des détails sur les phases, les étapes et les documents qu'il exécute dans le`detailedOutput`.

```
{
 \"executedStepCount\":1,\"executionId\":\"97054e22-06cc-11ec-9b14-acde48001122\",
\"failedStepCount\":0,\"failureMessage\":\"\",\"ignoredFailedStepCount\":0,\"logUrl\":
\"\",\"status\":\"success\"
}",
```

L'objet récapitulatif de sortie de chaque document de composant contient les détails suivants, comme indiqué ici, avec des exemples de valeurs :

- exécuté StepCount « :1
- « ID d'exécution » « 12345a67-89bc-01de-2f34-abcd56789012 »
- « échec StepCount » :0
- « Message d'échec » : « »
- « ignoré FailedStepCount » :0
- « URL du journal » : « »
- « statut » « succès »

Exemple de sortie

L'exemple suivant montre la sortie du module ExecuteDocumentd'action lorsqu'une exécution imbriquée se produit. Dans cet exemple, le document du `main.yaml` composant exécute correctement le document du `Sample-1.yaml` composant.

```
{
    "executionId": "12345a67-89bc-01de-2f34-abcd56789012",
    "status": "success",
    "startTime": "2021-08-26T17:20:31-07:00",
    "endTime": "2021-08-26T17:20:31-07:00",
    "failureMessage": "",
    "documents": [
        {
            "name": "",
            "filePath": "main.yaml",
            "status": "success",
            "description": "",
            "startTime": "2021-08-26T17:20:31-07:00",
            "endTime": "2021-08-26T17:20:31-07:00",
            "failureMessage": "",
            "phases": [
                {
                    "name": "build",
                    "status": "success",
                    "startTime": "2021-08-26T17:20:31-07:00",
                    "endTime": "2021-08-26T17:20:31-07:00",
                    "failureMessage": "",
                    "steps": [
                        {
                            "name": "ExecuteNestedDocument",
                            "status": "success",
```

```
                            "failureMessage": "",
                            "timeoutSeconds": -1,
                            "onFailure": "Abort",
                            "maxAttempts": 1,
                            "action": "ExecuteDocument",
                            "startTime": "2021-08-26T17:20:31-07:00",
                            "endTime": "2021-08-26T17:20:31-07:00",
                            "inputs": "[{\"document\":\"Sample-1.yaml\",\"document-s3-
bucket-owner\":\"\",\"phases\":\"\",\"parameters\":null}]",
                            "outputs": "[{\"executedStepCount\":1,\"executionId\":
\"98765f43-21ed-09cb-8a76-fedc54321098\",\"failedStepCount\":0,\"failureMessage\":\"\",
\"ignoredFailedStepCount\":0,\"logUrl\":\"\",\"status\":\"success\"}]",
                            "loop": null,
                            "detailedOutput": [
                                {
                                    "executionId": "98765f43-21ed-09cb-8a76-
fedc54321098",
                                    "status": "success",
                                    "startTime": "2021-08-26T17:20:31-07:00",
                                    "endTime": "2021-08-26T17:20:31-07:00",
                                    "failureMessage": "",
                                    "documents": [
                                        {
                                            "name": "",
                                            "filePath": "Sample-1.yaml",
                                            "status": "success",
                                            "description": "",
                                            "startTime": "2021-08-26T17:20:31-07:00",
                                            "endTime": "2021-08-26T17:20:31-07:00",
                                            "failureMessage": "",
                                            "phases": [
                                                {
                                                    "name": "build",
                                                    "status": "success",
                                                    "startTime":
 "2021-08-26T17:20:31-07:00",
                                                    "endTime":
 "2021-08-26T17:20:31-07:00",
                                                    "failureMessage": "",
                                                    "steps": [
                                                        {
                                                            "name": "ExecuteBashStep",
                                                            "status": "success",
                                                            "failureMessage": "",
```

```
                                                    "timeoutSeconds": 7200,
                                                    "onFailure": "Abort",
                                                    "maxAttempts": 1,
                                                    "action": "ExecuteBash",
                                                    "startTime":
 "2021-08-26T17:20:31-07:00",
                                                    "endTime":
 "2021-08-26T17:20:31-07:00",
                                                    "inputs": "[{\"commands\":
[\"echo \\\"Hello World!\\\"\"]}]",
                                                    "outputs": "[{\"stdout\":
\"Hello World!\"}]",
                                                    "loop": null,
                                                    "detailedOutput": null
                                                }]
                                            }]
                                        }]
                                    }]
                                }]

                            }]
                        }]
}
```

ExecutePowerShell (Fenêtres)

Le module ExecutePowerShelld'action vous permet d'exécuter PowerShell des scripts avec un shell code/commands intégré. Ce module prend en charge la plate-forme Windows et Windows PowerShell.

Tous les éléments commands/instructions spécifiés dans le bloc de commandes sont convertis en un fichier de script (par exemple,`input.ps1`) et exécutés sous WindowsPowerShell. Le résultat de l'exécution du fichier shell est le code de sortie.

Le ExecutePowerShellmodule gère les redémarrages du système si la commande shell se termine avec un code de sortie de. `3010` Lorsqu'elle est lancée, l'application exécute l'une des actions suivantes :

- Transmet le code de sortie à l'appelant s'il est exécuté par l'agent Systems Manager. L'agent Systems Manager gère le redémarrage du système et exécute la même étape que celle à l'origine du redémarrage, comme décrit dans la section Redémarrage d'une instance gérée à partir de scripts.

- Enregistre le courant`executionstate`, configure un déclencheur de redémarrage pour réexécuter l'application et redémarre le système.

Après le redémarrage du système, l'application exécute la même étape que celle à l'origine du redémarrage. Si vous avez besoin de cette fonctionnalité, vous devez écrire des scripts idempotents capables de gérer plusieurs invocations de la même commande shell.

Input

Nom de la touche	Description	Type	Obligatoire
`commands`	Contient une liste d'instructions ou de commandes à exécuter conformément à PowerShell la syntaxe. Multi-line Le YAML est autorisé.	Liste de chaînes	Oui. Doit spécifier `commands` ou `file` pas les deux.
`file`	Contient le chemin d'accès à un fichier de PowerShell script. PowerShell sera exécuté sur ce fichier en utilisant l'argument de ligne de `-file` commande. Le chemin doit pointer vers un `.ps1` fichier.	String	Oui. Doit spécifier `commands` ou `file` pas les deux.

Exemple de saisie : avant et après un redémarrage

```
name: ExitCode3010Example
description: This shows how the exit code can be used to restart a system with
 ExecutePowerShell
schemaVersion: 1.0
phases:
  - name: build
```

```
    steps:
      - name: RestartTrigger
        action: ExecutePowerShell
        inputs:
          commands:
            - |
              $rebootIndicator = Join-Path -Path $env:SystemDrive -ChildPath 'reboot-
indicator'
              if (Test-Path -Path $rebootIndicator) {
                Write-Host 'The reboot file exists. Deleting it and exiting with
 success.'
                Remove-Item -Path $rebootIndicator -Force | Out-Null
                [System.Environment]::Exit(0)
              }
              Write-Host 'The reboot file does not exist. Creating it and triggering a
 restart.'
              New-Item -Path $rebootIndicator -ItemType File | Out-Null
              [System.Environment]::Exit(3010)
```

Output

Champ	Description	Type
`stdout`	Sortie standard de l'exécution des commandes.	chaîne

Si vous exécutez un redémarrage et que vous renvoyez le code de sortie dans le `3010` cadre du module d'action, la compilation reprendra à l'étape du module d'action qui a initié le redémarrage. Si vous redémarrez sans le code de sortie, le processus de génération risque d'échouer.

Exemple de sortie : avant le redémarrage (première fois via le document)

```
{
 “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

Exemple de sortie : après le redémarrage, (deuxième fois dans le document)

```
{
 “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

Modules de téléchargement et de téléversement de fichiers

La section suivante contient des détails sur les modules d'action qui chargent ou téléchargent des fichiers.

Téléchargez et chargez des modules d'action

- Téléchargement de S3 (Linux, Windows, macOS)
- Téléchargement en mode S3 (Linux, Windows, macOS)
- WebDownload (Linux, Windows, macOS)

Téléchargement de S3 (Linux, Windows, macOS)

Avec le module `S3Download` d'action, vous pouvez télécharger un objet Amazon S3, ou un ensemble d'objets, dans un fichier ou un dossier local que vous spécifiez avec le `destination` chemin. Si un fichier existe déjà à l'emplacement spécifié et que l'`overwrite`indicateur est défini sur true, le fichier `S3Download` est remplacé.

Votre `source` position peut pointer vers un objet spécifique dans Amazon S3, ou vous pouvez utiliser un préfixe de clé avec un astérisque (`*`) pour télécharger un ensemble d'objets correspondant au chemin du préfixe clé. Lorsque vous spécifiez un préfixe clé dans votre `source` position, le module `S3Download` d'action télécharge tout ce qui correspond au préfixe (fichiers et dossiers inclus). Assurez-vous que le préfixe clé se termine par une barre oblique, suivie d'un astérisque (`/*`), afin de télécharger tout ce qui correspond au préfixe. Par exemple : *`s3://my-bucket/my-folder/*`*.

Si l'`S3Download`action pour un préfixe de clé spécifié échoue lors d'un téléchargement, le contenu du dossier n'est pas rétabli dans son état antérieur à l'échec. Le dossier de destination reste tel qu'il était au moment de l'échec.

Cas d'utilisation pris en charge

Le module `S3Download` d'action prend en charge les cas d'utilisation suivants :

- L'objet Amazon S3 est téléchargé dans un dossier local, comme indiqué dans le chemin de téléchargement.
- Les objets Amazon S3 (avec un préfixe clé dans le chemin du fichier Amazon S3) sont téléchargés dans le dossier local spécifié, qui copie de manière récursive tous les objets Amazon S3 correspondant au préfixe clé dans le dossier local.

Exigences relatives à l'IAM

Le rôle IAM que vous associez à votre profil d'instance doit être autorisé à exécuter le module `S3Download` d'action. Les politiques IAM suivantes doivent être associées au rôle IAM associé au profil d'instance :

- Fichier unique : `s3:GetObject` contre le bucket/object (par exemple,`arn:aws:s3:::BucketName/*`).
- Plusieurs fichiers : `s3:ListBucket` contre le bucket/object (par exemple,`arn:aws:s3:::BucketName`) et `s3:GetObject` contre le bucket/object (par exemple,`arn:aws:s3:::BucketName/*`).

Input

Clé	Description	Type	Obligatoire	Par défaut
`source`	Le compartiment Amazon S3 qui est la source de votre télécharg ement. Vous pouvez spécifier le chemin d'accès à un objet spécifiqu e ou utiliser un préfixe de clé se terminant par une barre oblique suivie d'un astérisqu e (`/*`) pour télécharger un ensemble d'objets	String	Oui	N/A

Clé	Description	Type	Obligatoire	Par défaut
	correspondant au préfixe de clé.			
`destination`	Le chemin local où les objets Amazon S3 sont téléchargés. Pour télécharger un seul fichier, vous devez spécifier le nom du fichier dans le chemin. Par exemple, */myfolder/package.zip* .	String	Oui	N/A
`expectedBucketOwner`	ID de compte propriétaire attendu du bucket indiqué dans le `source` chemin. Nous vous recommandons de vérifier la propriété du compartiment Amazon S3 spécifié dans la source.	String	Non	N/A

Clé	Description	Type	Obligatoire	Par défaut
`overwrite`	Lorsqu'il est défini sur true, si un fichier du même nom existe déjà dans le dossier de destination pour le chemin local spécifié, le fichier de télécharg ement remplace le fichier local. Lorsqu'il est défini sur false, le fichier existant sur le système local est protégé contre le remplacem ent et le module d'action échoue en raison d'une erreur de téléchargement. Par exemple, `Error: S3Download: File already exists and "overwrit e" property for`	Booléen	Non	true

Clé	Description	Type	Obligatoire	Par défaut
	"destination" file is set to false. Cannot download.			

> **Note**
>
> Dans les exemples suivants, le chemin du dossier Windows peut être remplacé par un chemin Linux. Par exemple, *C:\myfolder\package.zip* peut être remplacé par*/myfolder/package.zip*.

Exemple de saisie : copier un objet Amazon S3 dans un fichier local

L'exemple suivant montre comment copier un objet Amazon S3 dans un fichier local.

```
- name: DownloadMyFile
  action: S3Download
  inputs:
    - source: s3://amzn-s3-demo-source-bucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://amzn-s3-demo-source-bucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: true
    - source: s3://amzn-s3-demo-source-bucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
```

Exemple de saisie : copier tous les objets Amazon S3 dans un compartiment Amazon S3 avec un préfixe de clé dans un dossier local

L'exemple suivant montre comment copier tous les objets Amazon S3 d'un compartiment Amazon S3 avec le préfixe key dans un dossier local. Amazon S3 n'a aucune notion de dossier. Par conséquent, tous les objets correspondant au préfixe de clé sont copiés. Le nombre maximum d'objets pouvant être téléchargés est de 1 000.

```
- name: MyS3DownloadKeyprefix
  action: S3Download
  maxAttempts: 3
  inputs:
    - source: s3://amzn-s3-demo-source-bucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://amzn-s3-demo-source-bucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
      overwrite: true
    - source: s3://amzn-s3-demo-source-bucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
```

Output

Aucune.

Téléchargement en mode S3 (Linux, Windows, macOS)

Avec le module d'action S3Upload, vous pouvez télécharger un fichier depuis un fichier ou un dossier source vers un emplacement Amazon S3. Vous pouvez utiliser un caractère générique (*) dans le chemin indiqué pour votre emplacement source afin de télécharger tous les fichiers dont le chemin correspond au modèle générique.

Si l'action récursive S3Upload échoue, tous les fichiers déjà chargés resteront dans le compartiment Amazon S3 de destination.

Cas d'utilisation pris en charge

- Fichier local vers un objet Amazon S3.
- Fichiers locaux dans un dossier (avec caractère générique) avec le préfixe de clé Amazon S3.
- Copiez le dossier local (doit avoir été `recurse` défini sur`true`) dans le préfixe de clé Amazon S3.

Exigences relatives à l'IAM

Le rôle IAM que vous associez à votre profil d'instance doit être autorisé à exécuter le module `S3Upload` d'action. La politique IAM suivante doit être attachée au rôle IAM associé au profil d'instance. La politique doit accorder `s3:PutObject` des autorisations au compartiment Amazon S3 cible. Par exemple, `arn:aws:s3:::BucketName/*`).

Input

Clé	Description	Type	Obligatoire	Par défaut
`source`	Le chemin local d'où files/folders provient la source. `source`Supporte un caractère générique astérisque (`*`).	String	Oui	N/A
`destination`	Le chemin du compartiment Amazon S3 de destination où files/folders les sources sont téléchargées.	String	Oui	N/A
`recurse`	Lorsqu'il est défini sur`true`, exécute S3Upload de manière récursive.	String	Non	`false`
		String	Non	N/A

Clé	Description	Type	Obligatoire	Par défaut
`expectedBucketOwner`	L'ID de compte propriétaire attendu pour le compartiment Amazon S3 spécifié dans le chemin de destination. Nous vous recommandons de vérifier la propriété du compartiment Amazon S3 spécifié dans la destination.			

Exemple de saisie : copie d'un fichier local dans un objet Amazon S3

L'exemple suivant montre comment copier un fichier local dans un objet Amazon S3.

```
- name: MyS3UploadFile
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\package.zip
      destination: s3://amzn-s3-demo-destination-bucket/path/to/package.zip
      expectedBucketOwner: 123456789022
```

Exemple de saisie : copier tous les fichiers d'un dossier local dans un compartiment Amazon S3 avec le préfixe key

L'exemple suivant montre comment copier tous les fichiers du dossier local dans un compartiment Amazon S3 avec le préfixe key. Cet exemple ne copie pas les sous-dossiers ou leur contenu car cela n'`recurse`est pas spécifié, et sa valeur par défaut est. `false`

```
- name: MyS3UploadMultipleFiles
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\*
      destination: s3://amzn-s3-demo-destination-bucket/path/to/
      expectedBucketOwner: 123456789022
```

Exemple de saisie : copie récursive de tous les fichiers et dossiers d'un dossier local vers un compartiment Amazon S3

L'exemple suivant montre comment copier tous les fichiers et dossiers de manière récursive d'un dossier local vers un compartiment Amazon S3 avec le préfixe key.

```
- name: MyS3UploadFolder
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\*
      destination: s3://amzn-s3-demo-destination-bucket/path/to/
      recurse: true
      expectedBucketOwner: 123456789022
```

Output

Aucune.

WebDownload (Linux, Windows, macOS)

Le module WebDownloadd'action vous permet de télécharger des fichiers et des ressources depuis un emplacement distant via le HTTP/HTTPS protocole (le protocole HTTPS est recommandé). Il n'y a aucune limite quant au nombre ou à la taille des téléchargements. Ce module gère la logique des nouvelles tentatives et des interruptions exponentielles.

Chaque opération de téléchargement dispose d'un maximum de 5 tentatives pour réussir en fonction des entrées de l'utilisateur. Ces tentatives sont différentes de celles spécifiées dans le `maxAttempts` champ du document`steps`, qui sont liées à des défaillances du module d'action.

Ce module d'action gère implicitement les redirections. Tous les codes d'état HTTP, à l'exception de`200`, génèrent une erreur.

Input

Nom de la touche	Description	Type	Obligatoire	Par défaut
`source`	HTTP/HTTPS URL valide (HTTPS est recommandé), conforme à la norme RFC 3986. Les expressions de chaînage sont autorisées.	String	Oui	N/A
`destination`	Un chemin de fichier ou de dossier absolu ou relatif sur le système local. Les chemins de dossier doivent se terminer par`/`. S'ils ne se terminent pas par`/`, ils seront traités comme des chemins de fichiers. Le module crée tout fichier ou dossier requis pour des téléchargements réussis. Les expressions de	String	Oui	N/A

Nom de la touche	Description	Type	Obligatoire	Par défaut
	chaînage sont autorisées.			

Nom de la touche	Description	Type	Obligatoire	Par défaut
`overwrite`	Lorsque cette option est activée, elle remplace tous les fichiers existants sur le système local par le fichier ou la ressource téléchargé. Lorsque cette option n'est pas activée, aucun fichier existant sur le système local n'est remplacé et le module d'action échoue avec une erreur. Lorsque le remplacement est activé et que la somme de contrôle et l'algorithme sont spécifiés, le module d'action télécharge le fichier uniquement si la somme de contrôle et le hachage des	Booléen	Non	`true`

Nom de la touche	Description	Type	Obligatoire	Par défaut
	fichiers préexistants ne correspondent pas.			
`checksum`	Lorsque vous spécifiez la somme de contrôle, elle est comparée au hachage du fichier téléchargé généré avec l'algorithme fourni. Pour que la vérification des fichiers soit activée, la somme de contrôle et l'algorithme doivent être fournis. Les expressions de chaînage sont autorisées.	String	Non	N/A

Nom de la touche	Description	Type	Obligatoire	Par défaut
`algorithm`	Algorithme utilisé pour calculer le checksum. Les options sont MD5, SHA1, SHA256 et SHA512. Pour que la vérification des fichiers soit activée, la somme de contrôle et l'algorithme doivent être fournis. Les expressions de chaînage sont autorisées.	String	Non	N/A
`ignoreCertificateErrors`	La validation du certificat SSL est ignorée lorsqu'elle est activée.	Booléen	Non	`false`

Output

Nom de la touche	Description	Type
`destination`	Chaîne de nouvelle ligne délimitée par des	String

Nom de la touche	Description	Type				
	caractères qui indique le chemin de destination où sont stockés les fichiers ou les ressources téléchargés.					

Exemple de saisie : téléchargement d'un fichier distant vers une destination locale

```
  - name: DownloadRemoteFile
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: C:\testfolder\package.zip
```

Sortie :

```
{
 "destination": "C:\\testfolder\\package.zip"
}
```

Exemple de saisie : téléchargement de plusieurs fichiers distants vers plusieurs destinations locales

```
  - name: DownloadRemoteFiles
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: /tmp/java14_renamed.zip
      - source: https://testdomain/path/to/java14.zip
```

```
        destination: /tmp/create_new_folder_and_add_java14_as_zip/
```

Sortie :

```
{
 "destination": "/tmp/create_new_folder/java14_renamed.zip\n/tmp/
create_new_folder_and_add_java14_as_zip/java14.zip"
}
```

Exemple de saisie : téléchargement d'un fichier distant sans remplacer la destination locale et téléchargement d'un autre fichier distant avec vérification du fichier

```
  - name: DownloadRemoteMultipleProperties
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: C:\create_new_folder\java14_renamed.zip
        overwrite: false
      - source: https://testdomain/path/to/java14.zip
        destination: C:\create_new_folder_and_add_java14_as_zip\
        checksum: ac68bbf921d953d1cfab916cb6120864
        algorithm: MD5
        overwrite: true
```

Sortie :

```
{
 "destination": "C:\\create_new_folder\\java14_renamed.zip\nC:\
\create_new_folder_and_add_java14_as_zip\\java14.zip"
}
```

Exemple de saisie : télécharger un fichier distant et ignorer la validation de la certification SSL

```
  - name: DownloadRemoteIgnoreValidation
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://www.bad-ssl.com/resource
        destination: /tmp/downloads/
        ignoreCertificateErrors: true
```

Sortie :

```
{
 "destination": "/tmp/downloads/resource"
}
```

Modules d'opérations du système de fichiers

La section suivante contient des informations détaillées sur les modules d'action qui exécutent des opérations sur le système de fichiers.

Modules d'action relatifs au fonctionnement du système de fichiers

- AppendFile (Linux, Windows, macOS)
- CopyFile (Linux, Windows, macOS)
- CopyFolder (Linux, Windows, macOS)
- CreateFile (Linux, Windows, macOS)
- CreateFolder (Linux, Windows, macOS)
- CreateSymlink (Linux, Windows, macOS)
- DeleteFile (Linux, Windows, macOS)
- DeleteFolder (Linux, Windows, macOS)
- ListFiles (Linux, Windows, macOS)
- MoveFile (Linux, Windows, macOS)
- MoveFolder (Linux, Windows, macOS)
- ReadFile (Linux, Windows, macOS)
- SetFileEncoding (Linux, Windows, macOS)
- SetFileOwner (Linux, Windows, macOS)
- SetFolderOwner (Linux, Windows, macOS)
- SetFilePermissions (Linux, Windows, macOS)
- SetFolderPermissions (Linux, Windows, macOS)

AppendFile (Linux, Windows, macOS)

Le module AppendFiled'action ajoute le contenu spécifié au contenu préexistant d'un fichier.

Si la valeur de codage de fichier est différente de la valeur d'encodage (`utf-8`) par défaut, vous pouvez spécifier la valeur de codage de fichier à l'aide de l'`encoding`option. Par défaut, `utf-16` et `utf-32` sont supposés utiliser le codage little-endian.

Le module d'action renvoie une erreur dans les cas suivants :

- Le fichier spécifié n'existe pas au moment de l'exécution.
- Vous n'êtes pas autorisé à écrire pour modifier le contenu du fichier.
- Le module rencontre une erreur lors de l'opération sur le fichier.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Oui
`content`	Le contenu à ajouter au fichier.	String	Non	Chaîne vide	N/A	Oui
`encoding`	La norme de codage.	String	Non	`utf8`	`utf8`,`utf-8`, `utf-16-LE`, `utf16-BE`,`utf-16-BE`, `utf32`,`utf-LE`,`utf-32-LE`, `utf32-BE`, et `utf-32-`	Oui

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
					BE . La valeur de l'option de codage ne distingue pas les majuscules et minuscules.	

Exemple de saisie : ajout d'un fichier sans encodage (Linux)

```
- name: AppendingFileWithOutEncodingLinux
  action: AppendFile
  inputs:
    - path: ./Sample.txt
      content: "The string to be appended to the file"
```

Exemple de saisie : ajout d'un fichier sans encodage (Windows)

```
- name: AppendingFileWithOutEncodingWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolder\MyFile.txt
      content: "The string to be appended to the file"
```

Exemple de saisie : ajout d'un fichier avec encodage (Linux)

```
- name: AppendingFileWithEncodingLinux
  action: AppendFile
  inputs:
```

```
      - path: /FolderName/SampleFile.txt
        content: "The string to be appended to the file"
        encoding: UTF-32
```

Exemple de saisie : ajout d'un fichier avec encodage (Windows)

```
- name: AppendingFileWithEncodingWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolderName\SampleFile.txt
      content: "The string to be appended to the file"
      encoding: UTF-32
```

Exemple de saisie : ajouter un fichier avec une chaîne vide (Linux)

```
- name: AppendingEmptyStringLinux
  action: AppendFile
  inputs:
    - path: /FolderName/SampleFile.txt
```

Exemple de saisie : ajouter un fichier avec une chaîne vide (Windows)

```
- name: AppendingEmptyStringWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolderName\SampleFile.txt
```

Output

Aucune.

CopyFile (Linux, Windows, macOS)

Le module CopyFiled'action copie les fichiers de la source spécifiée vers la destination spécifiée. Par défaut, le module crée de manière récursive le dossier de destination s'il n'existe pas au moment de l'exécution.

Si un fichier portant le nom spécifié existe déjà dans le dossier spécifié, le module d'action remplace par défaut le fichier existant. Vous pouvez annuler ce comportement par défaut en définissant l'option de remplacement sur. `false` Lorsque l'option de remplacement est définie sur et qu'il existe déjà

un fichier portant le nom spécifié à l'emplacement spécifié, le module d'action renvoie une erreur. `false` Cette option fonctionne de la même manière que la `cp` commande sous Linux, qui remplace par défaut.

Le nom du fichier source peut inclure un caractère générique (`*`). Les caractères génériques ne sont acceptés qu'après le dernier séparateur de chemin de fichier (`/`ou`\`). Si des caractères génériques sont inclus dans le nom du fichier source, tous les fichiers correspondant au caractère générique sont copiés dans le dossier de destination. Si vous souhaitez déplacer plusieurs fichiers à l'aide d'un caractère générique, la saisie de l'`destination`option doit se terminer par un séparateur de chemin de fichier (`/`ou`\`), ce qui indique que l'entrée de destination est un dossier.

Si le nom du fichier de destination est différent du nom du fichier source, vous pouvez spécifier le nom du fichier de destination à l'aide de l'`destination`option. Si vous ne spécifiez pas de nom de fichier de destination, le nom du fichier source est utilisé pour créer le fichier de destination. Tout texte qui suit le dernier séparateur de chemin de fichier (`/`ou`\`) est traité comme nom de fichier. Si vous souhaitez utiliser le même nom de fichier que le fichier source, l'entrée de l'`destination`option doit se terminer par un séparateur de chemin de fichier (`/`ou`\`).

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à créer un fichier dans le dossier indiqué.
- Les fichiers source n'existent pas au moment de l'exécution.
- Il existe déjà un dossier portant le nom de fichier spécifié et l'`overwrite`option est définie sur`false`.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`source`	Le chemin du fichier source.	String	Oui	N/A	N/A	Oui

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`destination`	Le chemin du fichier de destination.	String	Oui	N/A	N/A	Oui
`overwrite`	Lorsqu'il est défini sur false, les fichiers de destination ne seront pas remplacés s'il existe déjà un fichier portant le nom spécifié à l'emplacement spécifié.	Booléen	Non	`true`	N/A	Oui

Exemple de saisie : copier un fichier (Linux)

```
- name: CopyingAFileLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
```

Exemple de saisie : copier un fichier (Windows)

```
- name: CopyingAFileWindows
  action: CopyFile
  inputs:
    - source: C:\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
```

Exemple de saisie : copier un fichier en utilisant le nom du fichier source (Linux)

```
- name: CopyingFileWithSourceFileNameLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/
```

Exemple de saisie : copier un fichier en utilisant le nom du fichier source (Windows)

```
- name: CopyingFileWithSourceFileNameWindows
  action: CopyFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder\
```

Exemple de saisie : copie d'un fichier à l'aide du caractère générique (Linux)

```
- name: CopyingFilesWithWildCardLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Exemple de saisie : copie d'un fichier à l'aide du caractère générique (Windows)

```
- name: CopyingFilesWithWildCardWindows
  action: CopyFile
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

Exemple de saisie : copier un fichier sans le remplacer (Linux)

```
- name: CopyingFilesWithoutOverwriteLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
      overwrite: false
```

Exemple de saisie : copier un fichier sans le remplacer (Windows)

```
- name: CopyingFilesWithoutOverwriteWindows
  action: CopyFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
      overwrite: false
```

Output

Aucune.

CopyFolder (Linux, Windows, macOS)

Le module CopyFolderd'action copie un dossier de la source spécifiée vers la destination spécifiée. L'entrée de l'`source`option est le dossier à copier, et l'entrée de l'`destination`option est le dossier dans lequel le contenu du dossier source est copié. Par défaut, le module crée de manière récursive le dossier de destination s'il n'existe pas au moment de l'exécution.

Si un dossier portant le nom spécifié existe déjà dans le dossier spécifié, le module d'action remplace par défaut le dossier existant. Vous pouvez annuler ce comportement par défaut en définissant l'option de remplacement sur. `false` Lorsque l'option de remplacement est définie sur et qu'il existe déjà un dossier portant le nom spécifié à l'emplacement spécifié, le module d'action renvoie une erreur. `false`

Le nom du dossier source peut inclure un caractère générique (`*`). Les caractères génériques ne sont acceptés qu'après le dernier séparateur de chemin de fichier (`/`ou`\`). Si des caractères génériques sont inclus dans le nom du dossier source, tous les dossiers correspondant au caractère générique sont copiés dans le dossier de destination. Si vous souhaitez copier plusieurs dossiers à l'aide d'un caractère générique, la saisie de l'`destination`option doit se terminer par un séparateur de chemin de fichier (`/`ou`\`), ce qui indique que l'entrée de destination est un dossier.

Si le nom du dossier de destination est différent du nom du dossier source, vous pouvez spécifier le nom du dossier de destination à l'aide de l'`destination`option. Si vous ne spécifiez pas de nom de dossier de destination, le nom du dossier source est utilisé pour créer le dossier de destination. Tout texte qui suit le dernier séparateur de chemin de fichier (`/`ou`\`) est traité comme le nom du dossier. Si vous souhaitez utiliser le même nom de dossier que le dossier source, l'entrée de l'`destination`option doit se terminer par un séparateur de chemin de fichier (`/`ou`\`).

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à créer un dossier dans le dossier spécifié.
- Les dossiers source n'existent pas au moment de l'exécution.
- Il existe déjà un dossier portant le nom de dossier spécifié et l'`overwrite`option est définie sur`false`.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`source`	Le chemin du dossier source.	String	Oui	N/A	N/A	Oui
`destination`	Le chemin du dossier de destination.	String	Oui	N/A	N/A	Oui
`overwrite`	Lorsqu'il est défini sur false, les dossiers de destinati	Booléen	Non	`true`	N/A	Oui

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
	on ne seront pas remplacés s'il existe déjà un dossier portant le nom spécifié à l'emplacement spécifié.					

Exemple de saisie : copier un dossier (Linux)

```
- name: CopyingAFolderLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
      destination: /MyFolder/destinationFolder
```

Exemple de saisie : copier un dossier (Windows)

```
- name: CopyingAFolderWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\destinationFolder
```

Exemple de saisie : copier un dossier en utilisant le nom du dossier source (Linux)

```
- name: CopyingFolderSourceFolderNameLinux
```

```
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/
```

Exemple de saisie : copier un dossier en utilisant le nom du dossier source (Windows)

```
- name: CopyingFolderSourceFolderNameWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\
```

Exemple de saisie : copier un dossier à l'aide du caractère générique (Linux)

```
- name: CopyingFoldersWithWildCardLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Exemple de saisie : copier un dossier à l'aide du caractère générique (Windows)

```
- name: CopyingFoldersWithWildCardWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

Exemple de saisie : copier un dossier sans le remplacer (Linux)

```
- name: CopyingFoldersWithoutOverwriteLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/destinationFolder
      overwrite: false
```

Exemple de saisie : copier un dossier sans le remplacer (Windows)

```
- name: CopyingFoldersWithoutOverwrite
```

```
    action: CopyFolder
    inputs:
      - source: C:\Sample\MyFolder\SourceFolder
        destination: C:\MyFolder\destinationFolder
        overwrite: false
```

Output

Aucune.

CreateFile (Linux, Windows, macOS)

Le module CreateFiled'action crée un fichier dans un emplacement spécifié. Par défaut, si nécessaire, le module crée également les dossiers parents de manière récursive.

Si le fichier existe déjà dans le dossier spécifié, le module d'action tronque ou remplace par défaut le fichier existant. Vous pouvez annuler ce comportement par défaut en définissant l'option de remplacement sur. `false` Lorsque l'option de remplacement est définie sur et qu'il existe déjà un fichier portant le nom spécifié à l'emplacement spécifié, le module d'action renvoie une erreur. `false`

Si la valeur de codage de fichier est différente de la valeur d'encodage (`utf-8`) par défaut, vous pouvez spécifier la valeur de codage de fichier à l'aide de l'`encoding`option. Par défaut, `utf-16` et `utf-32` sont supposés utiliser le codage little-endian.

`owner`group, et `permissions` sont des entrées facultatives. L'entrée pour `permissions` doit être une valeur de chaîne. Les fichiers sont créés avec des valeurs par défaut lorsqu'elles ne sont pas fournies. Ces options ne sont pas prises en charge sur les plateformes Windows. Ce module d'action valide et renvoie une erreur si les `permissions` options `owner`group, et sont utilisées sur les plateformes Windows.

Ce module d'action peut créer un fichier dont les autorisations sont définies par la `umask` valeur par défaut du système d'exploitation. Vous devez définir la `umask` valeur si vous souhaitez remplacer la valeur par défaut.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à créer un fichier ou un dossier dans le dossier parent spécifié.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Oui
`content`	Le contenu textuel du fichier.	String	Non	N/A	N/A	Oui
`encoding`	La norme de codage.	String	Non	`utf8`	`utf8`,`utf-8`, `utf-16-LE`, `utf16-BE`utf-16-BE ,`utf32`,`utfLE`,`utf-32-LE` ,`utf32-BE`, et `utf-32-BE` . La valeur de l'option de codage ne distingue pas les majuscules et	Oui

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
					minuscules.	
`owner`	Le nom d'utilisateur ou l'ID.	String	Non	N/A	N/A	Non pris en charge sous Windows.
`group`	Le nom ou l'identifiant du groupe.	String	Non	L'utilisateur actuel.	N/A	Non pris en charge sous Windows.
`permissions`	Les autorisations du fichier.	String	Non	`0666`	N/A	Non pris en charge sous Windows.
`overwrite`	Si le nom du fichier spécifié existe déjà, définissez cette valeur pour `false` empêcher le fichier d'être tronqué ou remplacé par défaut.	Booléen	Non	`true`	N/A	Oui

Exemple de saisie : créer un fichier sans le remplacer (Linux)

```
- name: CreatingFileWithoutOverwriteLinux
  action: CreateFile
  inputs:
    - path: /home/UserName/Sample.txt
      content: The text content of the sample file.
      overwrite: false
```

Exemple de saisie : créer un fichier sans le remplacer (Windows)

```
- name: CreatingFileWithoutOverwriteWindows
  action: CreateFile
  inputs:
    - path: C:\Temp\Sample.txt
      content: The text content of the sample file.
      overwrite: false
```

Exemple de saisie : création d'un fichier avec des propriétés de fichier

```
- name: CreatingFileWithFileProperties
  action: CreateFile
  inputs:
    - path: SampleFolder/Sample.txt
      content: The text content of the sample file.
      encoding: UTF-16
      owner: Ubuntu
      group: UbuntuGroup
      permissions: 0777
   - path: SampleFolder/SampleFile.txt
      permissions: 755
    - path: SampleFolder/TextFile.txt
      encoding: UTF-16
      owner: root
      group: rootUserGroup
```

Exemple de saisie : création d'un fichier sans propriétés de fichier

```
- name: CreatingFileWithoutFileProperties
  action: CreateFile
  inputs:
```

```
      - path: ./Sample.txt
      - path: Sample1.txt
```

Exemple de saisie : créer un fichier vide pour ignorer une section du script de nettoyage Linux

```
- name: CreateSkipCleanupfile
  action: CreateFile
  inputs:
    - path: <skip section file name>
```

Pour de plus amples informations, consultez Remplacez le script de nettoyage de Linux.

Output

Aucune.

CreateFolder (Linux, Windows, macOS)

Le module CreateFolderd'action crée un dossier à un emplacement spécifié. Par défaut, si nécessaire, le module crée également les dossiers parents de manière récursive.

Si le dossier existe déjà dans le dossier spécifié, le module d'action tronque ou remplace par défaut le dossier existant. Vous pouvez annuler ce comportement par défaut en définissant l'option de remplacement sur. `false` Lorsque l'option de remplacement est définie sur et qu'il existe déjà un dossier portant le nom spécifié à l'emplacement spécifié, le module d'action renvoie une erreur. `false`

`ownergroup`, et `permissions` sont des entrées facultatives. L'entrée pour `permissions` doit être une valeur de chaîne. Ces options ne sont pas prises en charge sur les plateformes Windows. Ce module d'action valide et renvoie une erreur si les `permissions` options `ownergroup`, et sont utilisées sur les plateformes Windows.

Ce module d'action peut créer un dossier dont les autorisations sont définies par la `umask` valeur par défaut du système d'exploitation. Vous devez définir la `umask` valeur si vous souhaitez remplacer la valeur par défaut.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à créer un dossier à l'emplacement indiqué.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du dossier.	String	Oui	N/A	N/A	Oui
`owner`	Le nom d'utilisateur ou l'ID.	String	Non	L'utilisateur actuel.	N/A	Non pris en charge sous Windows.
`group`	Le nom ou l'identifiant du groupe.	String	Non	Le groupe de l'utilisateur actuel.	N/A	Non pris en charge sous Windows.
`permissions`	Les autorisations relatives aux dossiers.	String	Non	`0777`	N/A	Non pris en charge sous Windows.
`overwrite`	Si le nom du fichier spécifié existe déjà, définissez cette valeur pour `false` empêcher le fichier	Booléen	Non	`true`	N/A	Oui

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
	d'être tronqué ou remplacé par défaut.					

Exemple de saisie : créer un dossier (Linux)

```
- name: CreatingFolderLinux
  action: CreateFolder
  inputs:
    - path: /Sample/MyFolder/
```

Exemple de saisie : créer un dossier (Windows)

```
- name: CreatingFolderWindows
  action: CreateFolder
  inputs:
    - path: C:\MyFolder
```

Exemple de saisie : création d'un dossier en spécifiant les propriétés du dossier

```
- name: CreatingFolderWithFolderProperties
  action: CreateFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
      owner: SampleOwnerName
      group: SampleGroupName
      permissions: 0777
    - path: /Sample/MyFolder/SampleFoler/
      permissions: 777
```

Exemple de saisie : créer un dossier qui remplace le dossier existant, s'il y en a un.

```
- name: CreatingFolderWithOverwrite
  action: CreateFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
      overwrite: true
```

Output

Aucune.

CreateSymlink (Linux, Windows, macOS)

Le module CreateSymlinkd'action crée des liens symboliques ou des fichiers contenant une référence à un autre fichier. Ce module n'est pas pris en charge sur les plateformes Windows.

L'entrée pour les `target` options `path` et peut être un chemin absolu ou relatif. Si l'entrée de l'`path`option est un chemin relatif, il est remplacé par le chemin absolu lors de la création du lien.

Par défaut, lorsqu'un lien portant le nom spécifié existe déjà dans le dossier spécifié, le module d'action renvoie une erreur. Vous pouvez annuler ce comportement par défaut en définissant l'`force`option sur. `true` Lorsque l'`force`option est définie sur`true`, le module remplace le lien existant.

Si aucun dossier parent n'existe, le module d'action crée le dossier de manière récursive, par défaut.

Le module d'action renvoie une erreur dans les cas suivants :

- Le fichier cible n'existe pas au moment de l'exécution.
- Un fichier de lien non symbolique portant le nom spécifié existe déjà.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Non pris en charge sous Windows.
`target`	Le chemin du fichier cible vers lequel pointe le lien symbolique.	String	Oui	N/A	N/A	Non pris en charge sous Windows.
`force`	Force la création d'un lien lorsqu'un lien portant le même nom existe déjà.	Booléen	Non	`false`	N/A	Non pris en charge sous Windows.

Exemple de saisie : créer un lien symbolique qui force la création d'un lien

```
- name: CreatingSymbolicLinkWithForce
  action: CreateSymlink
  inputs:
    - path: /Folder2/Symboliclink.txt
      target: /Folder/Sample.txt
```

```
        force: true
```

Exemple de saisie : créer un lien symbolique qui ne force pas la création d'un lien

```
- name: CreatingSymbolicLinkWithOutForce
  action: CreateSymlink
  inputs:
    - path: Symboliclink.txt
      target: /Folder/Sample.txt
```

Output

Aucune.

DeleteFile (Linux, Windows, macOS)

Le module DeleteFiled'action supprime un ou plusieurs fichiers dans un emplacement spécifié.

L'entrée de `path` doit être un chemin de fichier valide ou un chemin de fichier avec un caractère générique (*) dans le nom du fichier. Lorsque des caractères génériques sont spécifiés dans le nom du fichier, tous les fichiers du même dossier qui correspondent au caractère générique sont supprimés.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à effectuer des opérations de suppression.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Oui

Exemple de saisie : supprimer un seul fichier (Linux)

```
- name: DeletingSingleFileLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/Sample.txt
```

Exemple de saisie : supprimer un seul fichier (Windows)

```
- name: DeletingSingleFileWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\Sample.txt
```

Exemple de saisie : supprimer un fichier qui se termine par « log » (Linux)

```
- name: DeletingFileEndingWithLogLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/*log
```

Exemple de saisie : supprimer un fichier qui se termine par « log » (Windows)

```
- name: DeletingFileEndingWithLogWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\*log
```

Exemple de saisie : supprimer tous les fichiers d'un dossier spécifié (Linux)

```
- name: DeletingAllFilesInAFolderLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/*
```

Exemple de saisie : supprimer tous les fichiers d'un dossier spécifié (Windows)

```
- name: DeletingAllFilesInAFolderWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\*
```

Output

Aucune.

DeleteFolder (Linux, Windows, macOS)

Le module DeleteFolderd'action supprime les dossiers.

Si le dossier n'est pas vide, vous devez définir l'`force`option `true` de suppression du dossier et de son contenu. Si vous ne définissez pas l'`force`option sur `true` et que le dossier que vous essayez de supprimer n'est pas vide, le module d'action renvoie une erreur. La valeur par défaut de l'`force`option est`false`.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à effectuer des opérations de suppression.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du dossier.	String	Oui	N/A	N/A	Oui
`force`	Supprime le dossier, qu'il soit vide ou non.	Booléen	Non	`false`	N/A	Oui

Exemple de saisie : supprimer un dossier qui n'est pas vide à l'aide de l'**`force`**option (Linux)

```
  - name: DeletingFolderWithForceOptionLinux
    action: DeleteFolder
```

```
  inputs:
    - path: /Sample/MyFolder/Sample/
      force: true
```

Exemple de saisie : supprimer un dossier qui n'est pas vide à l'aide de l'**`force`**option (Windows)

```
- name: DeletingFolderWithForceOptionWindows
  action: DeleteFolder
  inputs:
    - path: C:\Sample\MyFolder\Sample\
      force: true
```

Exemple de saisie : suppression d'un dossier (Linux)

```
- name: DeletingFolderWithOutForceLinux
  action: DeleteFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
```

Exemple de saisie : suppression d'un dossier (Windows)

```
- name: DeletingFolderWithOutForce
  action: DeleteFolder
  inputs:
    - path: C:\Sample\MyFolder\Sample\
```

Output

Aucune.

ListFiles (Linux, Windows, macOS)

Le module ListFilesd'action répertorie les fichiers d'un dossier spécifié. Lorsque l'option récursive est définie sur`true`, elle répertorie les fichiers dans des sous-dossiers. Ce module ne répertorie pas les fichiers dans les sous-dossiers par défaut.

Pour répertorier tous les fichiers dont les noms correspondent à un modèle spécifié, utilisez l'`fileNamePattern`option permettant de fournir le modèle. L'`fileNamePattern`option accepte la valeur wildcard (`*`). Lorsque le `fileNamePattern` est fourni, tous les fichiers correspondant au format de nom de fichier spécifié sont renvoyés.

Le module d'action renvoie une erreur dans les cas suivants :

- Le dossier spécifié n'existe pas au moment de l'exécution.
- Vous n'êtes pas autorisé à créer un fichier ou un dossier dans le dossier parent spécifié.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du dossier.	String	Oui	N/A	N/A	Oui
`fileNamePattern`	Le modèle à associer pour répertorier tous les fichiers dont les noms correspondent au modèle.	String	Non	N/A	N/A	Oui
`recursive`	Répertorie les fichiers du dossier de manière récursive.	Booléen	Non	`false`	N/A	Oui

Exemple de saisie : liste des fichiers dans le dossier spécifié (Linux)

```
  - name: ListingFilesInSampleFolderLinux
    action: ListFiles
```

```
    inputs:
      - path: /Sample/MyFolder/Sample
```

Exemple de saisie : liste des fichiers dans le dossier spécifié (Windows)

```
- name: ListingFilesInSampleFolderWindows
  action: ListFiles
  inputs:
    - path: C:\Sample\MyFolder\Sample
```

Exemple de saisie : liste les fichiers qui se terminent par « log » (Linux)

```
- name: ListingFilesWithEndingWithLogLinux
  action: ListFiles
  inputs:
    - path: /Sample/MyFolder/
      fileNamePattern: *log
```

Exemple de saisie : liste les fichiers qui se terminent par « log » (Windows)

```
- name: ListingFilesWithEndingWithLogWindows
  action: ListFiles
  inputs:
    - path: C:\Sample\MyFolder\
      fileNamePattern: *log
```

Exemple de saisie : liste des fichiers de manière récursive

```
- name: ListingFilesRecursively
  action: ListFiles
  inputs:
    - path: /Sample/MyFolder/
      recursive: true
```

Output

Nom de la touche	Description	Type
`files`	La liste des fichiers.	String

Exemple de sortie

```
{
 "files": "/sample1.txt,/sample2.txt,/sample3.txt"
}
```

MoveFile (Linux, Windows, macOS)

Le module MoveFiled'action déplace les fichiers de la source spécifiée vers la destination spécifiée.

Si le fichier existe déjà dans le dossier spécifié, le module d'action remplace par défaut le fichier existant. Vous pouvez annuler ce comportement par défaut en définissant l'option de remplacement sur. `false` Lorsque l'option de remplacement est définie sur et qu'il existe déjà un fichier portant le nom spécifié à l'emplacement spécifié, le module d'action renvoie une erreur. `false` Cette option fonctionne de la même manière que la `mv` commande sous Linux, qui remplace par défaut.

Le nom du fichier source peut inclure un caractère générique (*). Les caractères génériques ne sont acceptés qu'après le dernier séparateur de chemin de fichier (/ou\). Si des caractères génériques sont inclus dans le nom du fichier source, tous les fichiers correspondant au caractère générique sont copiés dans le dossier de destination. Si vous souhaitez déplacer plusieurs fichiers à l'aide d'un caractère générique, la saisie de l'`destination`option doit se terminer par un séparateur de chemin de fichier (/ou\), ce qui indique que l'entrée de destination est un dossier.

Si le nom du fichier de destination est différent du nom du fichier source, vous pouvez spécifier le nom du fichier de destination à l'aide de l'`destination`option. Si vous ne spécifiez pas de nom de fichier de destination, le nom du fichier source est utilisé pour créer le fichier de destination. Tout texte qui suit le dernier séparateur de chemin de fichier (/ou\) est traité comme nom de fichier. Si vous souhaitez utiliser le même nom de fichier que le fichier source, l'entrée de l'`destination`option doit se terminer par un séparateur de chemin de fichier (/ou\).

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à créer un fichier dans le dossier indiqué.
- Les fichiers source n'existent pas au moment de l'exécution.
- Il existe déjà un dossier portant le nom de fichier spécifié et l'`overwrite`option est définie sur`false`.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`source`	Le chemin du fichier source.	String	Oui	N/A	N/A	Oui
`destination`	Le chemin du fichier de destination.	String	Oui	N/A	N/A	Oui
`overwrite`	Lorsqu'il est défini sur false, les fichiers de destination ne seront pas remplacés s'il existe déjà un fichier portant le nom spécifié à l'emplacement spécifié.	Booléen	Non	`true`	N/A	Oui

Exemple de saisie : déplacer un fichier (Linux)

```
- name: MovingAFileLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
```

Exemple de saisie : déplacer un fichier (Windows)

```
- name: MovingAFileWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
```

Exemple de saisie : déplacer un fichier en utilisant le nom du fichier source (Linux)

```
- name: MovingFileWithSourceFileNameLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/
```

Exemple de saisie : déplacer un fichier en utilisant le nom du fichier source (Windows)

```
- name: MovingFileWithSourceFileNameWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder
```

Exemple de saisie : déplacer un fichier à l'aide d'un caractère générique (Linux)

```
- name: MovingFilesWithWildCardLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Exemple de saisie : déplacer un fichier à l'aide d'un caractère générique (Windows)

```
- name: MovingFilesWithWildCardWindows
```

```
    action: MoveFile
    inputs:
      - source: C:\Sample\MyFolder\Sample*
        destination: C:\MyFolder
```

Exemple de saisie : déplacer un fichier sans le remplacer (Linux)

```
  - name: MovingFilesWithoutOverwriteLinux
    action: MoveFile
    inputs:
      - source: /Sample/MyFolder/Sample.txt
        destination: /MyFolder/destinationFile.txt
        overwrite: false
```

Exemple de saisie : déplacer un fichier sans le remplacer (Windows)

```
  - name: MovingFilesWithoutOverwrite
    action: MoveFile
    inputs:
      - source: C:\Sample\MyFolder\Sample.txt
        destination: C:\MyFolder\destinationFile.txt
        overwrite: false
```

Output

Aucune.

MoveFolder (Linux, Windows, macOS)

Le module MoveFolderd'action déplace les dossiers de la source spécifiée vers la destination spécifiée. L'entrée de l'`source`option est le dossier à déplacer, et l'entrée de l'`destination`option est le dossier dans lequel le contenu des dossiers source est déplacé.

Si le dossier parent de destination ou l'entrée de l'`destination`option n'existe pas au moment de l'exécution, le comportement par défaut du module consiste à créer le dossier de manière récursive à la destination spécifiée.

Si un dossier identique au dossier source existe déjà dans le dossier de destination, le module d'action remplace par défaut le dossier existant. Vous pouvez annuler ce comportement par défaut en définissant l'option de remplacement sur. `false` Lorsque l'option de remplacement est définie sur et qu'il existe déjà un dossier portant le nom spécifié à l'emplacement spécifié, le module d'action renvoie une erreur. `false`

Le nom du dossier source peut inclure un caractère générique (*). Les caractères génériques ne sont acceptés qu'après le dernier séparateur de chemin de fichier (/ou\). Si des caractères génériques sont inclus dans le nom du dossier source, tous les dossiers correspondant au caractère générique sont copiés dans le dossier de destination. Si vous souhaitez déplacer plusieurs dossiers à l'aide d'un caractère générique, la saisie de l'`destination`option doit se terminer par un séparateur de chemin de fichier (/ou\), ce qui indique que l'entrée de destination est un dossier.

Si le nom du dossier de destination est différent du nom du dossier source, vous pouvez spécifier le nom du dossier de destination à l'aide de l'`destination`option. Si vous ne spécifiez pas de nom de dossier de destination, le nom du dossier source est utilisé pour créer le dossier de destination. Tout texte qui suit le dernier séparateur de chemin de fichier (/ou\) est traité comme le nom du dossier. Si vous souhaitez utiliser le même nom de dossier que le dossier source, l'entrée de l'`destination`option doit se terminer par un séparateur de chemin de fichier (/ou\).

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à créer un dossier dans le dossier de destination.
- Les dossiers source n'existent pas au moment de l'exécution.
- Il existe déjà un dossier portant le nom spécifié et l'`overwrite`option est définie sur`false`.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`source`	Le chemin du dossier source.	String	Oui	N/A	N/A	Oui
`destination`	Le chemin du dossier de destination.	String	Oui	N/A	N/A	Oui

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`overwrite`	Lorsqu'il est défini sur false, les dossiers de destination ne seront pas remplacés s'il existe déjà un dossier portant le nom spécifié à l'emplacement spécifié.	Booléen	Non	`true`	N/A	Oui

Exemple de saisie : déplacer un dossier (Linux)

```
- name: MovingAFolderLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/destinationFolder
```

Exemple de saisie : déplacer un dossier (Windows)

```
- name: MovingAFolderWindows
  action: MoveFolder
```

```
    inputs:
      - source: C:\Sample\MyFolder\SourceFolder
        destination: C:\MyFolder\destinationFolder
```

Exemple de saisie : déplacer un dossier en utilisant le nom du dossier source (Linux)

```
- name: MovingFolderWithSourceFolderNameLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
      destination: /MyFolder/
```

Exemple de saisie : déplacer un dossier en utilisant le nom du dossier source (Windows)

```
- name: MovingFolderWithSourceFolderNameWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\
```

Exemple de saisie : déplacer un dossier à l'aide d'un caractère générique (Linux)

```
- name: MovingFoldersWithWildCardLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Exemple de saisie : déplacer un dossier à l'aide d'un caractère générique (Windows)

```
- name: MovingFoldersWithWildCardWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

Exemple de saisie : déplacer un dossier sans le remplacer (Linux)

```
- name: MovingFoldersWithoutOverwriteLinux
  action: MoveFolder
```

```
    inputs:
      - source: /Sample/MyFolder/SampleFolder
    destination: /MyFolder/destinationFolder
    overwrite: false
```

Exemple de saisie : déplacer un dossier sans le remplacer (Windows)

```
- name: MovingFoldersWithoutOverwriteWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\destinationFolder
      overwrite: false
```

Output

Aucune.

ReadFile (Linux, Windows, macOS)

Le module ReadFiled'action lit le contenu d'un fichier texte de type chaîne. Ce module peut être utilisé pour lire le contenu d'un fichier afin de l'utiliser dans les étapes suivantes via le chaînage ou pour lire des données dans le `console.log` fichier. Si le chemin spécifié est un lien symbolique, ce module renvoie le contenu du fichier cible. Ce module ne prend en charge que les fichiers texte.

Si la valeur de codage de fichier est différente de la valeur d'encodage (`utf-8`) par défaut, vous pouvez spécifier la valeur de codage de fichier à l'aide de l'`encoding`option. Par défaut, `utf-16` et `utf-32` sont supposés utiliser le codage little-endian.

Par défaut, ce module ne peut pas imprimer le contenu du fichier dans le `console.log` fichier. Vous pouvez annuler ce paramètre en attribuant à `true` la `printFileContent` propriété la valeur.

Ce module ne peut renvoyer que le contenu d'un fichier. Il ne peut pas analyser les fichiers, tels que les fichiers Excel ou JSON.

Le module d'action renvoie une erreur dans les cas suivants :

- Le fichier n'existe pas au moment de l'exécution.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Oui
`encoding`	La norme de codage.	String	Non	`utf8`	`utf8`,`utf-8-LE`, `utf-16-LE` `utf16-BE`utf-16-BE ,`utf32`,`utf-LE`,`utf-32-LE` ,`utf32-BE`, et `utf-32-BE` . La valeur de l'option de codage ne distingue pas les majuscules et minuscules.	Oui	6,utf1
`printFileContent`	Imprime le contenu du fichier	Booléen	Non	false	N/A	Oui.

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
	dans le `console.log` fichier.					

Exemple de saisie : lecture d'un fichier (Linux)

```
- name: ReadingFileLinux
  action: ReadFile
  inputs:
    - path: /home/UserName/SampleFile.txt
```

Exemple de saisie : lecture d'un fichier (Windows)

```
- name: ReadingFileWindows
  action: ReadFile
  inputs:
    - path: C:\Windows\WindowsUpdate.log
```

Exemple de saisie : lecture d'un fichier et spécification de la norme de codage

```
- name: ReadingFileWithFileEncoding
  action: ReadFile
  inputs:
    - path: /FolderName/SampleFile.txt
      encoding: UTF-32
```

Exemple de saisie : lecture d'un fichier et impression dans le **`console.log`** fichier

```
- name: ReadingFileToConsole
  action: ReadFile
  inputs:
    - path: /home/UserName/SampleFile.txt
      printFileContent: true
```

Output

Champ	Description	Type
`content`	Le contenu du fichier.	chaîne

Exemple de sortie

```
{
 "content" : "The file content"
}
```

SetFileEncoding (Linux, Windows, macOS)

Le module SetFileEncodingd'action modifie la propriété de codage d'un fichier existant. Ce module peut convertir le codage de fichiers `utf-8` à partir d'une norme de codage spécifiée. Par défaut, `utf-16` et `utf-32` sont supposés être des encodages little-endian.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à effectuer la modification spécifiée.
- Le fichier n'existe pas au moment de l'exécution.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Oui
`encoding`	La norme de codage.	String	Non	`utf8`	`utf8`,`utf-8-LE`, `utf-16-LE`	Oui

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
					`utf16-BE`utf-16-BE ,`utf32`,`utfLE`,`utf-32-LE` ,`utf32-BE`, et `utf-32-BE` . La valeur de l'option de codage ne distingue pas les majuscules et minuscules.	

Exemple de saisie : définir la propriété de codage du fichier

```
- name: SettingFileEncodingProperty
  action: SetFileEncoding
  inputs:
    - path: /home/UserName/SampleFile.txt
      encoding: UTF-16
```

Output

Aucune.

SetFileOwner (Linux, Windows, macOS)

Le module SetFileOwnerd'action modifie les propriétés `owner` et le `group` propriétaire d'un fichier existant. Si le fichier spécifié est un lien symbolique, le module modifie la `owner` propriété du fichier source. Ce module n'est pas pris en charge sur les plateformes Windows.

Ce module accepte les noms d'utilisateur et de groupe en entrée. Si le nom du groupe n'est pas fourni, le module affecte le propriétaire du fichier au groupe auquel appartient l'utilisateur.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à effectuer la modification spécifiée.
- Le nom d'utilisateur ou de groupe spécifié n'existe pas au moment de l'exécution.
- Le fichier n'existe pas au moment de l'exécution.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Non pris en charge sous Windows.
`owner`	Nom de l'utilisateur.	chaîne	Oui	N/A	N/A	Non pris en charge sous Windows.
`group`	Nom du groupe d'utilisateurs.	String	Non	Nom du groupe auquel	N/A	Non pris en charge sous Windows.

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
				appartient l'utilisateur.		

Exemple de saisie : définir la propriété du propriétaire du fichier sans spécifier le nom du groupe d'utilisateurs

```
- name: SettingFileOwnerPropertyNoGroup
  action: SetFileOwner
  inputs:
    - path: /home/UserName/SampleText.txt
      owner: LinuxUser
```

Exemple de saisie : définir la propriété du propriétaire du fichier en spécifiant le propriétaire et le groupe d'utilisateurs

```
- name: SettingFileOwnerProperty
  action: SetFileOwner
  inputs:
    - path: /home/UserName/SampleText.txt
      owner: LinuxUser
      group: LinuxUserGroup
```

Output

Aucune.

SetFolderOwner (Linux, Windows, macOS)

Le module SetFolderOwnerd'action modifie de manière récursive les propriétés `owner` et le `group` propriétaire d'un dossier existant. Par défaut, le module peut modifier la propriété de l'ensemble du contenu d'un dossier. Vous pouvez définir l'`recursive`option `false` pour annuler ce comportement. Ce module n'est pas pris en charge sur les plateformes Windows.

Ce module accepte les noms d'utilisateur et de groupe en entrée. Si le nom du groupe n'est pas fourni, le module affecte le propriétaire du fichier au groupe auquel appartient l'utilisateur.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à effectuer la modification spécifiée.
- Le nom d'utilisateur ou de groupe spécifié n'existe pas au moment de l'exécution.
- Le dossier n'existe pas au moment de l'exécution.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du dossier.	String	Oui	N/A	N/A	Non pris en charge sous Windows.
`owner`	Nom de l'utilisateur.	chaîne	Oui	N/A	N/A	Non pris en charge sous Windows.
`group`	Nom du groupe d'utilisateurs.	String	Non	Nom du groupe auquel appartient l'utilisateur.	N/A	Non pris en charge sous Windows.
`recursive`	Remplace le comportement par	Booléen	Non	`true`	N/A	Non pris en charge sous Windows.

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
	défaut qui consiste à modifier la propriété de l'ensemble du contenu d'un dossier lorsqu'il est défini sur. `false`					

Exemple de saisie : définir la propriété du propriétaire du dossier sans spécifier le nom du groupe d'utilisateurs

```
- name: SettingFolderPropertyWithOutGroup
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
```

Exemple de saisie : définir la propriété du propriétaire du dossier sans annuler la propriété de l'ensemble du contenu d'un dossier

```
- name: SettingFolderPropertyWithOutRecursively
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
      recursive: false
```

Exemple de saisie : définir la propriété de propriété du fichier en spécifiant le nom du groupe d'utilisateurs

```
- name: SettingFolderPropertyWithGroup
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
      group: LinuxUserGroup
```

Output

Aucune.

SetFilePermissions (Linux, Windows, macOS)

Le module SetFilePermissionsd'action modifie `permissions` un fichier existant. Ce module n'est pas pris en charge sur les plateformes Windows.

L'entrée pour `permissions` doit être une valeur de chaîne.

Ce module d'action peut créer un fichier dont les autorisations sont définies par la valeur umask par défaut du système d'exploitation. Vous devez définir la `umask` valeur si vous souhaitez remplacer la valeur par défaut.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à effectuer la modification spécifiée.
- Le fichier n'existe pas au moment de l'exécution.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du fichier.	String	Oui	N/A	N/A	Non pris en charge

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
						sous Windows.
`permissions`	Les autorisations du fichier.	String	Oui	N/A	N/A	Non pris en charge sous Windows.

Exemple de saisie : modification des autorisations de fichier

```
- name: ModifyingFilePermissions
  action: SetFilePermissions
  inputs:
    - path: /home/UserName/SampleFile.txt
      permissions: 766
```

Output

Aucune.

SetFolderPermissions (Linux, Windows, macOS)

Le module SetFolderPermissionsd'action modifie `permissions` de manière récursive le dossier existant et tous ses sous-fichiers et sous-dossiers. Par défaut, ce module peut modifier les autorisations pour tout le contenu du dossier spécifié. Vous pouvez définir l'`recursive`option `false` pour annuler ce comportement. Ce module n'est pas pris en charge sur les plateformes Windows.

L'entrée pour `permissions` doit être une valeur de chaîne.

Ce module d'action peut modifier les autorisations en fonction de la valeur umask par défaut du système d'exploitation. Vous devez définir la `umask` valeur si vous souhaitez remplacer la valeur par défaut.

Le module d'action renvoie une erreur dans les cas suivants :

- Vous n'êtes pas autorisé à effectuer la modification spécifiée.
- Le dossier n'existe pas au moment de l'exécution.
- Le module d'action rencontre une erreur lors de l'exécution de l'opération.

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
`path`	Le chemin du dossier.	String	Oui	N/A	N/A	Non pris en charge sous Windows.
`permissions`	Les autorisations relatives aux dossiers.	String	Oui	N/A	N/A	Non pris en charge sous Windows.
`recursive`	Remplace le comportement par défaut qui consiste à modifier les autorisations pour l'ensemble du contenu d'un dossier	Booléen	Non	`true`	N/A	Non pris en charge sous Windows.

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables	Pris en charge sur toutes les plateformes
	lorsqu'il est défini sur. `false`					

Exemple de saisie : définir les autorisations des dossiers

```
- name: SettingFolderPermissions
  action: SetFolderPermissions
  inputs:
    - path: SampleFolder/
      permissions: 0777
```

Exemple de saisie : définir les autorisations d'un dossier sans modifier les autorisations pour l'ensemble du contenu d'un dossier

```
- name: SettingFolderPermissionsNoRecursive
  action: SetFolderPermissions
  inputs:
    - path: /home/UserName/SampleFolder/
      permissions: 777
      recursive: false
```

Output

Aucune.

Actions d'installation du logiciel

La section suivante décrit les modules d'action qui installent ou désinstallent des logiciels.

Exigences relatives à l'IAM

Si le chemin de téléchargement de votre installation est un URI S3, le rôle IAM que vous associez à votre profil d'instance doit être autorisé à exécuter le module `S3Download` d'action. Pour accorder

l'autorisation requise, attachez la politique `S3:GetObject` IAM au rôle IAM associé à votre profil d'instance et spécifiez le chemin de votre bucket. Par exemple, *`arn:aws:s3:::BucketName/*`*).

Entrées MSI complexes

Si vos chaînes d'entrée contiennent des guillemets ("), vous devez utiliser l'une des méthodes suivantes pour vous assurer qu'elles sont interprétées correctement :

- Vous pouvez utiliser des guillemets simples (') à l'extérieur de votre chaîne pour la contenir, et des guillemets doubles («) à l'intérieur de votre chaîne, comme indiqué dans l'exemple suivant.

  ```
  properties:
    COMPANYNAME: '"Acme ""Widgets"" and ""Gizmos."""'
  ```

  Dans ce cas, si vous devez utiliser une apostrophe à l'intérieur de votre chaîne, vous devez y échapper. Cela signifie qu'il faut utiliser un autre guillemet simple (') avant l'apostrophe.
- Vous pouvez utiliser des guillemets («) à l'extérieur de votre chaîne pour la contenir. Et vous pouvez éviter les guillemets doubles à l'intérieur de votre chaîne en utilisant la barre oblique inverse (\), comme indiqué dans l'exemple suivant.

  ```
  properties:
    COMPANYNAME: "\"Acme \"\"Widgets\"\" and \"\"Gizmos.\"\"\""
  ```

Ces deux méthodes transmettent la valeur `COMPANYNAME="Acme ""Widgets"" and ""Gizmos."""` à la msiexec commande.

Modules d'action pour l'installation du logiciel

- Installez MSI (Windows)
- Désinstallez MSI (Windows)

Installez MSI (Windows)

Le module `InstallMSI` d'action installe une application Windows à l'aide d'un fichier MSI. Vous pouvez spécifier le fichier MSI à l'aide d'un chemin local, d'un URI d'objet S3 ou d'une URL Web. L'option de redémarrage configure le comportement de redémarrage du système.

AWSTOE génère la msiexec commande en fonction des paramètres d'entrée du module d'action. Les valeurs des paramètres d'entrée `path` (emplacement du fichier MSI) et `logFile` (emplacement du fichier journal) doivent être placées entre guillemets («).

Les codes de sortie MSI suivants sont considérés comme réussis :

- 0 (Succès)
- 1614 (ERROR_PRODUCT_UNINSTALL)
- 1641 (redémarrage initié)
- 3010 (redémarrage requis)

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`path`	Spécifiez l'emplacement du fichier MSI à l'aide de l'une des méthodes suivantes : • Le chemin du fichier local. Le chemin peut être absolu ou relatif • Un URI d'objet S3 valide.	String	Oui	N/A	N/A

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	• Une HTTP/HTTPS URL Web valide (HTTPS est recommandé) conforme à la norme RFC 3986. Les expressions de chaînage sont autorisées.				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`reboot`	Configurez le comportement de redémarrage du système après une exécution réussie du module d'action. Paramètres : • `Force`— Lance un redémarrage du système une fois la msiexec commande exécutée avec succès. • `Allow`— Lance un redémarrage du système si la msiexec commande renvoie	String	Non	`Allow`	`Allow`, `Force`, `Skip`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	un code de sortie indiquant qu'un redémarrage est nécessaire. • `Skip`— Enregistre un message d'information dans le `console.log` fichier indiquant qu'un redémarrage a été ignoré. Cette option empêche le redémarrage, même si la msiexec commande renvoie un code de sortie indiquant qu'un				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	redémarrage est nécessaire.				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`logOptions`	Spécifiez les options à utiliser pour la journalisation de l'installation MSI. Les indicateurs spécifiés sont transmis au programme d'installation MSI, avec le paramètre de ligne de `/L` commande pour activer la journalisation. Si aucun indicateur n'est spécifié, AWSTOE utilise la valeur par défaut. Pour plus d'informations sur les options de journalisation pour MSI,	String	Non	`*VX`	`i,w,e,a,r,u,c,m,o,p,v,x,+,!,*`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	consultez la section Options de ligne de commande dans la documentation du produit Microsoft Windows Installer.				
`logFile`	Un chemin absolu ou relatif vers l'emplacement du fichier journal. Si le chemin du fichier journal n'existe pas, il est créé. Si le chemin du fichier journal n'est pas fourni, AWSTOE ne stocke pas le journal d'installation MSI.	String	Non	N/A	N/A

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`properties`	Paires clé-valeur des propriétés de journalisation MSI, par exemple : `TARGETDIR: "C:\target\location"` Remarque : La modification des propriétés suivantes n'est pas autorisée : • `REBOOT="ReallySupress"` • `REINSTALLMODE="ecmus"` • `REINSTALL="ALL"`	Carte [String] Chaîne	Non	N/A	N/A

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`ignoreAuthenticodeSignatureErrors`	Indicateur permettant d'ignorer les erreurs de validation de signature authenticode pour le programme d'installation spécifié dans le chemin. La Get-AuthenticodeSignature commande est utilisée pour valider les programmes d'installation. Paramètres : • `true`— Les erreurs de validation sont ignorées et le programme d'install	Booléen	Non	`false`	`true`, `false`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	ation s'exécute. • `false`— Les erreurs de validation ne sont pas ignorées. Le programme d'installation ne s'exécute que lorsque la validation est réussie. Il s’agit du comportement de par défaut.				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`allowUnsignedInstaller`	Indicateur permettant d'exécuter le programme d'installation non signé spécifié dans le chemin. La Get-AuthenticodeSignature commande est utilisée pour valider les programmes d'installation. Paramètres : • `true`— Ignore le `NotSigned` statut renvoyé par la Get-AuthenticodeSignature commande et exécute le programme	Booléen	Non	`false`	`true`, `false`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	d'install ation. • `false`— Nécessite la signature du programme d'install ation. Les programme s d'install ation non signés ne s'exécute ront pas. Il s'agit du comportem ent de par défaut.				

Exemples

Les exemples suivants montrent des variantes de la section de saisie de votre document de composant, en fonction de votre chemin d'installation.

Exemple de saisie : installation du chemin de document local

```
- name: local-path-install
  steps:
    - name: LocalPathInstaller
      action: InstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-install.log
        logOptions: '*VX'
```

```
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

Exemple de saisie : installation du chemin Amazon S3

```
- name: s3-path-install
  steps:
    - name: S3PathInstaller
      action: InstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-install.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

Exemple de saisie : installation du chemin Web

```
- name: web-path-install
  steps:
    - name: WebPathInstaller
      action: InstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-install.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: false
```

Output

Voici un exemple de sortie du module `InstallMSI` d'action.

```
{
 "logFile": "web-path-install.log",
 "msiExitCode": 0,
 "stdout": ""
}
```

Désinstallez MSI (Windows)

Le module `UninstallMSI` d'action permet de supprimer une application Windows à l'aide d'un fichier MSI. Vous pouvez spécifier l'emplacement du fichier MSI à l'aide d'un chemin de fichier local, d'un URI d'objet S3 ou d'une URL Web. L'option de redémarrage configure le comportement de redémarrage du système.

AWSTOE génère la msiexec commande en fonction des paramètres d'entrée du module d'action. L'emplacement du fichier MSI (`path`) et l'emplacement du fichier journal (`logFile`) sont explicitement placés entre guillemets («) lors de la génération de la msiexec commande.

Les codes de sortie MSI suivants sont considérés comme réussis :

- 0 (Succès)
- 1605 (ERROR_UNKNOWN_PRODUCT)
- 1614 (ERROR_PRODUCT_UNINSTALL)
- 1641 (redémarrage initié)
- 3010 (redémarrage requis)

Input

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`path`	Spécifiez l'emplacement du fichier MSI à l'aide de l'une des méthodes suivantes : • Le chemin du fichier local. Le chemin	String	Oui	N/A	N/A

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	peut être absolu ou relatif. • Un URI d'objet S3 valide. • Une HTTP/HTTPS URL Web valide (HTTPS est recommandé) conforme à la norme RFC 3986.				
	Les expressions de chaînage sont autorisées.				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`reboot`	Configure le comportement de redémarrage du système après une exécution réussie du module d'action. Paramètres : • `Force`— Lance un redémarrage du système une fois la msiexec commande exécutée avec succès. • `Allow`— Lance un redémarrage du système si la msiexec commande renvoie	String	Non	`Allow`	`Allow`, `Force`, `Skip`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	un code de sortie indiquant qu'un redémarrage est nécessaire. • `Skip`— Enregistre un message d'information dans le `console.log` fichier indiquant qu'un redémarrage a été ignoré. Cette option empêche le redémarrage, même si la msiexec commande renvoie un code de sortie indiquant qu'un				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	redémarrage est nécessaire.				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`logOptions`	Spécifiez les options à utiliser pour la journalisation de l'installation MSI. Les indicateurs spécifiés sont transmis au programme d'installation MSI, avec le paramètre de ligne de `/L` commande pour activer la journalisation. Si aucun indicateur n'est spécifié, AWSTOE utilise la valeur par défaut. Pour plus d'informations sur les options de journalisation pour MSI,	String	Non	`*VX`	`i,w,e,a,r,u,c,m,o,p,v,x,+,!,*`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	consultez la section Options de ligne de commande dans la documentation du produit Microsoft Windows Installer.				
`logFile`	Un chemin absolu ou relatif vers l'emplacement du fichier journal. Si le chemin du fichier journal n'existe pas, il est créé. Si le chemin du fichier journal n'est pas fourni, AWSTOE ne stocke pas le journal d'installation MSI.	String	Non	N/A	N/A

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`properties`	Paires clé-valeur des propriétés de journalisation MSI, par exemple : `TARGETDIR: "C:\target\location"` Remarque : La modification des propriétés suivantes n'est pas autorisée : • `REBOOT="ReallySupress"` • `REINSTALLMODE="ecmus"` • `REINSTALL="ALL"`	Carte [String] Chaîne	Non	N/A	N/A

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`ignoreAuthenticodeSignatureErrors`	Indicateur permettant d'ignorer les erreurs de validation de signature authenticode pour le programme d'installation spécifié dans le chemin. La Get-AuthenticodeSignature commande est utilisée pour valider les programmes d'installation. Paramètres : • `true`— Les erreurs de validation sont ignorées et le programme d'install	Booléen	Non	`false`	`true`, `false`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	ation s'exécute. • `false`— Les erreurs de validation ne sont pas ignorées. Le programme d'installation ne s'exécute que lorsque la validation est réussie. Il s’agit du comportement de par défaut.				

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
`allowUnsignedInstaller`	Indicateur permettant d'exécuter le programme d'installation non signé spécifié dans le chemin. La Get-AuthenticodeSignature commande est utilisée pour valider les programmes d'installation. Paramètres : • `true`— Ignore le `NotSigned` statut renvoyé par la Get-AuthenticodeSignature commande et exécute le programme	Booléen	Non	`false`	`true`, `false`

Nom de la touche	Description	Type	Obligatoire	Valeur par défaut	Valeurs acceptables
	d'installation. • `false`— Nécessite la signature du programme d'installation. Les programmes d'installation non signés ne s'exécuteront pas. Il s'agit du comportement de par défaut.				

Exemples

Les exemples suivants montrent des variantes de la section de saisie de votre document de composant, en fonction de votre chemin d'installation.

Exemple de saisie : installation de suppression du chemin de document local

```
- name: local-path-uninstall
  steps:
    - name: LocalPathUninstaller
      action: UninstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-uninstall.log
        logOptions: '*VX'
```

```
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

Exemple de saisie : supprimer l'installation du chemin Amazon S3

```
- name: s3-path-uninstall
  steps:
    - name: S3PathUninstaller
      action: UninstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-uninstall.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

Exemple de saisie : installation de suppression du chemin Web

```
- name: web-path-uninstall
  steps:
    - name: WebPathUninstaller
      action: UninstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-uninstall.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: false
```

Output

Voici un exemple de sortie du module `UninstallMSI` d'action.

```
{
 "logFile": "web-path-uninstall.log",
 "msiExitCode": 0,
 "stdout": ""
}
```

Modules d'action du système

La section suivante décrit les modules d'action qui exécutent des actions système ou mettent à jour les paramètres système.

Modules d'action du système

- Redémarrer (Linux, Windows)
- SetRegistry (Fenêtres)
- Mettre à jour le système d'exploitation (Linux, Windows)

Redémarrer (Linux, Windows)

Le module d'action Reboot redémarre l'instance. Il dispose d'une option configurable pour retarder le démarrage du redémarrage. Par défaut, `delaySeconds` est défini sur`0`, ce qui signifie qu'il n'y a aucun délai. Le délai d'expiration des étapes n'est pas pris en charge pour le module d'action Reboot, car il ne s'applique pas au redémarrage de l'instance.

Si l'application est appelée par l'agent Systems Manager, elle transmet le code de sortie (`3010`pour Windows, `194` pour Linux) à l'agent Systems Manager. L'agent Systems Manager gère le redémarrage du système comme décrit dans Reboot Managed Instance from Scripts.

Si l'application est invoquée sur l'hôte en tant que processus autonome, elle enregistre l'état d'exécution actuel, configure un déclencheur d'exécution automatique après le redémarrage pour réexécuter l'application après le redémarrage, puis redémarre le système.

Post-reboot déclencheur automatique :

- Fenêtres. AWSTOE crée une entrée du planificateur de tâches Windows avec un déclencheur qui s'exécute automatiquement sur `SystemStartup`
- Linux. AWSTOE ajoute une tâche dans crontab qui s'exécute automatiquement après le redémarrage du système.

```
@reboot /download/path/awstoe run --document s3://bucket/key/doc.yaml
```

Ce déclencheur est nettoyé au démarrage de l'application.

Nouvelles tentatives

Par défaut, le nombre maximum de tentatives est défini sur le Systems Manager`CommandRetryLimit`. Si le nombre de redémarrages dépasse la limite de nouvelles tentatives, l'automatisation échoue. Vous pouvez modifier la limite en modifiant le fichier de configuration de l'agent Systems Manager (`Mds.CommandRetryLimit`). Consultez la section Configuration du runtime dans l'agent open source de Systems Manager.

Pour utiliser le module d'action Redémarrer, pour les étapes contenant un redémarrage `exitcode` (par exemple,`3010`), vous devez exécuter le binaire de l'application sous le nom de`sudo user`.

Input

Nom de la touche	Description	Type	Obligatoire	Par défaut
`delaySeconds`	Retarde un certain temps avant de lancer un redémarrage.	Entier	Non	`0`

Exemple de saisie : étape de redémarrage

```
- name: RebootStep
  action: Reboot
  onFailure: Abort
  maxAttempts: 2
  inputs:
    delaySeconds: 60
```

Sortie

Aucune.

Lorsque le module Reboot est terminé, Image Builder passe à l'étape suivante de la génération.

SetRegistry (Fenêtres)

Le module SetRegistryd'action accepte une liste d'entrées et vous permet de définir la valeur de la clé de registre spécifiée. Si aucune clé de registre n'existe, elle est créée dans le chemin défini. Cette fonctionnalité s'applique uniquement à Windows.

Input

Nom de la touche	Description	Type	Obligatoire
`path`	Chemin de la clé de registre.	String	Oui
`name`	Nom de la clé de registre.	String	Oui
`value`	Valeur de la clé de registre.	String/Number/Array	Oui
`type`	Type de valeur de la clé de registre.	String	Oui

Préfixes de chemin pris en charge

- `HKEY_CLASSES_ROOT / HKCR:`
- `HKEY_USERS / HKU:`
- `HKEY_LOCAL_MACHINE / HKLM:`
- `HKEY_CURRENT_CONFIG / HKCC:`
- `HKEY_CURRENT_USER / HKCU:`

Types pris en charge

- `BINARY`
- `DWORD`
- `QWORD`
- `SZ`
- `EXPAND_SZ`

- MULTI_SZ

Exemple de saisie : définir les valeurs des clés de registre

```
- name: SetRegistryKeyValues
  action: SetRegistry
  maxAttempts: 3
  inputs:
    - path: HKLM:\SOFTWARE\MySoftWare
      name: MyName
      value: FirstVersionSoftware
      type: SZ
    - path: HKEY_CURRENT_USER\Software\Test
      name: Version
      value: 1.1
      type: DWORD
```

Sortie

Aucune.

Mettre à jour le système d'exploitation (Linux, Windows)

Le module d'action UpdateOS prend en charge l'installation des mises à jour Windows et Linux. Il installe toutes les mises à jour disponibles par défaut. Vous pouvez également configurer une liste d'une ou de plusieurs mises à jour spécifiques à installer par le module d'action. Vous pouvez également spécifier les mises à jour à exclure de l'installation.

Si des listes « à inclure » et « à exclure » sont fournies, la liste des mises à jour qui en résulte ne peut inclure que celles répertoriées dans la liste « à inclure » qui ne figurent pas dans la liste « à exclure ».

> **Note**
>
> UpdateOS ne prend pas en charge Amazon Linux 2023 (AL2023). Nous vous recommandons de mettre à jour votre AMI de base vers la nouvelle version fournie avec chaque nouvelle version. Pour d'autres alternatives, consultez la section Contrôler les mises à jour reçues à partir des versions majeures et mineures dans le Guide de l'utilisateur Amazon Linux 2023.

- Fenêtres. Les mises à jour sont installées à partir de la source de mise à jour configurée sur la machine cible.

- Linux. L'application recherche le gestionnaire de packages pris en charge sur la plate-forme Linux et utilise l'un ou l'autre `yum` ou le gestionnaire de `apt-get` packages. Si aucun des deux n'est pris en charge, une erreur est renvoyée. Vous devez être `sudo` autorisé à exécuter le module d'action UpdateOS. Si vous n'avez pas d'`sudo`autorisations, un `error.Input` est renvoyé.

Input

Nom de la touche	Description	Type	Obligatoire
`include`	Pour Windows, vous pouvez spécifier les éléments suivants : • Un ou plusieurs numéros d'article de la Base de connaissances Microsoft (KB) à inclure dans la liste des mises à jour qui peuvent être installées. Les formats valides sont `KB1234567` ou `1234567`. • Un nom de mise à jour utilisant une valeur générique (*). Les formats valides sont `Security*` ou `*Security*` .	Liste de chaînes	Non

Nom de la touche	Description	Type	Obligatoire
	Pour Linux, vous pouvez spécifier un ou plusieurs packages à inclure dans la liste des mises à jour à installer.		

Nom de la touche	Description	Type	Obligatoire
`exclude`	Pour Windows, vous pouvez spécifier les éléments suivants : • Un ou plusieurs numéros d'article de la Base de connaissances Microsoft (KB) à inclure dans la liste des mises à jour à exclure de l'installation. Les formats valides sont `KB1234567` ou `1234567`. • Un nom de mise à jour utilisant une valeur générique (*). Les formats valides sont : `Security*` ou`*Security*` . Pour Linux, vous pouvez spécifier un ou plusieurs packages à exclure de la liste des mises à jour à installer.	Liste de chaînes	Non

Exemple de saisie : ajout d'un support pour l'installation de mises à jour Linux

```
- name: UpdateMyLinux
  action: UpdateOS
  onFailure: Abort
  maxAttempts: 3
  inputs:
    exclude:
      - ec2-hibinit-agent
```

Exemple de saisie : ajout d'un support pour l'installation des mises à jour Windows

```
- name: UpdateWindowsOperatingSystem
  action: UpdateOS
  onFailure: Abort
  maxAttempts: 3
  inputs:
    include:
      - KB1234567
      - '*Security*'
```

Sortie

Aucune.

Configurez l'entrée pour AWSTOE exécuter la commande

Pour rationaliser la saisie de votre commande sur la ligne de AWSTOE run commande, vous pouvez inclure les paramètres et options de commande dans un fichier de configuration d'entrée au format JSON avec une extension de `.json` fichier. AWSTOE peut lire votre fichier depuis l'un des emplacements suivants :

- Un chemin de fichier local (*`./config.json`*).
- Un compartiment S3 (*`s3://<bucket-path>/<bucket-name>/config.json`*).

Lorsque vous entrez la run commande, vous pouvez spécifier le fichier de configuration d'entrée à l'aide du --config paramètre. Par exemple :

```
awstoe run --config <file-path>/config.json
```

Fichier de configuration d'entrée

Le fichier JSON de configuration d'entrée inclut des paires clé-valeur pour tous les paramètres que vous pouvez fournir directement via les paramètres et options de run commande. Si vous spécifiez un paramètre à la fois dans le fichier de configuration d'entrée et dans la run commande, en tant que paramètre ou option, les règles de priorité suivantes s'appliquent :

Règles de priorité

1. Un paramètre fourni directement à la run commande dans le AWS CLI, via un paramètre ou une option, remplace toute valeur définie dans le fichier de configuration d'entrée pour le même paramètre.
2. Un paramètre du fichier de configuration d'entrée remplace la valeur par défaut d'un composant.
3. Si aucun autre paramètre n'est transmis au document du composant, celui-ci peut appliquer une valeur par défaut, s'il en existe une.

Il existe deux exceptions à cette règle : les documents et les paramètres. Ces paramètres fonctionnent différemment dans la configuration d'entrée et en tant que paramètres de commande. Si vous utilisez le fichier de configuration d'entrée, vous ne devez pas spécifier ces paramètres directement dans la run commande. Cela générera une erreur.

Réglages des composants

Le fichier de configuration d'entrée contient les paramètres suivants. Pour rationaliser votre fichier, vous pouvez omettre les paramètres facultatifs inutiles. Tous les paramètres sont facultatifs, sauf indication contraire.

- cw IgnoreFailures (Boolean) — Ignorez les échecs de journalisation dans les journaux. CloudWatch
- cw LogGroup (String) — Le `LogGroup` nom des CloudWatch journaux.
- cw LogRegion (String) — La AWS région qui s'applique aux CloudWatch journaux.
- cw LogStream (String) — Le `LogStream` nom des CloudWatch journaux, qui indique AWSTOE où diffuser le `console.log` fichier.
- document S3BucketOwner (String) — L'ID de compte du propriétaire du compartiment pour les URI-based documents S3.

- documents (tableau d'objets, obligatoire) — Un tableau d'objets JSON représentant les documents du composant YAML exécutés par la AWSTOE run commande. Au moins un document composant doit être spécifié.

  Chaque objet comprend les champs suivants :

  - path (String, obligatoire) — Emplacement du fichier du document du composant YAML. Il doit s'agir de l'une des options suivantes :
    - Un chemin de fichier local (*./component-doc-example.yaml*).
    - Un URI S3 (s3://*bucket/key*).
    - Un composant Image Builder construit la version ARN (arn:aws:imagebuilder:us-west-:component/ /2021.12. *2:123456789012 my-example-component* 02/1).
  - paramètres (tableau d'objets) — Tableau d'objets d'une paire clé-valeur, chacun représentant un paramètre spécifique au composant que la run commande transmet lorsqu'elle exécute le document du composant. Les paramètres sont facultatifs pour les composants. Le document du composant peut avoir des paramètres définis ou non.

    Chaque objet comprend les champs suivants :

    - name (String, obligatoire) — Nom du paramètre du composant.
    - value (String, obligatoire) — La valeur à transmettre au document du composant pour le paramètre nommé.

    Pour en savoir plus sur les paramètres des composants, consultez la section Paramètres de la Utiliser des variables dans votre document de composant personnalisé page.
- ExecutionId (String) — Il s'agit de l'identifiant unique qui s'applique à l'exécution de la commande en coursrun. Cet identifiant est inclus dans les noms des fichiers de sortie et des fichiers journaux, afin d'identifier ces fichiers de manière unique et de les lier à l'exécution de la commande en cours. Si ce paramètre est omis, AWSTOE génère un GUID.
- LogDirectory (String) — Le répertoire de destination où sont AWSTOE stockés tous les fichiers journaux de l'exécution de cette commande. Par défaut, ce répertoire se trouve dans le répertoire parent suivant :TOE_<DATETIME>_<EXECUTIONID>. Si vous ne spécifiez pas le répertoire du journal, AWSTOE utilise le répertoire de travail actuel (.).
- log S3BucketName (String) — Si les journaux des composants sont stockés dans Amazon S3 (recommandé), AWSTOE télécharge les journaux des applications des composants dans le compartiment S3 nommé dans ce paramètre.

- log S3BucketOwner (String) — Si les journaux des composants sont stockés dans Amazon S3 (recommandé), il s'agit de l'ID de compte propriétaire du compartiment dans lequel AWSTOE les fichiers journaux sont écrits.
- log S3KeyPrefix (String) — Si les journaux des composants sont stockés dans Amazon S3 (recommandé), il s'agit du préfixe de clé d'objet S3 pour l'emplacement du journal dans le compartiment.
- paramètres (tableau d'objets) — Tableau d'objets d'une paire clé-valeur qui représente des paramètres qui s'appliquent globalement à tous les composants inclus dans l'exécution de la run commande en cours.
  - name (String, obligatoire) — Le nom du paramètre global.
  - value (String, obligatoire) — La valeur à transmettre à tous les documents du composant pour le paramètre nommé.
- phases (String) — Liste séparée par des virgules qui spécifie les phases à exécuter à partir des documents du composant YAML. Si le document d'un composant inclut des phases supplémentaires, celles-ci ne seront pas exécutées.
- StateDirectory (String) — Le chemin du fichier dans lequel les fichiers de suivi d'état sont stockés.
- trace (booléen) — Active la journalisation détaillée sur la console.

Exemples

L'exemple suivant montre un fichier de configuration d'entrée qui exécute les `test` phases `build` et pour deux documents de composants : `sampledoc.yaml` et`conversation-intro.yaml`. Chaque document composant possède un paramètre qui ne s'applique qu'à lui-même, et les deux utilisent un paramètre partagé. Le `project` paramètre s'applique aux deux documents du composant.

```
{
   "documents": [
     {
       "path": "<file path>/awstoe/sampledoc.yaml>",
       "parameters": [
         {
           "name": "dayofweek",
           "value": "Monday"
         }
       ]
     },
     {
```

```
      "path": "<file path>/awstoe/conversation-intro.yaml>",
      "parameters": [
        {
          "name": "greeting",
          "value": "Hello, HAL."
        }
      ]
    }
  ],
  "phases": "build,test",
  "parameters": [
    {
      "name": "project",
      "value": "examples"
    }
  ],
  "cwLogGroup": "<log_group_name>",
  "cwLogStream": "<log_stream_name>",
  "documentS3BucketOwner": "<owner_aws_account_number>",
  "executionId": "<id_number>",
  "logDirectory": "<local_directory_path>",
  "logS3BucketName": "<bucket_name_for_log_files>",
  "logS3KeyPrefix": "<key_prefix_for_log_files>",
  "logS3BucketOwner": "<owner_aws_account_number>"
}
```

Ressources d'image de sortie d'Image Builder

Après avoir créé des ressources d'image pour les images AMI ou de conteneur avec Image Builder, vous pouvez les gérer à l'aide de la console Image Builder, via l'API Image Builder ou à l'aide des imagebuilder commandes du AWS CLI.

> **Tip**
>
> Lorsque vous disposez de plusieurs ressources du même type, le balisage vous aide à identifier une ressource spécifique en fonction des balises que vous lui avez attribuées. Pour plus d'informations sur le balisage de vos ressources à l'aide des commandes Image Builder dans le AWS CLI, consultez la Balisage des ressources section de ce guide.

Cette section explique comment répertorier, afficher et créer des images. Pour plus d'informations sur les flux de production d'images et sur la façon de les gérer, consultezGérez les flux de travail de création, de test et de distribution pour les images Image Builder.

Table des matières

- Répertorier les images et créer des versions
- Afficher les détails des ressources d'images
- Créez des images personnalisées avec Image Builder
- Importez et exportez des images de machines virtuelles avec Image Builder
- Importez des images de disque ISO Windows vérifiées avec Image Builder
- Gestion des résultats de sécurité pour les images d'Image Builder
- Nettoyez les ressources d'Image Builder

Répertorier les images et créer des versions

Sur la page Images de la console Image Builder, vous pouvez consulter la liste de toutes les ressources d'image Image Builder que vous possédez, qui sont partagées avec vous et auxquelles vous avez accès. Les résultats de la liste incluent des informations clés sur ces ressources.

Vous pouvez également voir toutes les images de votre compte pour lesquelles des actions de flux de travail sont en attente.

Table des matières

- Liste des images
- Répertorier les images en attente d'action
- Lister les versions de génération d'images

Liste des images

Cette section décrit les différentes manières de répertorier les informations relatives à vos images.

Vous pouvez utiliser l'une des méthodes suivantes pour répertorier les ressources d'image Image Builder auxquelles vous avez accès. Pour l'action de l'API, voir le ListImagesGuide de référence de l'API EC2 Image Builder. Pour la demande de SDK associée, reportez-vous au lien Voir aussi sur la même page.

Table des matières

- Répertorier les images dans la console
- Répertorier les images avec des AWS CLI commandes

Répertorier les images dans la console

Pour ouvrir la page de liste des images dans la console, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder/.
2. Choisissez Images dans le volet de navigation.

La page Images de la console est divisée en onglets, en fonction de la propriété de l'image ou des actions de flux de travail en attente. Cette section couvre les trois premiers onglets qui affichent les images que vous possédez ou auxquelles vous avez accès.

Onglet console : Je suis propriétaire

Dans l'onglet Owned by me, vous pouvez utiliser les filtres suivants pour rationaliser les résultats de la liste d'images.

- Vous pouvez rechercher tout ou partie du nom dans la barre de recherche.

- Vous pouvez filtrer les images en fonction de la plate-forme de leur système d'exploitation (Windows, Linux ou macOS).
- Vous pouvez filtrer les images en fonction du type de sortie qu'elles produisent (AMI ou image de conteneur).
- Vous pouvez utiliser le filtre source pour rechercher des images importées depuis une machine virtuelle (VMIE) ou depuis une image de disque ISO.

Après les contrôles de filtre, l'onglet Owned by me affiche la liste des images Image Builder que vous avez créées, avec les détails suivants pour les ressources répertoriées :

Nom/Version

Les noms des ressources d'image Image Builder commencent par le nom de la recette et la version à partir de laquelle elles ont été créées. Sélectionnez le lien pour voir toutes les versions de génération d'image associées.

Type

Type d'image de sortie créée par Image Builder pour cette ressource d'image (une AMI ou une image de conteneur).

Plateforme

La plate-forme du système d'exploitation de la ressource image, par exemple « Linux », « Windows » ou « macOS ».

Source de l'image

Origine de l'image de base utilisée par Image Builder pour créer cette ressource d'image. Ceci est principalement utilisé pour filtrer les résultats des images importées depuis une machine virtuelle (VMIE).

Heure de création

Date et heure auxquelles Image Builder a créé la version actuelle de la ressource d'image.

ARN

Le nom de ressource Amazon (ARN) de la version actuelle de la ressource image.

Onglet console : partagé avec moi

Dans l'onglet Partagé avec moi, vous pouvez utiliser les filtres suivants pour rationaliser les résultats de la liste d'images.

- Vous pouvez rechercher tout ou partie du nom dans la barre de recherche.
- Vous pouvez filtrer les images en fonction de la plate-forme de leur système d'exploitation (Windows, Linux ou macOS).
- Vous pouvez filtrer les images en fonction du type de sortie qu'elles produisent (AMI ou image de conteneur).
- Vous pouvez utiliser le filtre source pour rechercher des images importées depuis une machine virtuelle (VMIE) ou depuis une image de disque ISO.

Après les contrôles de filtre, l'onglet Shared with me affiche une liste des images Image Builder qui ont été partagées avec vous, avec les détails suivants pour les ressources répertoriées :

Nom de l'image

Le nom de la ressource d'image qui a été partagée avec vous. Pour utiliser une image partagée dans une recette, vous devez sélectionner l'option Sélectionner les images gérées et modifier l'origine de l'image en Images partagées avec moi.

Type

Type d'image de sortie créée par Image Builder pour cette ressource d'image (une AMI ou une image de conteneur).

Version

<major>Version de la plate-forme du système d'exploitation pour la ressource image, généralement un champ numérique au format suivant :. <minor>. <patch>.

Source de l'image

Origine de l'image de base utilisée par Image Builder pour créer cette ressource d'image, le cas échéant. Ceci est principalement utilisé pour filtrer les résultats des images importées depuis une machine virtuelle (VMIE).

Plateforme

La plate-forme du système d'exploitation de la ressource image, par exemple « Linux », « Windows » ou « macOS ».

Heure de création

Date et heure auxquelles Image Builder a créé la version de la ressource d'image qui a été partagée avec vous.

Propriétaire

Le propriétaire de la ressource d'image partagée.

ARN

Le nom de ressource Amazon (ARN) de la version de la ressource image qui a été partagée avec vous.

Onglet console : géré par Amazon

Dans l'onglet Managed by Amazon, vous pouvez utiliser les filtres suivants pour rationaliser les résultats des listes d'images.

- Vous pouvez rechercher tout ou partie du nom dans la barre de recherche.
- Vous pouvez filtrer les images en fonction de la plate-forme de leur système d'exploitation (Windows, Linux ou macOS).
- Vous pouvez filtrer les images en fonction du type de sortie qu'elles produisent (AMI ou image de conteneur).
- Vous pouvez utiliser le filtre source pour rechercher des images importées depuis une machine virtuelle (VMIE) ou depuis une image de disque ISO.

Après les contrôles de filtre, l'onglet Managed by Amazon affiche une liste d'images Image Builder gérées par Amazon que vous pouvez utiliser comme images de base pour vos recettes. Image Builder affiche les informations suivantes pour les ressources répertoriées :

Nom de l'image

Nom de l'image gérée. Lorsque vous créez une recette, l'image de base par défaut est Quick start (géré par Amazon). Les images répertoriées dans cet onglet alimentent la liste des noms d'images associée à la plate-forme du système d'exploitation que vous choisissez pour votre image de base lorsque vous créez une recette.

Type

Type d'image de sortie créée par Image Builder pour cette ressource d'image (une AMI ou une image de conteneur).

Version

<major>Version de la plate-forme du système d'exploitation pour la ressource image, généralement un champ numérique au format suivant :. <minor>. <patch>.

Plateforme

La plate-forme du système d'exploitation de la ressource image, par exemple « Linux », « Windows » ou « macOS ».

Heure de création

Date et heure auxquelles Image Builder a créé la version de la ressource d'image qui a été partagée avec vous.

Propriétaire

Amazon est propriétaire des images gérées.

ARN

Le nom de ressource Amazon (ARN) de la version de la ressource image qui a été partagée avec vous.

Répertorier les images avec des AWS CLI commandes

Lorsque vous exécutez la list-images commande dans le AWS CLI, vous pouvez obtenir une liste des images que vous possédez ou auxquelles vous avez accès.

L'exemple de commande suivant montre comment utiliser la list-images commande sans filtre pour répertorier toutes les ressources d'image Image Builder que vous possédez.

Exemple : liste toutes les images

```
aws imagebuilder list-images
```

Sortie :

```
{
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
```

```
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  },
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-win/1.0.1",
    "name": "image-recipe-win",
    "type": "AMI",
    "version": "1.0.1",
    "platform": "Windows",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  },
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
macos/1.1.1",
    "name": "image-recipe-macos",
    "type": "AMI",
    "version": "1.1.1",
    "platform": "macOS",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

Lorsque vous exécutez la list-images commande, vous pouvez appliquer des filtres pour rationaliser les résultats, comme le montre l'exemple suivant. Pour plus d'informations sur le filtrage des résultats, consultez la commande list-images dans la référence des AWS CLI commandes.

Exemple : filtre pour images Linux

```
aws imagebuilder list-images --filters name="platform",values="Linux"
```

Sortie :

```
{
```

```
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

Répertorier les images en attente d'action

Lorsque vous utilisez l'action d'`WaitForAction`étape dans votre flux de travail d'images, le flux de travail est suspendu jusqu'à ce que vous lui envoyiez un signal lui demandant de reprendre le traitement ou d'échouer. Vous pouvez utiliser cette action d'étape si un processus externe doit être exécuté avant de continuer. Vous pouvez ensuite utiliser le `SendWorkflowStepAction` pour envoyer un signal à l'étape interrompue vers `RESUME` ou`STOP`. Vous pouvez également arrêter ou reprendre votre flux de travail depuis la console.

Les onglets suivants indiquent comment obtenir une liste de toutes les ressources d'image de votre compte avec les étapes du flux de travail actuellement suspendues pour attendre la reprise ou l'arrêt d'un signal. Les onglets couvrent les étapes de la console et la AWS CLI commande.

Vous pouvez également utiliser l'API ou un SDK pour obtenir une liste des étapes du flux de travail en attente d'une action. Pour l'action de l'API, voir le ListWaitingWorkflowStepsGuide de référence de l'API EC2 Image Builder. Pour la demande de SDK associée, reportez-vous au lien Voir aussi sur la même page.

Console

Pour accéder à l'onglet En attente d'action de la console, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder/.
2. Choisissez Images dans le volet de navigation. Cela ouvre la page de liste des images.
3. Sélectionnez l'onglet En attente d'action dans la page de liste.

4. (facultatif) Pour arrêter ou reprendre une étape, cochez la case à côté du nom, puis choisissez Arrêter l'étape ou Reprendre l'étape. Vous pouvez cocher plusieurs cases pour exécuter la même action pour toutes les étapes sélectionnées.

Détails des étapes du flux de travail en attente

Les détails du flux de travail pour l'étape en attente sont les suivants :

- Nom de l'image : nom de la ressource d'image dont l'étape est en attente. Vous pouvez sélectionner le lien du nom pour afficher la page détaillée de cette image.
- Nom de l'étape en attente : nom de l'étape du flux de travail en attente d'une action.
- ID d'exécution de l'étape : identifie de manière unique l'instance d'exécution de l'étape du flux de travail. Vous pouvez sélectionner l'ID lié pour afficher les détails d'exécution de l'étape.
- Début de l'étape : horodatage du début de l'instance d'exécution de l'étape du flux de travail.
- ARN du flux de travail : nom de ressource Amazon (ARN) du flux de travail avec l'étape en attente.
- Actions : action d'étape en état d'attente.

AWS CLI

Lorsque vous exécutez la list-waiting-workflow-steps commande dans le AWS CLI, vous obtenez une liste de toutes les images de votre compte dont les étapes du flux de travail sont en attente d'une action avant de terminer le processus de création d'image.

L'exemple de commande suivant montre comment utiliser la list-waiting-workflow-steps commande pour répertorier toutes les images de votre compte avec les étapes du flux de travail en attente d'une action.

Exemple : répertorier les images de votre compte avec les étapes du flux de travail en attente

```
aws imagebuilder list-waiting-workflow-steps
```

Sortie :

Le résultat de cet exemple montre une image du compte avec une étape en attente d'action.

```
{
```

```
    "steps": [
        {
            "imageBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:image/example-image/1.0.0/8",
            "name": "WaitForAction",
            "workflowExecutionId": "wf-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "stepExecutionId": "step-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "workflowBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:workflow/test/wait-for-action/1.0.0/1",
            "startTime": "2023-11-21T23:21:23.609Z",
            "action": "WaitForAction"
        }
    ]
}
```

Lister les versions de génération d'images

Sur la page des versions de génération d'images de la console Image Builder, vous pouvez consulter la liste des versions de génération et des informations supplémentaires sur une ressource d'image dont vous êtes le propriétaire. Vous pouvez également utiliser des commandes ou des actions avec l'API Image Builder ou AWS CLI pour répertorier les versions de génération d'image. SDKs

Vous pouvez utiliser l'une des méthodes suivantes pour répertorier les versions de génération d'image pour les ressources d'images que vous possédez. Pour l'action de l'API, voir le ListImageBuildVersionsGuide de référence de l'API EC2 Image Builder. Pour la demande de SDK associée, reportez-vous au lien Voir aussi sur la même page.

Console

Détails de la version

La page des versions d'Image build de la console Image Builder contient les informations suivantes :

- Version — Version de génération de la ressource image. Dans la console Image Builder, la version renvoie à une page détaillée de l'image.
- Type : type de sortie distribué par Image Builder lors de la création de cette ressource d'image (une AMI ou une image de conteneur).
- Date de création : date et heure auxquelles Image Builder a créé la version de génération de l'image.

- État de l'image — État actuel de la version créée par l'image. L'état peut être lié à la création ou à la disposition de l'image. Par exemple, pendant le processus de création, vous pouvez voir le statut `Building` ou`Distributing`. Pour la disposition de l'image, le statut `Deprecated` ou peut s'afficher`Deleted`.
- Raison de l'échec : raison de l'état de l'image. La console Image Builder affiche uniquement la raison de l'échec de la génération (le statut de l'image est égal à`Failed`).
- Résultats de sécurité : résultats de numérisation d'images agrégés pour la version de génération d'image référencée.
- ARN — Le nom de ressource Amazon (ARN) pour la version référencée de la ressource image.
- Flux de journal : lien vers le détail du flux de journal pour la version de génération de l'image référencée.

Versions de liste

Pour répertorier les versions de génération d'image dans la console Image Builder, effectuez les opérations suivantes :

1. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder/.
2. Choisissez Images dans le volet de navigation. Par défaut, la liste des images affiche la version actuelle de chacune des images que vous possédez.
3. Pour voir la liste de toutes les versions d'une image, cliquez sur le lien de la version actuelle. Le lien ouvre la page des versions de génération d'image qui répertorie toutes les versions de génération pour une image spécifique.

AWS CLI

Lorsque vous exécutez la list-image-build-versions commande dans le AWS CLI, vous obtenez une liste complète des versions de build pour la ressource d'image spécifiée. Vous devez posséder l'image pour exécuter cette commande.

L'exemple de commande suivant montre comment utiliser la list-image-build-versions commande pour répertorier toutes les versions de compilation pour l'image spécifiée.

Exemple : liste des versions de compilation pour une image spécifique

```
aws imagebuilder list-image-build-versions --image-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0
```

Sortie :

La sortie de cet exemple inclut deux versions de génération pour la recette d'image spécifiée.

```
{
  "requestId": "12f3e45d-67cb-8901-af23-45ed678c9b01",
  "imageSummaryList": [
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/2",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/2",
    "platform": "Linux",
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T01:04:40.609Z",
    "outputResources": {
      "amis": [
        {
        "region": "us-west-2",
        "image": "ami-012b3456789012c3d",
        "name": "image-recipe-name 2023-03-10T01-05-12.541Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    },
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/1",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/1",
    "platform": "Linux",
```

```
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T00:07:16.384Z",
    "outputResources": {
      "amis": [
        {
        "region": "us-west-2",
        "image": "ami-0d1e23456789f0a12",
        "name": "image-recipe-name 2023-03-10T00-07-18.146132Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    }
  ]
}
```


> **Note**
>
> Le résultat de la list-image-build-versions commande n'inclut pas les résultats de sécurité ni les flux de journaux pour le moment.

Afficher les détails des ressources d'images

Sur la page des détails de l'image de la console Image Builder, vous pouvez consulter les détails d'une ressource d'image spécifique dont vous êtes le propriétaire. Vous pouvez également utiliser des commandes ou des actions avec l'API Image Builder ou AWS CLI pour obtenir des informations détaillées sur les images. SDKs

Pour plus d'informations sur les ressources qu'une autre personne Compte AWS a partagées avec vous par le biais d'un partage de ressources AWS Resource Access Manager (AWS RAM), consultez la section Accès aux AWS ressources partagées avec vous dans le guide de AWS RAM l'utilisateur.

Table des matières

- Afficher les détails de l'image dans la console Image Builder

- Pour plus d'informations sur la politique relative aux images, consultez le AWS CLI

Afficher les détails de l'image dans la console Image Builder

La page détaillée de l'image dans la console Image Builder inclut une section récapitulative, avec des informations supplémentaires regroupées sous forme d'onglets. L'en-tête de page correspond au nom et à la version de compilation de la recette qui a créé l'image. Si un onglet ne s'applique pas à votre image, il est inactif et n'affiche aucune donnée.

Sections et onglets détaillés de la console

- Section récapitulative
- Onglet Ressources de sortie
- Onglet Configuration de l'infrastructure
- Onglet Paramètres de distribution
- Onglet Workflow
- Onglet Résultats de sécurité
- Onglet Balises

Section récapitulative

La section récapitulative couvre toute la largeur de la page et inclut les détails suivants. Ces informations sont toujours affichées.

Recette

Le nom de la recette et la version qui n'incluent pas la version de compilation. Par exemple, si la version de construction l'est`sample-linux-recipe | 1.0.1/2`, la recette l'est `sample-linux-recipe | 1.0.1` et la version de construction l'est2.

Date de création

Date et heure auxquelles Image Builder a créé la version de génération de l'image.

État de l'image

État actuel de la version de création de l'image. L'état peut être lié à la création ou à la disposition de l'image. Par exemple, pendant le processus de création, vous pouvez voir le statut

`Building` ou`Distributing`. Pour la disposition de l'image, le statut `Deprecated` ou peut s'afficher`Deleted`.

Raison de l'échec

La raison de l'état de l'image. La console Image Builder affiche uniquement la raison de l'échec de la génération (le statut de l'image est égal à`Failed`).

Onglet Ressources de sortie

L'onglet Ressources de sortie répertorie les détails de sortie et de distribution de la ressource image actuellement affichée. Les informations affichées par Image Builder dépendent du type de recette utilisé par le pipeline pour créer l'image, comme suit.

Recette d'images

- Région : région de distribution pour l'Amazon Machine Image (AMI) en sortie spécifiée dans la colonne Image.
- Image : ID de l'AMI qu'Image Builder a distribuée à la destination. Cet identifiant est lié à la page Amazon Machine Images (AMIs) de la console Amazon EC2.

> **Note**
>
> Image Builder crée l'AMI après avoir créé la ressource d'image de sortie et avant de distribuer l'AMI à la destination.

- Nom : nom de l'AMI qu'Image Builder a distribuée à la destination.
- Description — Description facultative de la recette d'image utilisée par le pipeline pour créer la ressource d'image de sortie.
- Compte : propriétaire Compte AWS de la ressource d'image Image Builder actuellement affichée.

Recette de contenant

Image Builder affiche les détails suivants pour la sortie créée à partir d'une recette de conteneur.

- Région : région de distribution de l'image du conteneur spécifiée dans la colonne URI de l'image.
- URI de l'image : URI de l'image du conteneur de sortie qu'Image Builder a distribuée au référentiel ECR de la région de destination.

> **Note**
>
> Image Builder affiche une ligne par destination. L'image de sortie comporte toujours au moins une entrée destinée à être distribuée au compte qui a créé l'image. Les destinations supplémentaires peuvent inclure des distributions entre les régions Comptes AWS, ou AWS Organizations. Pour de plus amples informations, veuillez consulter Gérer les paramètres de distribution d'Image Builder.

Onglet Configuration de l'infrastructure

L'onglet Configuration de l'infrastructure affiche les paramètres d'infrastructure Amazon EC2 utilisés par Image Builder pour créer et tester l'image actuellement affichée. Image Builder affiche toujours le nom de la ressource de configuration de l'infrastructure (nom de configuration) et son Amazon Resource Name (ARN). Si la configuration de votre infrastructure définit les valeurs, les détails supplémentaires de l'infrastructure peuvent inclure les éléments suivants

- Types d'instances
- Un profil d'instance
- Infrastructure réseau
- Paramètres du groupe de sécurité
- Un emplacement Amazon S3 où Image Builder stocke les journaux des applications
- Une paire de clés Amazon EC2 pour le dépannage
- Rubrique Amazon SNS pour les notifications d'événements

Pour de plus amples informations, veuillez consulter Gérer la configuration de l'infrastructure d'Image Builder.

Onglet Paramètres de distribution

L'onglet Paramètres de distribution affiche les paramètres utilisés par Image Builder pour distribuer vos images de sortie. Image Builder affiche toujours le nom de la ressource de configuration de distribution (nom de configuration) et son Amazon Resource Name (ARN). Les détails de distribution supplémentaires dépendent du type de recette utilisé par le pipeline Image Builder pour créer l'image, comme suit :

Recette d'images

Si votre ressource de configuration de distribution définit les valeurs, les détails de distribution supplémentaires peuvent inclure les éléments suivants :

- Région : région de distribution pour l'Amazon Machine Image (AMI) en sortie.
- Nom de l'AMI de sortie : nom de l'AMI qu'Image Builder a distribuée à la destination.
- Chiffrement (clé KMS) : si configuré, Image Builder utilise pour chiffrer l'image afin de la distribuer dans la région cible. AWS KMS key
- Comptes cibles pour la distribution : si vous avez configuré la distribution entre comptes, cette colonne affiche une liste séparée par des virgules des personnes avec Comptes AWS lesquelles partager l'image de sortie dans la région cible.
- Principaux dotés d'autorisations partagées : liste séparée par des AWS virgules des principaux autorisés à lancer votre image, par exemple, des groupes Comptes AWS ou des unités organisationnelles (). AWS Organizations OUs

> **Note**
>
> Lorsque vous autorisez d'autres personnes à lancer votre image, vous êtes toujours propriétaire de l'image. AWS facture votre compte pour toutes les instances lancées par Amazon EC2 à partir de votre image.

- Comptes cibles pour une configuration de lancement plus rapide : c'est Comptes AWS là qu'EC2 Fast Launch distribue des instantanés préprovisionnés pour le lancement.
- Configurations de licence associées : configuration de ARNs licence License Manager associée à l'AMI dans la région spécifiée.
- Configuration du modèle de lancement — Identifie un modèle de lancement Amazon EC2 à utiliser pour un compte spécifique.
- Définir la version par défaut du modèle de lancement : définit le modèle de lancement Amazon EC2 spécifié comme modèle de lancement par défaut pour le modèle spécifié. Compte AWS

Recette de contenant

Les distributions de conteneurs incluent toujours les informations suivantes :

- Région : région de distribution pour l'image du conteneur spécifiée dans la colonne URI de l'image.
- URI de l'image : URI de l'image du conteneur de sortie qu'Image Builder a distribuée au référentiel Amazon ECR dans la région de destination.

> **Note**
>
> Image Builder affiche une ligne par destination. L'image de sortie comporte toujours au moins une entrée destinée à être distribuée au compte qui a créé l'image. Les destinations supplémentaires peuvent inclure des distributions entre les régions Comptes AWS, ou AWS Organizations. Pour de plus amples informations, veuillez consulter Gérer les paramètres de distribution d'Image Builder.

Onglet Workflow

Les flux de travail définissent la séquence d'étapes qu'Image Builder exécute lorsqu'il crée une nouvelle image. Toutes les images ont des flux de travail de création, de test et de distribution. L'onglet Workflow affiche les flux de travail applicables exécutés par Image Builder pour votre image.

Types de flux de travail de filtrage

Image Builder affiche initialement le résumé du flux de travail de création ou d'importation et les étapes du flux de travail par défaut. Cependant, le filtre de flux de travail affiche tous les flux de travail en cours ou terminés pour votre image. Pour afficher un autre flux de travail, sélectionnez-le dans la liste.

Les flux de travail d'images qui produisent une sortie AMI peuvent comporter des flux de travail de création, d'importation ou de test. Les flux de travail de conteneurs qui produisent des sorties de conteneurs peuvent avoir des flux de travail de création, de test ou de distribution.

> **Note**
>
> Si le flux de travail n'a pas encore démarré, il n'apparaît pas dans la liste. Par exemple, si la création de votre image dans laquelle les flux de travail de génération et de test sont configurés vient de démarrer, le flux de travail de génération est le seul type de flux de travail qui apparaît dans la liste. Lorsque le flux de travail de test commence, Image Builder l'ajoute à la liste.

Après le filtre de flux de travail, le flux de travail sélectionné affiche un résumé de l'exécution qui inclut les détails suivants pour chaque type de flux de travail :

État du flux de travail

État d'exécution actuel de ce flux de travail. Les valeurs peuvent inclure les éléments suivants :

- En attente
- Ignoré
- En cours d'exécution
- Terminé
- Échec
- R ollback-in-progress
- Annulation terminée

ID d'exécution

Identifiant unique attribué par Image Builder pour suivre les ressources d'exécution chaque fois qu'il exécute un flux de travail.

Démarrer

Horodatage du démarrage de l'instance d'exécution de ce flux de travail.

Fin

Horodatage auquel cette instance d'exécution du flux de travail s'est terminée.

Nombre total d'étapes

Nombre total d'étapes du flux de travail. Cela doit être égal à la somme du nombre d'étapes réussies, ignorées et échouées.

Étapes réussies

Nombre d'exécution correspondant au nombre d'étapes du flux de travail exécutées avec succès.

Les étapes ont échoué

Nombre d'exécution correspondant au nombre d'étapes du flux de travail qui ont échoué.

Étapes ignorées

Nombre d'exécution correspondant au nombre d'étapes du flux de travail qui ont été ignorées.

Les détails de la liste suivante indiquent l'état actuel de toutes les étapes de cette instance d'exécution du flux de travail. Image Builder affiche les mêmes détails pour tous les types d'images.

Étape #

Un nombre qui représente l'ordre dans lequel Image Builder exécute les étapes du flux de travail.

ID de l'étape

Identifiant unique pour l'étape du flux de travail, attribué lors de l'exécution.

État de l'étape

État d'exécution actuel de l'étape de flux de travail spécifiée.

État du rollback

État actuel de la restauration en cas d'échec de cette instance d'exécution du flux de travail.

Nom de l'étape

Nom de l'étape de flux de travail spécifiée.

Démarrer

Horodatage du début de l'étape spécifiée pour cette instance d'exécution du flux de travail.

Fin

Horodatage de fin de l'étape spécifiée pour cette instance d'exécution du flux de travail.

Onglet Résultats de sécurité

Si vous avez activé le scan, l'onglet Résultats de sécurité affiche les résultats relatifs aux vulnérabilités et expositions courantes (CVE). Amazon Inspector a identifié ces résultats sur l'instance de test lancée par Image Builder pour créer votre nouvelle image. Pour garantir qu'Image Builder capture les résultats de votre image, vous devez configurer la numérisation comme suit :

1. Activez les scans Amazon Inspector pour votre compte. Pour plus d'informations, consultez Getting started with Amazon Inspector dans le guide de l'utilisateur d'Amazon Inspector.
2. Activez les résultats de sécurité pour le pipeline qui crée cette image. Lorsque vous activez les résultats de sécurité pour votre pipeline, Image Builder enregistre un instantané des résultats avant de mettre fin à l'instance de test. Pour de plus amples informations, consultez Configurez les scans de sécurité pour les images d'Image Builder dans Console de gestion AWS.

L'onglet Résultats de sécurité inclut les informations suivantes pour chaque vulnérabilité identifiée par Amazon Inspector pour votre image.

Sévérité

Niveau de gravité du résultat de la CVE. Les valeurs sont les suivantes :

- Non trié
- Informationnel
- Faible
- Medium
- Élevée
- Critique

ID de résultat

Identifiant unique du résultat CVE détecté par Amazon Inspector pour votre image lors de la numérisation de l'instance de test. L'ID est lié à la page Résultats de sécurité > Par vulnérabilité. Pour de plus amples informations, veuillez consulter Gérez les résultats de sécurité pour les images d'Image Builder dans Console de gestion AWS.

Source

Source des informations de vulnérabilité pour la découverte de la CVE.

Âge

Nombre de jours écoulés depuis que le résultat a été observé pour la première fois pour votre image.

Note de l'Inspecteur

Le score attribué par Amazon Inspector à la recherche CVE.

Onglet Balises

L'onglet Tags affiche tous les tags que vous avez définis pour votre image.

Pour plus d'informations sur la politique relative aux images, consultez le AWS CLI

L'exemple suivant montre comment obtenir les détails d'une politique relative aux images à l'aide de son Amazon Resource Name (ARN).

```
aws imagebuilder get-image-policy --image-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/example-image/2019.12.02
```

Créez des images personnalisées avec Image Builder

Vous pouvez créer une nouvelle image Image Builder de différentes manières. Par exemple, vous pouvez utiliser l'une des méthodes suivantes pour créer une image avec le Console de gestion AWS ou AWS CLI. Vous pouvez également utiliser l'action CreateImageAPI ou exécuter un pipeline de génération pour créer l'image. Pour la demande de SDK associée à l'action d'API, vous pouvez vous référer au lien Voir aussi correspondant à cette commande dans le manuel de référence de l'API EC2 Image Builder.

Console de gestion AWS

Pour créer une nouvelle image à partir d'un pipeline existant, vous pouvez exécuter le pipeline manuellement, comme suit. Vous pouvez également utiliser l'assistant de pipeline pour créer une nouvelle image à partir de zéro. Voir Assistant de pipeline : création d'une AMI ouAssistant de pipeline : création d'une image de conteneur, selon le type d'image que vous souhaitez créer.

1. Ouvrez la console https://console.aws.amazon.com/imagebuilder/EC2 Image Builder à l'adresse.
2. Choisissez Image pipelines dans le volet de navigation.
3. Cochez la case à côté du nom du pipeline que vous souhaitez exécuter.
4. Pour créer l'image, sélectionnez Exécuter le pipeline dans le menu Actions. Cela démarre le pipeline.

Vous pouvez également définir un calendrier pour exécuter votre pipeline ou utiliser Amazon EventBridge pour exécuter votre pipeline en fonction des règles que vous configurez.

AWS CLI

Avant d'exécuter la create-image commande dans le AWS CLI, vous devez créer les ressources suivantes si elles n'existent pas déjà :

Ressources requises

- Recette — Vous devez spécifier exactement une recette pour votre image, comme suit :

Recette d'images

Spécifiez le nom de ressource Amazon (ARN) pour votre ressource de recette d'image à l'aide du `--image-recipe-arn` paramètre.

Recette de contenant

Spécifiez l'ARN de votre ressource de recette de conteneur à l'aide du `--container-recipe-arn` paramètre.

- Configuration de l'infrastructure — Spécifiez l'ARN de votre ressource de configuration d'infrastructure à l'aide du `--infrastructure-configuration-arn` paramètre.

Vous pouvez également spécifier l'une des ressources suivantes dont votre image a besoin :

Ressources et configuration facultatives

- Configuration de distribution — Par défaut, Image Builder distribue la ressource d'image de sortie à votre compte dans la région où vous exécutez la create-image commande. Pour fournir des destinations ou une configuration supplémentaires pour votre distribution, spécifiez l'ARN de votre ressource de configuration de distribution à l'aide du `--distribution-configuration-arn` paramètre.
- Numérisation d'images : pour configurer des instantanés correspondant aux résultats d'Amazon Inspector sur votre instance de test d'image ou de conteneur, utilisez le `--image-scanning-configuration` paramètre. Pour les images de conteneurs, vous spécifiez également le référentiel ECR qu'Amazon Inspector utilise pour ses scans.
- Tests d'image : pour supprimer la phase de test Image Builder, utilisez le `--image-tests-configuration` paramètre. Vous pouvez également définir un délai d'expiration pour sa durée d'exécution.
- Balises d'image : utilisez le `--tags` paramètre pour ajouter des balises à votre ressource d'image de sortie.
- Flux de travail d'images : si vous ne spécifiez aucun flux de création ou de test, Image Builder crée votre image avec son flux de travail d'image par défaut. Pour spécifier les flux de travail que vous avez créés, utilisez le `--workflows` paramètre.

> **Note**
>
> Si vous spécifiez des flux de travail d'image, vous devez également fournir le nom ou l'ARN du rôle IAM qu'Image Builder utilise pour exécuter vos actions de flux de travail dans le `--execution-role` paramètre.

L'exemple suivant montre comment créer une image à l'aide de la commande create-image AWS CLI . Pour plus d'informations, consultez la référence de la commande AWS CLI .

Exemple : créer une image de base avec une distribution par défaut

```
aws imagebuilder create-image --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/simple-recipe-linux/1.0.0 --infrastructure-
configuration-arn arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/simple-infra-config-linux
```

Sortie :

```
{
"requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
"imageVersionList": [
    {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/simple-recipe-
linux/1.0.0",
        "name": "simple-recipe-linux",
        ...
    }
]
}
```

Annuler la création d'image à partir du AWS CLI

Pour annuler une création d'image en cours, utilisez la cancel-image-creation commande suivante :

```
aws imagebuilder cancel-image-creation --image-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-recipe/2019.12.03/1
```

Importez et exportez des images de machines virtuelles avec Image Builder

Lorsque vous exportez une machine virtuelle depuis son environnement de virtualisation, ce processus crée un ensemble d'un ou plusieurs fichiers de conteneur de disque qui agissent comme des instantanés de l'environnement, des paramètres et des données de votre machine virtuelle. Vous pouvez utiliser ces fichiers pour importer votre machine virtuelle et l'utiliser comme image de base pour vos recettes d'images. Pour exporter, vous pouvez créer des fichiers de disque de machine virtuelle en sortie à partir de votre création d'image personnalisée et distribuer les fichiers.

Image Builder prend en charge les formats de fichiers suivants pour les conteneurs de disques de votre machine virtuelle :

- Archive de virtualisation ouverte (OVA)
- Disque de machine virtuelle (VMDK)
- Disque dur virtuel (VHD/VHDX)
- Raw

L'importation utilise les disques pour créer une image Amazon Machine Image (AMI) et une ressource d'image Image Builder, l'une ou l'autre pouvant servir d'image de base pour votre recette d'image personnalisée. Les disques de machine virtuelle doivent être stockés dans des compartiments S3 pour l'importation. Vous pouvez également effectuer une importation à partir d'un instantané EBS existant.

Dans la console Image Builder, vous pouvez importer l'image directement, puis utiliser l'image de sortie ou l'AMI dans vos recettes, ou vous pouvez spécifier des paramètres d'importation lorsque vous créez votre recette ou version de recette. Pour plus d'informations sur l'importation dans le cadre de votre recette d'image, consultezConfiguration de l'importation de machines virtuelles.

Importer une machine virtuelle dans Image Builder

Image Builder s'intègre à l' Import/Export API Amazon EC2 VM pour permettre au processus d'importation de s'exécuter de manière asynchrone en arrière-plan. Image Builder fait référence à l'ID de tâche issu de l'importation de la machine virtuelle pour suivre sa progression et crée une ressource image Image Builder en sortie. Cela vous permet de référencer la ressource image Image Builder dans vos recettes avant la fin de l'importation de la machine virtuelle.

Console

Pour importer une machine virtuelle à l'aide de la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder/.
2. Choisissez Images dans le volet de navigation.
3. Pour ouvrir la boîte de dialogue d'importation, choisissez Importer une image.
4. Entrez les informations générales suivantes :
   - Spécifiez un nom unique pour votre image.
   - Spécifiez une version pour l'image de base. Utilisez le format suivant : *major.minor.patch*.
5. Choisissez le type d'importation : importation de machines virtuelles.
6. Fournissez des détails pour chacune des sections suivantes sur la page Importer une image. Choisissez ensuite Importer une image lorsque vous avez terminé.

Système d'exploitation de l'image de base

1. Sélectionnez l'option Système d'exploitation d'image (OS) correspondant à votre plate-forme VM OS.
2. Sélectionnez la version du système d'exploitation qui correspond à la version de votre machine virtuelle dans la liste.

Configuration de l'importation de machines virtuelles

1. Lorsque vous exportez votre machine virtuelle depuis son environnement de virtualisation, ce processus crée un ensemble d'un ou plusieurs fichiers de conteneur de disque. Ils agissent comme des instantanés de l'environnement, des paramètres et des données de votre machine virtuelle. Vous pouvez utiliser ces fichiers pour importer votre machine virtuelle en tant qu'image de base pour votre recette d'image. Pour plus d'informations sur l'importation VMs dans Image Builder, consultezImportation et exportation d'images de machines virtuelles.

   Pour spécifier l'emplacement de votre source d'importation, procédez comme suit :

Source d'importation

Spécifiez la source du premier conteneur de disque ou d'instantané d'image de machine virtuelle à importer dans la section Conteneur de disque 1.

a. Source : il peut s'agir d'un compartiment S3 ou d'un instantané EBS.
b. Sélectionnez l'emplacement S3 du disque : entrez l'emplacement dans Amazon S3 où vos images de disque sont stockées. Pour rechercher l'emplacement, choisissez Parcourir S3.
c. Pour ajouter un conteneur de disques, choisissez Ajouter un conteneur de disques.

2. Rôle IAM

Pour associer un rôle IAM à la configuration d'importation de votre machine virtuelle, sélectionnez le rôle dans la liste déroulante des rôles IAM ou choisissez Créer un nouveau rôle pour en créer un nouveau. Si vous créez un nouveau rôle, la page de console IAM Roles s'ouvre dans un onglet distinct.

3. Réglages avancés — optionnel

Les paramètres suivants sont facultatifs. Ces paramètres vous permettent de configurer le chiffrement, les licences, les balises, etc., pour l'image de base créée par l'importation.

Architecture d'image de base

Pour spécifier l'architecture de la source d'importation de votre machine virtuelle, sélectionnez une valeur dans la liste Architecture.

Chiffrement

Si les images de disque de votre machine virtuelle sont chiffrées, vous devez fournir une clé à utiliser pour le processus d'importation. Pour spécifier une clé KMS pour l'importation, sélectionnez une valeur dans la liste Chiffrement (clé KMS). La liste contient les clés KMS auxquelles votre compte a accès dans la région actuelle.

Gestion des licences

Lorsque vous importez une machine virtuelle, le processus d'importation détecte automatiquement le système d'exploitation de la machine virtuelle et applique la licence appropriée à l'image de base. Selon la plate-forme de votre système d'exploitation, les types de licence sont les suivants :

- Licence incluse : une AWS licence adaptée à votre plateforme est appliquée à votre image de base.
- Apportez votre propre licence (BYOL) : conserve la licence de votre machine virtuelle, le cas échéant.

Pour associer des configurations de licence créées avec AWS License Manager à votre image de base, sélectionnez-les dans la liste des noms de configuration de licence. Pour plus d'informations sur License Manager, consultez Working with AWS License Manager

> **Note**
>
> - Les configurations de licence contiennent des règles de licence basées sur les termes de vos contrats d'entreprise.
> - Linux ne prend en charge que les licences BYOL.

Balises (image de base)

Les balises utilisent des paires clé-valeur pour attribuer du texte consultable à votre ressource Image Builder. Pour spécifier des balises pour l'image de base importée, entrez des paires clé-valeur à l'aide des cases Clé et Valeur.

Pour ajouter une balise, choisissez Ajouter une balise. Pour supprimer une balise, choisissez Remove tag (Supprimer une balise).

AWS CLI

Pour importer une machine virtuelle depuis des disques vers une AMI et créer une ressource d'image Image Builder à laquelle vous pouvez immédiatement faire référence, procédez comme suit à partir du AWS CLI :

1. Lancez une importation de machine virtuelle, avec la Import/Export import-image commande Amazon EC2 VM dans le AWS CLI. Notez l'ID de tâche renvoyé dans la réponse à la commande. Vous en aurez besoin à l'étape suivante. Pour plus d'informations, consultez la section Importation d'une machine virtuelle sous forme d'image à l'aide de VM Import/Export dans le guide de l' Import/Export utilisateur de la machine virtuelle.

2. Créer un fichier JSON d'entrée CLI

   Pour rationaliser la import-vm-image commande Image Builder utilisée dans le AWS CLI, nous créons un fichier JSON contenant toutes les configurations d'importation que nous voulons transmettre à la commande.

   > **Note**
   >
   > La convention de dénomination des valeurs de données dans le fichier JSON suit le modèle spécifié pour les paramètres de demande d'opération de l'API Image Builder. Pour consulter les paramètres de demande d'opération d'API, consultez l'ImportVmImageopération dans le manuel EC2 Image Builder API Reference. Pour fournir les valeurs de données sous forme de paramètres de ligne de commande, reportez-vous aux noms de paramètres spécifiés dans la AWS CLI Command Reference. À la import-vm-image commande Image Builder en tant qu'options.

   Voici un résumé des paramètres que nous indiquons dans cet exemple :

   - name (chaîne, obligatoire) : nom de la ressource d'image Image Builder à créer en sortie de l'importation.
   - SemanticVersion <major>(chaîne, obligatoire) — Version sémantique de l'image de sortie qui spécifie la version au format suivant, avec des valeurs numériques à chaque position pour indiquer une version spécifique :. <minor>. <patch>. Par exemple, `1.0.0`. Pour en savoir plus sur le versionnement sémantique des ressources Image Builder, consultez. Versionnage sémantique dans Image Builder
   - description (chaîne) — Description de la recette d'image.
   - platform (string, obligatoire) : plate-forme du système d'exploitation de la machine virtuelle importée.

- vmImportTaskId (chaîne, obligatoire) — Le `ImportTaskId` (AWS CLI) issu du processus d'importation de la EC2 machine virtuelle Amazon. Image Builder surveille le processus d'importation pour intégrer l'AMI qu'il crée et créer une ressource d'image Image Builder qui peut être utilisée immédiatement dans des recettes.
- tags (chaîne de caractères) — Les tags sont des paires clé-valeur associées aux ressources d'importation. Jusqu'à 50 paires clé-valeur sont autorisées.

Enregistrez le fichier sous`import-vm-image.json`, pour l'utiliser dans la import-vm-image commande Image Builder.

```
{
    "name": "example-request",
    "semanticVersion": "1.0.0",
    "description": "vm-import-test",
    "platform": "Linux",
    "vmImportTaskId": "import-ami-01ab234567890cd1e",
    "tags": {
     "Usage": "VMIE"
    }
}
```

3. Importer l'image

Exécutez la import-vm-image commande en utilisant le fichier que vous avez créé en entrée :

```
aws imagebuilder import-vm-image --cli-input-json file://import-vm-image.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Distribuez des disques de machine virtuelle à partir de votre image créée à partir du AWS CLI

Vous pouvez configurer la distribution des fichiers au format de disque de machine virtuelle pris en charge dans les compartiments S3 des régions cibles dans le cadre de votre processus de création d'image habituel, à l'aide des configurations de distribution d'Image Builder dans le AWS CLI. Pour de plus amples informations, veuillez consulter Exemple : créez des paramètres de distribution pour les disques de sortie des machines virtuelles à partir du AWS CLI.

Importez des images de disque ISO Windows vérifiées avec Image Builder

Un fichier ISO du système d'exploitation Windows est un fichier image disque qui contient le package d'installation complet pour une version spécifique du système d'exploitation Windows. Microsoft fournit des fichiers ISO officiels du système d'exploitation Windows à télécharger, soit directement depuis son site Web, soit par l'intermédiaire de revendeurs agréés. Pour éviter les logiciels malveillants potentiels ou les versions non autorisées, procurez-vous les fichiers ISO auprès d'une source fiable et légitime.

EC2 Image Builder utilise le flux de travail `build-image-from-iso` d'importation pour importer le fichier disque ISO et créer un volume secondaire à partir de celui-ci. Une fois la configuration terminée, Image Builder prend un instantané du volume créé lors de l'importation et l'utilise pour créer une Amazon Machine Image (AMI).

Considérations de sécurité pour l'importation d'images de disque ISO

L'image disque ISO que vous fournissez fournit le système d'exploitation de l'AMI de sortie. Lors de l'importation, Image Builder monte l'ISO et applique l'image Windows qu'elle contient.

Le processus d'importation exécute des exécutables qu'il résout à partir de l'ISO et de l'image Windows appliquée. Ces exécutables s'exécutent avec les privilèges SYSTEM sur l'instance de génération, sous le rôle de profil d'instance de la configuration d'infrastructure que vous spécifiez pour l'importation.

Étant donné que l'ISO fournit le système d'exploitation, vous êtes responsable de l'obtention d'un support authentique. Une image ISO provenant d'une source non fiable peut affecter l'AMI de sortie et toutes les instances que vous lancez à partir de celle-ci. Procurez-vous des fichiers ISO auprès de Microsoft ou d'un revendeur agréé.

L'autorisation de démarrer une importation dans une configuration d'infrastructure inclut la possibilité de s'exécuter sous le rôle de profil d'instance de cette configuration. Image Builder évalue le `iam:PassRole` moment où vous créez la configuration de l'infrastructure, et non lorsque vous démarrez l'importation. Nous vous recommandons d'accorder l'autorisation de démarrer les importations uniquement aux principaux auxquels vous faites confiance avec le rôle de profil d'instance.

Pour de plus amples informations, veuillez consulter Protection des données et AWS modèle de responsabilité partagée dans Image Builder.

Systèmes d'exploitation pris en charge pour l'importation d'images disque ISO

Image Builder prend en charge les images de disque ISO du système d'exploitation Windows suivantes :

- Windows 11 Enterprise version 25H2 (x64)
- Windows 11 Enterprise version 24H2 (x64)
- Windows 11 Entreprise version 23H2 (x64)

Image Builder ne prend pas en charge les images de disque ISO du système d'exploitation Windows suivantes :

- Long-Term Images du canal de maintenance (LTSC)
- Images de disque ISO créées à partir de l'outil de création Windows Media
- Images d'évaluation

Conditions requises pour importer une image disque ISO

> **Note**
>
> Au cours du processus d'importation, l'instance de compilation télécharge AWS les pilotes, EC2Launch v2 et l'agent Systems Manager depuis Amazon S3, et Microsoft Defender depuis l'Internet public. Ces composants sont préconfigurés sur l'AMI de sortie. L'instance de génération communique également avec les API Amazon EC2, Amazon CloudWatch Logs et SSM.

L'instance de compilation télécharge AWS les pilotes depuis un compartiment Amazon S3 régional situé au même Région AWS endroit que l'importation. Comme ce trafic reste dans la région, le point de terminaison Amazon S3 Gateway est suffisant pour télécharger les pilotes dans n'importe quelle région.
Cependant, Microsoft Defender nécessite un accès public à Internet pour le téléchargement, quelle que soit la région. Si l'instance de compilation n'a pas accès à Internet, l'importation réussit toujours mais Microsoft Defender n'est pas installé sur l'AMI de sortie.
Pour obtenir la liste des points de terminaison VPC et des URL S3 minimaux requis, consultez. Configuration réseau minimale requise pour un VPC privé

Pour importer une image disque ISO, vous devez remplir les conditions suivantes :

- Le système d'exploitation de l'image disque doit être compatible avec Image Builder. Pour obtenir la liste des systèmes d'exploitation pris en charge, consultezSystèmes d'exploitation pris en charge pour l'importation d'images disque ISO.
- Pour vous assurer de pouvoir importer votre image ISO, téléchargez-la depuis le centre d'administration Microsoft 365.
- Vous devez télécharger votre fichier de disque ISO sur Amazon S3 au même Région AWS endroit Compte AWS et à l'endroit où l'importation s'effectue avant de pouvoir lancer le processus d'importation.
- L'extension de fichier fait la distinction entre majuscules et minuscules pour le processus d'importation et doit l'être `.ISO`. Si l'extension de votre fichier est en minuscules, exécutez l'une des commandes suivantes pour le renommer :

  Command

  ```
  aws s3 cp s3://amzn-s3-demo-bucket/Win11_24H2_English.iso s3://amzn-s3-demo-bucket/Win11_24H2_English.ISO
  ```

  PowerShell

  ```
  Copy-S3Object -BucketName amzn-s3-demo-bucket -Key Win11_24H2_English.iso -DestinationKey Win11_24H2_English.ISO
  ```

- Les licences Microsoft ne sont pas automatiquement incluses dans l'importation. Vous devez apporter votre propre licence (BYOL). Pour plus d'informations sur les licences pour les logiciels

Microsoft, consultez la section Licences sur la page des questions fréquemment posées par Amazon Web Services et Microsoft.

- Le processus d'importation utilise deux rôles IAM distincts :

  Rôle d'exécution

  Ce rôle permet à Image Builder d'appeler Services AWS en votre nom pendant le processus d'importation.

  > **Important**
  >
  > Nous vous recommandons de ne pas transmettre le rôle AWSServiceRoleForImageBuilder lié à un service en tant que rôle d'exécution. Créez plutôt un rôle IAM personnalisé et associez la politique EC2ImageBuilderExecutionPolicy AWS gérée. Cette politique accorde les mêmes autorisations dont Image Builder a besoin pour appeler Services AWS en votre nom. L'utilisation d'un rôle personnalisé vous permet de contrôler totalement les autorisations utilisées par Image Builder. Il permet également de maintenir vos politiques de contrôle des services (SCP) et vos politiques de contrôle des ressources (RCP) en vigueur pour les opérations qu'Image Builder effectue en votre nom.

  Rôle du profil d'instance

  Ce rôle autorise les actions que le service effectue sur l'instance EC2. Vous pouvez spécifier un rôle de profil d'instance dans la ressource de configuration de votre infrastructure. Associez les politiques gérées suivantes à votre rôle de profil d'instance afin de vous assurer que vous disposez de toutes les autorisations nécessaires au processus d'importation :

  - EC2InstanceProfileForImageBuilder
  - AmazonSSMManagedInstanceCore

  La EC2InstanceProfileForImageBuilder politique inclut l'`s3:GetObject`autorisation que l'instance de build utilise pour télécharger votre fichier ISO. Cet accès s'applique aux objets dotés d'une extension de fichier ISO dans votre propre compte. Par conséquent, vous n'avez pas besoin d'accorder d'autorisations Amazon S3 supplémentaires pour l'ISO.

> **Note**
>
> Nous vous recommandons d'accorder au rôle de profil d'instance uniquement les autorisations requises pour l'importation. Le processus d'importation exécute des exécutables à partir de l'ISO et de l'image Windows appliquée avec les privilèges SYSTEM dans le cadre de ce rôle. Toutes les autorisations supplémentaires que vous accordez s'appliquent également à ces exécutables.

Pour plus d'informations, consultez Gérer la configuration de l'infrastructure d'Image Builder, EC2InstanceProfileForImageBuilder policy et Considérations de sécurité pour l'importation d'images de disque ISO.

Paramètres d'importation facultatifs

Vous pouvez éventuellement configurer les paramètres suivants lorsque vous importez une image disque ISO. Ces paramètres contrôlent le démarrage sécurisé, les données UEFI et la sélection de l'index d'image pour l'image importée.

Démarrage sécurisé

Le démarrage sécurisé est une fonctionnalité de sécurité UEFI qui garantit que seuls les logiciels fiables s'exécutent pendant le processus de démarrage. Par défaut, le démarrage sécurisé est activé pour les importations d'images disque ISO. Vous pouvez désactiver le démarrage sécurisé si vous devez utiliser des pilotes non signés personnalisés à des fins de test ou de compatibilité avec les applications existantes.

Données UEFI personnalisées

Vous pouvez fournir un blob de données UEFI personnalisé sous forme de Base64-encoded chaîne à utiliser pendant le processus de démarrage au lieu des données UEFI par défaut générées par Image Builder. Vous pouvez spécifier des données UEFI personnalisées uniquement lorsque le démarrage sécurisé est activé (par défaut). Les données peuvent atteindre 64 Ko au maximum.

Vous pouvez inspecter et modifier les données UEFI à l'aide de l'outil https://github.com/awslabs/python-uefivars python-uefivars sur le site Web. GitHub Pour plus d'informations, consultez la section Variables UEFI pour les instances Amazon EC2.

Index des images

Un fichier ISO Windows peut contenir un `.wim` fichier comportant plusieurs index d'images, chaque index représentant une édition Windows différente (Home ou Pro, par exemple). Par défaut, Image Builder utilise le premier index d'image valide du fichier ISO. Vous pouvez spécifier un index d'image basé sur une seule base pour sélectionner une édition spécifique à partir d'un fichier ISO multi-édition.

Importer une image disque ISO dans Image Builder

Avant de commencer le processus d'importation, assurez-vous que vous avez rempli toutes les conditionsConditions préalables.

Le processus d'importation installe les logiciels et pilotes suivants sur votre image :

- EC2Launch v2
- AWS Systems Manager agent
- AWS pilote NVMe
- AWS pilote réseau ENA
- AWS Pilote série PCI
- Pilote utilitaire Windows EC2
- Kit de mise à jour Microsoft Defender

Le processus d'importation apporte les mises à jour de configuration suivantes à votre image :

- Configure le système pour qu'il utilise le serveur Amazon Time.

Choisissez un onglet pour afficher les étapes d'importation correspondant à votre méthode préférée :

Console

Pour importer une image disque ISO à l'aide de la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez Images dans le volet de navigation.

3. Pour ouvrir la boîte de dialogue d'importation, choisissez Importer une image.
4. Entrez les informations générales suivantes :

   - Spécifiez un nom unique pour votre image.
   - Spécifiez une version pour l'image de base. Utilisez le format suivant : *major.minor.patch*.
5. Choisissez le type d'importation : importation ISO.
6. Entrez les informations de configuration d'importation ISO suivantes. Choisissez ensuite Importer une image lorsque vous avez terminé.

   - URI S3 — Entrez l'emplacement où votre fichier de disque ISO est stocké. Pour rechercher le fichier, choisissez Parcourir S3.
   - Rôle IAM : pour associer un rôle IAM à votre configuration d'importation, sélectionnez le rôle dans la liste déroulante des rôles IAM ou choisissez Créer un nouveau rôle pour en créer un nouveau. Si vous créez un nouveau rôle, la page de console IAM Roles s'ouvre dans un onglet distinct.

     Vous pouvez spécifier le rôle AWSServiceRoleForImageBuilder lié au service, ou vous pouvez spécifier votre propre rôle personnalisé pour l'accès au service.
7. Vous pouvez éventuellement configurer les paramètres avancés suivants pour l'importation. Pour plus d'informations sur ces paramètres, consultez la page Paramètres d'importation facultatifs.

   - Démarrage sécurisé : le démarrage sécurisé est activé par défaut. Pour désactiver le démarrage sécurisé pour l'image importée, désactivez la case à cocher Démarrage sécurisé.
   - Données UEFI personnalisées : pour fournir un blob de données UEFI personnalisé, entrez la chaîne. Base64-encoded Cette option n'est disponible que lorsque le démarrage sécurisé est activé.
   - Index des images : pour sélectionner une édition spécifique de Windows à partir d'un fichier ISO multiédition, entrez l'index des images basé sur une seule base.
8. Vous pouvez éventuellement ajouter des balises à votre ressource d'images Image Builder. L'ajout de balises ici n'ajoute pas les balises à votre AMI.
9. La configuration de l'infrastructure ISO définit les paramètres de l'instance qu'Image Builder lance pour héberger le processus d'importation. Vous pouvez utiliser une configuration d'infrastructure créée par Image Builder en fonction des paramètres par défaut du service,

ou vous pouvez utiliser une configuration d'infrastructure existante. Pour de plus amples informations, veuillez consulter Gérer la configuration de l'infrastructure d'Image Builder.

Pour créer une nouvelle configuration d'infrastructure, choisissez Créer une configuration d'infrastructure. Cela s'ouvre dans un onglet séparé. Après avoir créé la nouvelle ressource, vous pouvez revenir à la configuration d'importation et choisir Utiliser la configuration d'infrastructure existante.

10. Pour démarrer le processus d'importation, choisissez Importer une image.

Une fois l'importation terminée, votre image apparaît dans la liste des images que vous possédez. Pour en savoir plus, consultez Liste des images.

AWS CLI

L'exemple suivant montre comment importer une image à partir d'un fichier disque ISO et créer une AMI à partir de celui-ci avec le AWS CLI.

Voici un résumé des paramètres que nous spécifions dans cet exemple :

- name (chaîne, obligatoire) : nom de la ressource d'image Image Builder à créer en sortie de l'importation.
- <major>SemanticVersion (chaîne, obligatoire) — Version sémantique de l'image de sortie qui spécifie la version au format suivant, avec des valeurs numériques à chaque position pour indiquer une version spécifique :. <minor>. <patch>. Par exemple, `1.0.0`. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder
- description (chaîne) — Description de la recette imagée.
- ExecutionRole (chaîne) : nom ou Amazon Resource Name (ARN) du rôle IAM qui autorise Image Builder à effectuer des actions de flux de travail pour importer une image à partir d'un fichier ISO Microsoft. Nous vous recommandons de spécifier un rôle personnalisé auquel la stratégie EC2ImageBuilderExecutionPolicy AWS gérée est attachée.
- platform (chaîne, obligatoire) — Plateforme du système d'exploitation pour l'image disque ISO. Les valeurs valides incluent`Windows`.
- OSVersion (chaîne, obligatoire) — Version du système d'exploitation pour l'image disque ISO. Les valeurs valides incluent`Microsoft Windows 11`.

- infrastructure ConfigurationArn (chaîne, obligatoire) : nom de ressource Amazon (ARN) de la ressource de configuration d'infrastructure utilisée pour lancer l'instance EC2 sur laquelle l'image ISO est créée.
- uri (chaîne, obligatoire) — URI du fichier disque ISO stocké dans Amazon S3.
- register ImageOptions (object) — Configure les paramètres Secure Boot et UEFI pour l'image importée. Contient les champs suivants :
  - secure BootEnabled (boolean) — Spécifie si le démarrage sécurisé est activé pour l'AMI de sortie. La valeur par défaut est `true`. Pour désactiver le démarrage sécurisé pour les pilotes personnalisés non signés, définissez cette valeur sur`false`.
  - UEFIdata (chaîne) — Base64-encoded Représentation du magasin de variables UEFI non volatil. Vous ne pouvez spécifier ce paramètre que s'il `secureBootEnabled` est défini `true` ou non.
- WindowsConfiguration (objet) : paramètres Windows-specific de configuration pour l'importation ISO. Contient les champs suivants :
  - ImageIndex (entier) : index basé sur 1 qui spécifie l'édition de Windows à installer à partir d'un fichier ISO Windows multi-édition. Une image ISO Windows peut contenir un `.wim` fichier comportant plusieurs index d'images, chacun représentant une édition différente.

```
aws imagebuilder import-disk-image \
    --name "example-iso-disk-import" \
    --semantic-version "1.0.0" \
    --description "Import an ISO disk image" \
    --execution-role "ImageBuilderExecutionRole" \
    --platform "Windows" \
    --os-version "Microsoft Windows 11" \
    --infrastructure-configuration-arn "arn:aws:imagebuilder:us-
east-1:111122223333:infrastructure-configuration/example-infrastructure-
configuration-123456789abc" \
    --uri "s3://amzn-s3-demo-source-bucket/examplefile.ISO" \
    --register-image-options '{"secureBootEnabled": true, "uefiData": "custom-
base64-encoded-uefi-data"}' \
    --windows-configuration '{"imageIndex": 1}'
```

Une fois l'importation terminée, votre image apparaît dans la liste des images que vous possédez. Pour en savoir plus, consultez Liste des images.

PowerShell

L'exemple suivant montre comment importer une image à partir d'un fichier disque ISO et créer une AMI à partir de celui-ci avec PowerShell.

Voici un résumé des paramètres que nous spécifions dans cet exemple :

- name (chaîne, obligatoire) : nom de la ressource d'image Image Builder à créer en sortie de l'importation.
- <major>SemanticVersion (chaîne, obligatoire) — Version sémantique de l'image de sortie qui spécifie la version au format suivant, avec des valeurs numériques à chaque position pour indiquer une version spécifique :. <minor>. <patch>. Par exemple, `1.0.0`. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder
- description (chaîne) — Description de la recette imagée.
- ExecutionRole (chaîne) : nom ou Amazon Resource Name (ARN) du rôle IAM qui autorise Image Builder à effectuer des actions de flux de travail pour importer une image à partir d'un fichier ISO Microsoft. Nous vous recommandons de spécifier un rôle personnalisé auquel la stratégie EC2ImageBuilderExecutionPolicy AWS gérée est attachée.
- platform (chaîne, obligatoire) — Plateforme du système d'exploitation pour l'image disque ISO. Les valeurs valides incluent`Windows`.
- OSVersion (chaîne, obligatoire) — Version du système d'exploitation pour l'image disque ISO. Les valeurs valides incluent`Microsoft Windows 11`.
- infrastructure ConfigurationArn (chaîne, obligatoire) : nom de ressource Amazon (ARN) de la ressource de configuration d'infrastructure utilisée pour lancer l'instance EC2 sur laquelle l'image ISO est créée.
- uri (chaîne, obligatoire) — URI du fichier disque ISO stocké dans Amazon S3.
- register ImageOptions (object) — Configure les paramètres Secure Boot et UEFI pour l'image importée. Contient les champs suivants :
  - secure BootEnabled (boolean) — Spécifie si le démarrage sécurisé est activé pour l'AMI de sortie. La valeur par défaut est `true`. Pour désactiver le démarrage sécurisé pour les pilotes personnalisés non signés, définissez cette valeur sur`false`.
  - UEFIdata (chaîne) — Base64-encoded Représentation du magasin de variables UEFI non volatil. Vous ne pouvez spécifier ce paramètre que s'il `secureBootEnabled` est défini `true` ou non.

- WindowsConfiguration (objet) : paramètres Windows-specific de configuration pour l'importation ISO. Contient les champs suivants :
  - ImageIndex (entier) : index basé sur 1 qui spécifie l'édition de Windows à installer à partir d'un fichier ISO Windows multi-édition. Une image ISO Windows peut contenir un `.wim` fichier comportant plusieurs index d'images, chacun représentant une édition différente.

```
Import-EC2IBDiskImage `
    -Name "example-iso-disk-import" `
    -SemanticVersion "1.0.0" `
    -Description "Import an ISO disk image" `
    -ExecutionRole "ImageBuilderExecutionRole" `
    -Platform "Windows" `
    -OsVersion "Microsoft Windows 11" `
    -InfrastructureConfigurationArn "arn:aws:imagebuilder:us-
east-1:111122223333:infrastructure-configuration/example-infrastructure-
configuration-123456789abc" `
    -Uri "s3://amzn-s3-demo-source-bucket/examplefile.ISO" `
    -RegisterImageOptions_SecureBootEnabled $true `
    -RegisterImageOptions_UefiData "custom-base64-encoded-uefi-data" `
    -WindowsConfiguration_ImageIndex 1
```

Une fois l'importation terminée, votre image apparaît dans la liste des images que vous possédez. Pour en savoir plus, consultez Liste des images.

Lancer une instance depuis l'AMI de sortie

Vous pouvez désormais utiliser l'AMI de sortie comme une AMI normale et lancer des instances à partir de celle-ci. Lorsque vous lancez une instance à partir de l'AMI de sortie, le système d'exploitation Windows exécute Sysprep Specialize pour finaliser la configuration de l'instance.

> **Note**
>
> Les exigences réseau décrites sur cette page s'appliquent uniquement au processus de génération d'importation. Une fois l'importation terminée avec succès, le lancement d'instances à partir de l'AMI de sortie répond aux exigences réseau standard d'Amazon EC2. Une importation réussie ne garantit pas le lancement réussi de l'instance. Votre lancement peut tout de même échouer en fonction de la configuration de votre réseau.

Configuration réseau minimale requise pour un VPC privé

Points de terminaison VPC requis :

Endpoint	Type	Objectif
com.amazonaws.*{region}*.s3	Passerelle	Accédez aux compartiments Amazon S3 (EC2Launch v2, agent SSM, AWS pilotes et votre ISO)
com.amazonaws.*{region}*.ec2	Interface	Créez des instantanés et décrivez les volumes
com.amazonaws.*{region}*.logs	Interface	Écrire les journaux de construction dans CloudWatch Logs
com.amazonaws.*{region}*.ssm	Interface	Appels d'API SMS (SendCommand, DescribeInstanceInformation)
com.amazonaws.*{region}*.ssmmessages	Interface	Canal de données SSM Agent (réception de commandes, envoi de sortie)
com.amazonaws.*{region}*.ec2messages	Interface	Sondage des messages de l'agent SSM

Points de terminaison Amazon S3 consultés lors de l'importation :

- https://ec2-windows-drivers-downloads-*{region}*.s3.dualstack.*{region}*.amazonaws.com/NVMe/Latest/AWSNVMe.zip
- https://ec2-windows-drivers-downloads-*{region}*.s3.dualstack.*{region}*.amazonaws.com/ENA/Latest/AwsEnaNetworkDriver.zip

- `https://ec2-windows-drivers-downloads-{region}.s3.dualstack.{region}.amazonaws.com/AWSPCISerialDriver/Latest/AWSPCISerialDriver.zip`
- `https://ec2-windows-drivers-downloads-{region}.s3.dualstack.{region}.amazonaws.com/EC2WinUtil/Latest/EC2WinUtil.zip`
- `https://amazon-ec2launch-v2-{region}.s3.dualstack.{region}.amazonaws.com/windows/amd64/latest/AmazonEC2Launch.msi`
- `https://amazon-ssm-{region}.s3.{region}.amazonaws.com/latest/windows_amd64/AmazonSSMAgentSetup.exe`

> **Note**
>
> Dans `us-east-1`, le nom du compartiment de pilotes n'a pas de suffixe de région. L'instance de compilation télécharge `ec2-windows-drivers-downloads.s3.dualstack.us-east-1.amazonaws.com` plutôt les pilotes. Dans les deux cas, le compartiment se trouve dans la même région que l'importation, donc le point de terminaison Amazon S3 Gateway est suffisant.

Cela représente la configuration minimale requise pour que l'importation s'effectue dans un sous-réseau privé. En fonction des paramètres de votre VPC et du résultat souhaité, vous aurez peut-être besoin d'une configuration réseau supplémentaire pour autoriser le trafic entre l'instance de build et les points de terminaison du VPC.

Étapes suivantes

Vous pouvez utiliser l'AMI de sortie comme n'importe quelle autre AMI : lancer des instances directement à partir de celle-ci ou l'utiliser comme image de base dans Image Builder pour créer et personnaliser davantage. Pour de plus amples informations, veuillez consulter Créez des images personnalisées avec Image Builder.

Résoudre les problèmes liés aux importations d'images de disque ISO

Si l'importation de votre image disque ISO échoue, vous pouvez utiliser Amazon CloudWatch Logs pour identifier les erreurs d'importation. Image Builder diffuse les journaux de génération

dans CloudWatch les journaux une fois la génération terminée. Pour trouver les journaux de votre importation, utilisez le groupe de journaux et le flux suivants, en les *ImageName* remplaçant par le nom que vous avez donné à votre image :

LogGroup: `/aws/imagebuilder/`*ImageName*

LogStream: *ImageVersion*`/`*ImageBuildVersion*

Pour plus d'informations sur les journaux d'Image Builder dans CloudWatch les journaux, consultezSurveillez les journaux d'Image Builder avec Amazon CloudWatch Logs. Pour obtenir des conseils de dépannage supplémentaires, consultezRésoudre les problèmes liés à Image Builder.

Gestion des résultats de sécurité pour les images d'Image Builder

Lorsque vous activez l'analyse de sécurité avec Amazon Inspector, celle-ci analyse en permanence les images des machines et les instances en cours d'exécution de votre compte pour détecter les vulnérabilités du système d'exploitation et du langage de programmation. Si cette option est activée, l'analyse de sécurité est automatique et Image Builder peut enregistrer un instantané des résultats de votre instance de test lorsque vous créez une nouvelle image. Amazon Inspector est un service payant.

Lorsqu'Amazon Inspector découvre des vulnérabilités dans vos paramètres réseau ou logiciels, il prend les mesures suivantes :

- Vous informe qu'une découverte a été faite.
- Évalue la gravité de la découverte. L'indice de gravité classe les vulnérabilités pour vous aider à hiérarchiser vos résultats et inclut les valeurs suivantes :
  - Non trié
  - Informationnel
  - Faible
  - Medium
  - Élevée
  - Critique
- Fournit des informations sur la découverte et des liens vers des ressources supplémentaires pour plus de détails.
- Propose des conseils de remédiation pour vous aider à résoudre les problèmes à l'origine de cette découverte.

Configurez les scans de sécurité pour les images d'Image Builder dans Console de gestion AWS

Si vous avez activé Amazon Inspector pour votre compte, Amazon Inspector analyse automatiquement les instances EC2 lancées par Image Builder pour créer et tester une nouvelle image. Ces instances ont une courte durée de vie pendant le processus de création et de test, et leurs résultats expirent normalement dès leur arrêt. Pour vous aider à étudier et à corriger les résultats de votre nouvelle image, Image Builder peut éventuellement enregistrer sous forme de capture instantanée tous les résultats identifiés par Amazon Inspector sur votre instance de test pendant le processus de création.

Étape 1 : activer les scans de sécurité Amazon Inspector pour votre compte

Pour activer les analyses de sécurité Amazon Inspector pour votre compte depuis la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez les paramètres d'analyse de sécurité dans le volet de navigation. Cela ouvre la boîte de dialogue d'analyse de sécurité.

   La boîte de dialogue affiche l'état de numérisation de votre compte. Si Amazon Inspector est déjà activé pour votre compte, le statut indique Activé.
3. Suivez les étapes 1 et 2 des instructions pour activer le scan Amazon Inspector.

> **Note**
>
> Amazon Inspector entraîne des frais. Pour plus d'informations, consultez la Tarification d'Amazon Inspector.

Si vous avez activé la numérisation pour votre pipeline, Image Builder prend un instantané des résultats de votre instance de test lorsque vous créez une nouvelle image. De cette façon, vous pouvez accéder aux résultats une fois qu'Image Builder a mis fin à l'instance de test.

Étape 2 : configurez votre pipeline pour enregistrer des instantanés afin de détecter les vulnérabilités

Pour configurer des instantanés de détection de vulnérabilités pour votre pipeline, effectuez les étapes suivantes :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez Image pipelines dans le volet de navigation.
3. Choisissez l'une des méthodes suivantes pour spécifier les détails du pipeline :

   Création d'un nouveau pipeline

   1. Sur la page Pipelines d'images, choisissez Créer un pipeline d'images. Cela ouvre la page Spécifier les détails du pipeline dans l'assistant de pipeline.

   Mettre à jour un pipeline existant

   1. Sur la page Image des pipelines, cliquez sur le lien Nom du pipeline pour le pipeline que vous souhaitez mettre à jour. Cela ouvre la page détaillée du pipeline.

      > **Note**
      >
      > Vous pouvez également cocher la case à côté du nom du pipeline que vous souhaitez mettre à jour, puis choisir Afficher les détails.

   2. Sur la page des détails du pipeline, sélectionnez Modifier le pipeline dans le menu Action. Cela vous amène à la page Modifier le pipeline.
4. Dans la section Général de l'assistant de pipeline ou sur la page Modifier le pipeline, cochez la case Activer l'analyse de sécurité.

   > **Note**
   >
   > Si vous souhaitez désactiver les instantanés ultérieurement, vous pouvez modifier votre pipeline pour désactiver la case à cocher. Cela ne désactive pas l'analyse d'Amazon Inspector pour votre compte. Pour désactiver la numérisation d'Amazon Inspector, consultez la section Désactivation d'Amazon Inspector dans le guide de l'utilisateur d'Amazon Inspector.

Gérez les résultats de sécurité pour les images d'Image Builder dans Console de gestion AWS

Les pages de liste des résultats de sécurité affichent des informations de haut niveau sur les résultats de vos ressources, avec des vues basées sur plusieurs filtres différents que vous pouvez appliquer. Chaque vue inclut les options suivantes en haut pour modifier votre vue :

- Tous les résultats de sécurité : il s'agit de l'affichage par défaut si vous choisissez la page des résultats de sécurité dans le volet de navigation de la console Image Builder.
- Par vulnérabilité : cette vue affiche une liste détaillée de toutes les ressources d'images de votre compte qui contiennent des résultats. L'ID de découverte est lié à des informations plus détaillées sur la découverte. Ces informations apparaissent dans un panneau qui s'ouvre sur le côté droit de la page. Le panneau contient les informations suivantes :
  - Description détaillée de la découverte.
  - Un onglet Recherche de détails. Cet onglet comprend une vue d'ensemble des recherches, les packages concernés, des conseils de correction récapitulatifs, des informations détaillées sur les vulnérabilités et les vulnérabilités associées. L'identifiant de vulnérabilité renvoie à des informations détaillées sur les vulnérabilités dans la base de données nationale sur les vulnérabilités.
  - Un onglet de répartition des scores. Cet onglet inclut une comparaison côte à côte des scores CVSS et Amazon Inspector afin que vous puissiez voir où Amazon Inspector a modifié un score, le cas échéant.
- Par pipeline d'images : cet affichage indique le nombre de résultats pour chaque pipeline d'images de votre compte. Image Builder affiche le nombre de résultats correspondant à un niveau de gravité moyen et élevé, ainsi qu'un total pour tous les résultats. Toutes les données de la liste sont liées, comme suit :
  - La colonne Nom du pipeline d'images renvoie à la page détaillée du pipeline d'images spécifié.
  - Les liens des colonnes du niveau de gravité ouvrent la vue Tous les résultats de sécurité, filtrée en fonction du nom du pipeline d'images et du niveau de gravité associés.

  Vous pouvez également utiliser des critères de recherche pour affiner vos résultats.
- Par image : cette vue indique le nombre de résultats pour chaque image créée dans votre compte. Image Builder affiche le nombre de résultats correspondant à un niveau de gravité moyen et élevé, ainsi qu'un total pour tous les résultats. Toutes les données de la liste sont liées, comme suit :

- La colonne Nom de l'image renvoie à la page détaillée de l'image pour la création d'image spécifiée. Pour de plus amples informations, veuillez consulter Afficher les détails des ressources d'images.
- Les liens des colonnes du niveau de gravité ouvrent la vue Tous les résultats de sécurité, filtrée en fonction du nom de création de l'image et du niveau de gravité associés.

Vous pouvez également utiliser des critères de recherche pour affiner vos résultats.

Image Builder affiche les informations suivantes dans la section Liste des résultats de la vue par défaut Tous les résultats de sécurité.

Sévérité

Le niveau de gravité du résultat de la CVE. Les valeurs sont les suivantes :

- Non trié
- Informationnel
- Faible
- Medium
- Élevée
- Critique

ID de résultat

Identifiant unique du résultat CVE détecté par Amazon Inspector pour votre image lors de la numérisation de l'instance de test. L'ID est lié à la page Résultats de sécurité > Par vulnérabilité.

ARN de l'image

Le nom de ressource Amazon (ARN) de l'image dont la découverte est spécifiée dans la colonne Identifiant de la recherche.

Gazoduc

Le pipeline qui a créé l'image spécifiée dans la colonne Image ARN.

Description

Brève description de la découverte.

Note de l'inspecteur

Le score attribué par Amazon Inspector à la découverte de la CVE.

Assainissement

Liens vers des informations détaillées sur le plan d'action recommandé pour remédier à cette constatation.

Date de publication

Date et heure auxquelles cette vulnérabilité a été ajoutée pour la première fois à la base de données du fournisseur.

Nettoyez les ressources d'Image Builder

Pour éviter des frais imprévus, veillez à nettoyer les ressources et les pipelines que vous avez créés à partir des exemples de ce guide. Pour plus d'informations sur la suppression de ressources dans Image Builder, consultezSupprimer les ressources d'Image Builder obsolètes ou inutilisées.

Gérez les politiques de cycle de vie des images Image Builder

Les politiques de cycle de vie d'EC2 Image Builder automatisent la dépréciation, la désactivation et la suppression des images obsolètes et de leurs ressources associées. Chaque build de pipeline crée une nouvelle version de l'image et distribue les ressources associées. Sans politiques de cycle de vie, les versions antérieures restent dans votre compte jusqu'à ce que vous les supprimiez manuellement.

Les politiques de cycle de vie définissent des règles qui contrôlent quand et comment Image Builder retire les images obsolètes. Les ressources associées incluent les images de sortie distribuées à d'autres Comptes AWS organisations et unités organisationnelles (UO). Ces ressources peuvent couvrir plusieurs types de ressources Régions AWS. Vous spécifiez les étapes du cycle de vie à inclure et la date d'exécution de chaque étape.

Les politiques de cycle de vie activées s'exécutent automatiquement une fois par jour. Image Builder gère le calendrier d'exécution. Pour plus de détails sur l'exécution, consultezComment fonctionne l'exécution des politiques de cycle de vie.

> **Tip**
>
> Pour plus d'informations sur un exemple de politique de cycle de vie déployable avec des documents de politique prêts à l'emploi, consultez l'exemple de cycle de vie sur. GitHub Pour de plus amples informations, veuillez consulter Découvrez des exemples de projets Image Builder sur GitHub.

Avantages de la gestion automatisée du cycle de vie

Gestion automatisée du cycle de vie :

- Automatise le retrait des images et des ressources associées sur l'ensemble des comptes et des régions.
- Élimine les risques de conformité liés au lancement de nouvelles instances avec des images obsolètes.
- Maintient à jour les inventaires d'images en supprimant les versions obsolètes.

- Réduit les coûts d'infrastructure en supprimant les ressources associées lorsque vous supprimez des images.

Économies sur les coûts

L'utilisation d'EC2 Image Builder pour créer des images d'AMI ou de conteneur personnalisées est gratuite. Cependant, la tarification standard s'applique aux autres services utilisés dans le processus. La suppression des images obsolètes et des ressources associées permet de réduire les coûts :

- Réduit le nombre d'images à corriger en supprimant les versions obsolètes.
- Supprime les AMI distribuées et leurs instantanés associés lorsque vous supprimez des ressources d'image AMI, éliminant ainsi les coûts de stockage des instantanés.
- Supprime les ressources sous-jacentes lorsque vous supprimez des ressources d'image de conteneur, éliminant ainsi les coûts de stockage Amazon ECR et les frais de transfert de données.

> **Note**
>
> Avant de configurer les actions de stratégie, vérifiez qu'aucune dépendance en aval ne fait référence à aucune image de référence supprimée par la stratégie. Image Builder n'évalue pas l'impact sur les dépendances en aval, telles que les groupes Auto Scaling ou les modèles de lancement.

Table des matières

- Conditions requises pour la gestion du cycle de vie des images Image Builder
- Répertorier les politiques de gestion du cycle de vie pour les ressources d'image Image Builder
- Afficher les détails des politiques de cycle de vie
- Créez des politiques de cycle de vie
- Comment fonctionne l'exécution des politiques de cycle de vie
- Comment fonctionnent les règles de gestion du cycle de vie pour les ressources d'image Image Builder

Conditions requises pour la gestion du cycle de vie des images Image Builder

Effectuez ces étapes avant de créer des politiques de cycle de vie.

- Créez un rôle IAM qui autorise Image Builder à exécuter des politiques de cycle de vie. Trois possibilités s'offrent à vous :
  - Choisissez Créer un rôle d'exécution du cycle de vie à l'aide des valeurs par défaut du service dans la console Image Builder lorsque vous créez une politique de cycle de vie. Cela associe automatiquement la politique `EC2ImageBuilderLifecycleExecutionPolicy` gérée.
  - Choisissez Créer un nouveau rôle d'exécution du cycle de vie dans la console Image Builder. Cela ouvre IAM avec des paramètres préremplis pour la création de rôles en un clic.
  - Créez le rôle manuellement. Pour obtenir des instructions, veuillez consulter Création d'un rôle IAM pour la gestion du cycle de vie d'Image Builder.
- Créez un rôle IAM dans chaque compte de destination où vous avez distribué les ressources associées entre les comptes. Ce rôle autorise Image Builder à effectuer des actions de cycle de vie sur ce compte. Pour obtenir des instructions, veuillez consulter Création d'un rôle IAM pour la gestion du cycle de vie entre comptes d'Image Builder.

  > **Note**
  >
  > Cette condition préalable ne s'applique pas si vous n'avez accordé des autorisations de lancement que pour une AMI de sortie. Avec les autorisations de lancement, toutes les ressources AMI restent sur votre compte.

- Pour les images de conteneurs, ajoutez la balise `LifecycleExecutionAccess: EC2 Image Builder` à vos référentiels ECR. Cette balise permet à Image Builder d'accéder à l'exécution d'actions de cycle de vie sur les images de conteneurs du référentiel.

Création d'un rôle IAM pour la gestion du cycle de vie d'Image Builder

Créez un rôle de service qui autorise Image Builder à effectuer des actions de cycle de vie sur vos ressources d'images.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Rôles dans le panneau de navigation.

3. Choisissez Créer un rôle.
4. Sélectionnez Politique de confiance personnalisée pour le type d'entité de confiance.
5. Collez la politique de confiance JSON suivante dans la zone de texte de la politique de confiance personnalisée, en remplacement de l'exemple de texte. Cette politique permet à Image Builder d'assumer le rôle des actions relatives au cycle de vie.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "sts:AssumeRole"
            ],
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "imagebuilder.amazonaws.com"
                ]
            }
        }
    ]
}
```

6. Sélectionnez dans EC2ImageBuilderLifecycleExecutionPolicy la liste des politiques gérées, puis choisissez Suivant.

> **Tip**
>
> Filtrez `image` pour rationaliser les résultats.

7. Entrez un nom de rôle.
8. Vérifiez vos paramètres, puis choisissez Créer un rôle.

Création d'un rôle IAM pour la gestion du cycle de vie entre comptes d'Image Builder

Créez ce rôle dans chaque compte de destination sur lequel Image Builder a distribué les ressources associées. Le rôle autorise Image Builder à effectuer des actions de cycle de vie sur ce compte.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Rôles dans le panneau de navigation.
3. Choisissez Créer un rôle.
4. Sélectionnez Politique de confiance personnalisée pour le type d'entité de confiance.
5. Collez la politique de confiance JSON suivante dans la zone de texte de la politique de confiance personnalisée, en remplacement de l'exemple de texte. Cette politique permet à Image Builder d'assumer le rôle des actions relatives au cycle de vie.

> **Note**
>
> Image Builder utilise ce rôle pour le compte du propriétaire du compte de destination. Définissez le `aws:SourceAccount` compte sur lequel Image Builder a distribué les ressources.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "imagebuilder.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "444455556666"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:*:imagebuilder:*:*:image/*/*/*"
                }
```

```
            }
        }
    ]
}
```

6. Sélectionnez dans EC2ImageBuilderLifecycleExecutionPolicy la liste des politiques gérées, puis choisissez Suivant.

> **Tip**
>
> Filtrez `image` pour rationaliser les résultats.

7. Entrez `Ec2ImageBuilderCrossAccountLifecycleAccess` comme nom de rôle. Vous devez utiliser ce nom exact.
8. Vérifiez vos paramètres, puis choisissez Créer un rôle.

Répertorier les politiques de gestion du cycle de vie pour les ressources d'image Image Builder

Vous pouvez obtenir une liste des politiques de gestion du cycle de vie de vos images qui inclut des colonnes détaillées clés sur la page de liste des politiques de cycle de vie du Console de gestion AWS ou avec des commandes ou des actions dans l'API Image Builder SDKs, ou AWS CLI.

Vous pouvez utiliser l'une des méthodes suivantes pour répertorier les ressources relatives à la politique de cycle de vie des images Image Builder dans votre Compte AWS. Pour l'action de l'API, voir le ListLifecyclePoliciesmanuel de référence de l'API EC2 Image Builder. Pour la demande de SDK associée, reportez-vous au lien Voir aussi sur la même page.

Console de gestion AWS

Les informations suivantes sont affichées dans la console pour vos politiques existantes. Vous pouvez sélectionner n'importe quelle colonne pour modifier l'ordre de tri de vos résultats. La liste des politiques est initialement triée par nom de stratégie. Le nom de colonne correspondant à l'ordre de tri actuel est en gras.

Si vous avez plusieurs pages de résultats, les flèches de pagination situées dans le coin supérieur droit du panneau deviennent actives. Vous pouvez filtrer les résultats par nom de politique, statut de la politique, type d'image de sortie et ARN de la ressource d'image à l'aide de la barre de recherche.

- Nom de la stratégie : nom de la stratégie.
- État de la politique : indique si la stratégie est active ou inactive.
- Type : type d'image de sortie distribuée par Image Builder lorsque vous créez une nouvelle version d'image (AMI ou image de conteneur).
- Date de dernière exécution : dernière exécution de la politique de cycle de vie.
- Date de création : date de création de la politique de cycle de vie.
- ARN — Le nom de ressource Amazon (ARN) de la ressource de politique de cycle de vie.

Pour répertorier les politiques de cycle de vie dans le Console de gestion AWS, procédez comme suit :

1. Ouvrez la console https://console.aws.amazon.com/imagebuilder/EC2 Image Builder à l'adresse.
2. Sélectionnez Lifecycle policies dans le volet de navigation. Cela affiche une liste des politiques de cycle de vie des images de votre compte.

Actions disponibles

Vous pouvez également effectuer les actions suivantes pour votre politique de cycle de vie à partir de la page de liste des politiques de cycle de vie.

Pour créer une nouvelle politique de cycle de vie des images, choisissez Créer une politique de cycle de vie. Pour plus d'informations sur la création d'une politique, consultezCréez des politiques de cycle de vie.

Pour toutes les actions suivantes, vous devez d'abord sélectionner la politique. Pour sélectionner une politique, vous pouvez cocher la case à côté du nom de la politique.

- Pour activer ou désactiver la politique, sélectionnez Désactiver la politique ou Activer la politique dans le menu Actions.
- Pour modifier la politique, sélectionnez Modifier la politique dans le menu Actions.
- Pour supprimer une politique, sélectionnez Supprimer la politique dans le menu Actions.
- Pour créer une nouvelle politique qui utilise la stratégie que vous avez sélectionnée pour les paramètres de base, sélectionnez Cloner la politique dans le menu Actions.

AWS CLI

L'exemple de commande suivant montre comment utiliser le pour AWS CLI répertorier les politiques de cycle de vie des images pour une image spécifique Région AWS. Pour plus d'informations sur les paramètres et les options que vous pouvez utiliser avec cette commande, consultez la list-lifecycle-policiescommande dans le manuel de référence des AWS CLI commandes.

Exemple :

```
aws imagebuilder list-lifecycle-policies \
--region us-west-1
```

Sortie :

```
{
    "lifecyclePolicySummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy1",
            "name": "sample-lifecycle-policy1",
            "status": "DISABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-11-07T14:57:01.603000-08:00",
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy2",
            "name": "sample-lifecycle-policy2",
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-09-06T10:43:21.436000-07:00",
            "dateLastRun": "2023-11-13T04:43:46.106000-08:00",
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy3",
            "name": "sample-lifecycle-policy3",
```

```
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-10-19T15:16:40.046000-07:00",
            "dateUpdated": "2023-10-21T20:07:15.958000-07:00",
            "dateLastRun": "2023-11-12T09:27:45.830000-08:00"
}]}
```

> **Note**
>
> Pour utiliser votre valeur par défaut Région AWS, exécutez cette commande sans le `--region` paramètre.

Afficher les détails des politiques de cycle de vie

Utilisez la console EC2 Image Builder, l'API ou AWS CLI l'API pour consulter la configuration des politiques de cycle de vie, l'historique d'exécution et les définitions des règles.

Table des matières

- Afficher les détails de la politique de cycle de vie dans la console Image Builder

Afficher les détails de la politique de cycle de vie dans la console Image Builder

La page détaillée de la politique de cycle de vie affiche une section récapitulative et des onglets contenant des informations supplémentaires.

Sections et onglets détaillés de la console

- Section récapitulative
- Onglet Règles
- Onglet Étendue
- RunLog onglet

Section récapitulative

La section récapitulative présente les informations suivantes :

État de la politique

Actif (la politique s'exécute comme prévu) ou Inactif (la politique est suspendue).

Type

Type d'image de sortie : AMI ou image de conteneur.

Date de création

Horodatage lorsque vous avez créé la politique.

Date de modification

Horodatage de la dernière mise à jour de la politique.

Date de la dernière diffusion

Horodatage de la dernière exécution de la politique.

Rôle IAM

Rôle IAM assumé par Image Builder pour effectuer les actions du cycle de vie.

ARN

Amazon Resource Name (ARN) de la politique de cycle de vie.

Description

Description de la politique, si elle est fournie.

Onglet Règles

L'onglet Règles affiche les règles de cycle de vie configurées pour cette politique :

- Nom : nom fixe correspondant au type d'action politique :
  - `Deprecation rule`
  - `Disable rule`
  - `Deletion rule`
- Règle — Brève description de l'action configurée.
- Conditions des règles : configuration de la gestion des ressources, exceptions aux règles et paramètres de rétention.

Pour plus d'informations sur la configuration des règles, consultezComment fonctionnent les règles du cycle de vie.

Onglet Étendue

L'onglet Étendue affiche les critères de sélection des ressources pour cette politique :

- Filtre : ***`type of filter`*** — Le type de filtre qui définit le champ d'application :
  - `recipes`— Recettes utilisées pour créer les images auxquelles cette politique s'applique.
  - `tags`— Tags utilisés par Image Builder pour sélectionner les images auxquelles cette politique s'applique.
- Barre de recherche : filtre la liste par nom.
- Nom : nom ou balise configuré comme critère de filtre.
- Version — Version de recette (filtre de recette uniquement).

RunLog onglet

Image Builder enregistre les détails de l'exécution pour chaque exécution de politique. Chaque ligne du tableau représente une exécution :

- ID d'exécution — Identifiant unique pour l'instance d'exécution.
- État d'exécution — État d'exécution actuel.
- Ressource affectée : indique si l'exécution a identifié des ressources d'image pour les actions du cycle de vie.
- Date de début — Horodatage du début de l'exécution.
- Date de fin — Horodatage de la fin de l'exécution. Ce champ est vide si l'exécution est en cours ou en attente.

Pour les descriptions de l'état d'exécution, voirStatuts d'exécution.

Affichage des détails par ressource

Exécutez la AWS CLI commande list-lifecycle-execution-resources avec l'ID d'exécution pour afficher les résultats des ressources individuelles. Chaque entrée indique l'action entreprise, son résultat (réussite, échec ou omission) et la raison.

Créez des politiques de cycle de vie

Lorsque vous créez une nouvelle politique de cycle de vie d'EC2 Image Builder, la configuration dépend du type d'image auquel cette politique est destinée. L'action de l'API pour créer une politique de cycle de vie pour les ressources d'images AMI et les ressources d'images de conteneurs est la même (CreateLifecyclePolicy). Cependant, la configuration des ressources d'image et des ressources associées est différente. Cette section explique comment créer des politiques de gestion du cycle de vie pour les deux.

> **Note**
>
> Avant de créer une politique de cycle de vie, assurez-vous de respecter toutes les règlesConditions préalables.

Créez des politiques de gestion du cycle de vie pour les ressources d'images AMI d'Image Builder

Vous pouvez utiliser l'une des méthodes suivantes pour créer une politique de cycle de vie des images AMI à l'aide du Console de gestion AWS ou AWS CLI. Vous pouvez également utiliser l'action d'CreateLifecyclePolicyAPI. Pour la demande de SDK associée, vous pouvez vous référer au lien Voir aussi pour cette commande dans la référence de l'API EC2 Image Builder.

Console de gestion AWS

Pour créer une politique de cycle de vie pour les ressources d'images AMI dans le Console de gestion AWS, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez les politiques de cycle de vie dans le volet de navigation.
3. Choisissez Créer une politique de cycle de vie.
4. Configurez les paramètres de stratégie décrits dans les procédures suivantes.
5. Pour créer la politique de cycle de vie après avoir configuré les paramètres, choisissez Créer une politique.

Configurez les paramètres généraux de votre politique.

1. Sélectionnez l'option AMI dans Type de politique.
2. Entrez le nom de la politique.
3. Entrez éventuellement une description pour votre politique de cycle de vie.
4. Par défaut, l'option Activer est activée. Le paramètre par défaut active la politique de cycle de vie et l'ajoute immédiatement au calendrier. Pour créer une politique initialement désactivée, vous pouvez désactiver Activer.
5. Pour le rôle IAM, sélectionnez l'une des options suivantes :

   - Créer un rôle d'exécution du cycle de vie à l'aide des paramètres par défaut du service : génère un rôle préconfiguré auquel est associée la politique `EC2ImageBuilderLifecycleExecutionPolicy` gérée. Il s'agit de l'option recommandée si vous n'avez pas d'exigences d'autorisation personnalisées spécifiques.
   - Choisissez un rôle existant : sélectionnez un rôle IAM existant dans la liste déroulante. La liste filtre les Service-Linked rôles (SLR) qui ne sont pas compatibles avec l'exécution de la politique de cycle de vie.
   - Créez un nouveau rôle d'exécution du cycle de vie : ouvre la console IAM avec des paramètres de politique de confiance et d'exécution du cycle de vie préremplis pour la création de rôles en un clic. Pour obtenir des instructions détaillées, consultez Création d'un rôle IAM pour la gestion du cycle de vie d'Image Builder.

   > **Note**
   >
   > Service-Linked Les rôles ne sont pas compatibles avec l'exécution de la politique de cycle de vie et sont automatiquement filtrés à partir de la sélection des rôles. Si vous avez déjà créé une politique de cycle de vie avec un Service-Linked rôle, vous pouvez la mettre à jour pour utiliser un rôle d'exécution compatible.

Configurez la portée de la règle pour votre politique.

Cette section configure la sélection des ressources pour votre politique de cycle de vie, en fonction du type de filtre que vous utilisez.

1. Type de filtre : recettes — Pour appliquer des règles de cycle de vie aux ressources d'image en fonction de la recette qui les a créées, sélectionnez jusqu'à 50 versions de recettes pour la

politique. Vous pouvez utiliser le navigateur à panneau partagé pour rechercher des recettes et sélectionner des versions spécifiques.

Vous pouvez également utiliser des caractères génériques pour la version sémantique afin de cibler plusieurs versions d'une recette avec une seule politique. Les modèles de caractères génériques suivants sont pris en charge :

- `x.x.x`— Correspond à toutes les versions de la recette.
- `1.x.x`— Correspond à toutes les versions mineures et patchs de la version majeure 1.
- `1.0.x`— Correspond à toutes les versions de patch de la version 1.0.

Les caractères génériques sont résolus pour toutes les versions de recettes correspondantes au moment de l'exécution. Cela signifie que les nouvelles versions de recettes créées après la configuration de la politique sont automatiquement incluses dans la prochaine exécution planifiée.

> **Note**
>
> Les versions de recettes créées ou supprimées après le début de l'exécution d'une politique ne sont incluses que lors de la prochaine exécution planifiée.

2. Type de filtre : balises — Pour appliquer des règles de cycle de vie aux ressources d'image en fonction des balises de ressources, entrez une liste de 50 paires clé-valeur maximum sur lesquelles la politique doit correspondre. Image Builder découvre les ressources grâce à l'analyse des balises et applique des règles de cycle de vie aux ressources correspondantes.

   > **Note**
   >
   > Les ressources balisées ou non balisées après le début de l'exécution d'une politique ne sont incluses que lors de la prochaine exécution planifiée.

Configurez les règles d'exclusion pour votre politique (facultatif).

Les règles d'exclusion définissent des exceptions aux règles du cycle de vie. Les ressources qui répondent aux critères spécifiés par les règles d'exclusion sont exclues des actions du cycle de vie.

1. Pour les politiques relatives aux images AMI, vous pouvez configurer les règles d'exclusion d'AMI suivantes :

   - Exclure les AMI publiques : sélectionnez cette option pour exclure les AMI publiques des actions du cycle de vie.
   - Exclure les AMI par région : spécifiez Régions AWS les actions à exclure du cycle de vie.
   - Exclure les AMI partagées avec des comptes — Comptes AWS Spécifiez quelles AMI partagées doivent être exclues des actions du cycle de vie.
   - Exclure les AMI récemment lancées — Spécifiez une période pour exclure les AMI récemment utilisées pour lancer des instances.
   - Exclure les AMI par balises : spécifiez des balises pour les AMI qui doivent être exclues des actions du cycle de vie.
2. Pour les politiques d'image Image Builder basées sur des balises, vous pouvez spécifier des balises pour les ressources d'image Image Builder qui doivent être exclues des actions du cycle de vie.

Activez une ou plusieurs des règles de cycle de vie suivantes à appliquer aux ressources sélectionnées par la politique de cycle de vie. Si une ressource correspond à plusieurs règles de cycle de vie lors de l'exécution de la politique, Image Builder exécute les actions relatives aux règles dans l'ordre suivant : 1) Déprécier, 2) Désactiver, 3) Supprimer.

Règle obsolète

Définit l'état de la ressource d'image Image Builder sur`Deprecated`. Les pipelines Image Builder fonctionnent toujours pour les images obsolètes. Vous pouvez éventuellement définir la durée de dépréciation des AMI associées sans affecter votre capacité à lancer de nouvelles instances.

- Nombre d'unités : spécifiez la valeur entière correspondant à la période qui doit s'écouler après la création d'une ressource d'image avant qu'elle ne soit marquée comme`Deprecated`.
- Unité — Sélectionnez la plage de temps à utiliser. La plage peut être `Days`|`Weeks`|`Months`, ou`Years`.
- AMI obsolètes : cochez la case pour marquer les AMI Amazon EC2 associées avec une date de dépréciation. Les AMI restent disponibles et vous pouvez toujours lancer de nouvelles instances à partir de celles-ci.

Désactiver la règle

Définit l'état de la ressource d'image Image Builder sur`Disabled`. Cela empêche les pipelines Image Builder de s'exécuter pour cette image. Vous pouvez éventuellement désactiver l'AMI associée pour empêcher le lancement de nouvelles instances.

- Nombre d'unités : spécifiez la valeur entière correspondant à la période qui doit s'écouler après la création d'une ressource d'image avant qu'elle ne soit marquée comme`Disabled`.
- Unité — Sélectionnez la plage de temps à utiliser. La plage peut être `Days`,`Weeks`,`Months`, ou`Years`.
- Désactiver les AMI : cochez la case pour désactiver les AMI Amazon EC2 associées. Vous ne pouvez plus utiliser les AMI ni lancer de nouvelles instances à partir de celles-ci.

Supprimer la règle

Supprime les ressources d'images par âge ou par nombre. Vous définissez le seuil qui répond à vos besoins. Lorsqu'une ressource d'image Image Builder dépasse le seuil, elle est supprimée. Vous pouvez éventuellement désenregistrer les AMI associées ou supprimer les instantanés de ces AMI. Vous pouvez également spécifier des balises pour les ressources que vous souhaitez conserver au-delà du seuil.

Lorsque vous configurez la règle de suppression par âge, Image Builder supprime la ressource d'image au bout d'un certain temps. Supprimez, par exemple, les ressources d'images au bout de 6 mois. Lorsque vous configurez par nombre, Image Builder conserve le nombre d'images le plus récent que vous spécifiez, ou le plus proche possible de ce nombre, et supprime les versions précédentes.

- Par âge
  - Nombre d'unités : spécifiez la valeur entière correspondant à la période qui doit s'écouler entre la création d'une ressource d'image et sa suppression.
  - Unité — Sélectionnez la plage de temps à utiliser. La plage peut être `Days`,`Weeks`,`Months`, ou`Years`.
  - Conserver au moins une image par recette : cochez la case pour conserver la dernière ressource d'images disponible pour chaque version de recette concernée par cette règle.

  Par décompte

  - Nombre d'images — Spécifiez la valeur entière du nombre de ressources d'images récentes à conserver pour chaque version de recette.

- Désenregistrer les AMI : cochez la case pour désenregistrer les AMI Amazon EC2 associées. Vous ne pouvez plus utiliser les AMI ni lancer de nouvelles instances à partir de celles-ci.
- Conservez les images, les AMI et les instantanés avec les balises associées : cochez la case pour entrer une liste de balises pour les ressources d'images que vous souhaitez conserver. Les balises s'appliquent aux ressources d'images et aux AMI Amazon EC2. Vous pouvez saisir jusqu'à 50 paires clé-valeur.

Balises (facultatif)

Ajoutez des balises à votre politique de cycle de vie.

AWS CLI

Pour créer une nouvelle politique de cycle de vie d'Image Builder, vous pouvez utiliser la create-lifecycle-policy commande du. AWS CLI

Créez des politiques de gestion du cycle de vie pour les ressources d'images de conteneurs Image Builder

Vous pouvez utiliser l'une des méthodes suivantes pour créer une politique de cycle de vie des images de conteneur à l'aide du Console de gestion AWS ou AWS CLI. Vous pouvez également utiliser l'action d'CreateLifecyclePolicyAPI. Pour la demande de SDK associée, vous pouvez vous référer au lien Voir aussi pour cette commande dans la référence de l'API EC2 Image Builder.

Console de gestion AWS

Pour créer une politique de cycle de vie pour les ressources d'images de conteneurs dans le Console de gestion AWS, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez les politiques de cycle de vie dans le volet de navigation.
3. Choisissez Créer une politique de cycle de vie.
4. Configurez les paramètres de stratégie décrits dans les procédures suivantes.
5. Pour créer la politique de cycle de vie après avoir configuré les paramètres, choisissez Créer une politique.

Configuration de la politique : paramètres généraux

Configurez les paramètres généraux de votre politique.

1. Sélectionnez l'option Image du conteneur dans Type de politique.
2. Entrez le nom de la politique.
3. Entrez éventuellement une description pour votre politique de cycle de vie.
4. Par défaut, Activer est activé. Le paramètre par défaut active la politique de cycle de vie et l'ajoute immédiatement au calendrier. Pour créer une politique initialement désactivée, vous pouvez désactiver Activer.
5. Pour le rôle IAM, sélectionnez l'une des options suivantes :

   - Créer un rôle d'exécution du cycle de vie à l'aide des paramètres par défaut du service : génère un rôle préconfiguré auquel est associée la politique `EC2ImageBuilderLifecycleExecutionPolicy` gérée. Il s'agit de l'option recommandée si vous n'avez pas d'exigences d'autorisation personnalisées spécifiques.
   - Choisissez un rôle existant : sélectionnez un rôle IAM existant dans la liste déroulante. La liste filtre les Service-Linked rôles (SLR) qui ne sont pas compatibles avec l'exécution de la politique de cycle de vie.
   - Créez un nouveau rôle d'exécution du cycle de vie : ouvre la console IAM avec des paramètres de politique de confiance et d'exécution du cycle de vie préremplis pour la création de rôles en un clic. Pour obtenir des instructions détaillées, consultez Création d'un rôle IAM pour la gestion du cycle de vie d'Image Builder.

   > **Note**
   >
   > Service-Linked Les rôles ne sont pas compatibles avec l'exécution de la politique de cycle de vie et sont automatiquement filtrés à partir de la sélection des rôles. Si vous avez déjà créé une politique de cycle de vie avec un Service-Linked rôle, vous pouvez la mettre à jour pour utiliser un rôle d'exécution compatible.

Configurez la portée de la règle pour votre politique.

Cette section configure la sélection des ressources pour votre politique de cycle de vie, en fonction du type de filtre que vous utilisez.

1. Type de filtre : recettes — Pour appliquer des règles de cycle de vie aux ressources d'image en fonction de la recette qui les a créées, sélectionnez jusqu'à 50 versions de recettes pour la politique. Vous pouvez utiliser le navigateur à panneau partagé pour rechercher des recettes et sélectionner des versions spécifiques.

   Vous pouvez également utiliser des caractères génériques pour la version sémantique afin de cibler plusieurs versions d'une recette avec une seule politique. Les modèles de caractères génériques suivants sont pris en charge :

   - `x.x.x`— Correspond à toutes les versions de la recette.
   - `1.x.x`— Correspond à toutes les versions mineures et patchs de la version majeure 1.
   - `1.0.x`— Correspond à toutes les versions de patch de la version 1.0.

   Les modèles de caractères génériques sont résolus pour toutes les versions de recettes correspondantes au moment de l'exécution. Cela signifie que les nouvelles versions de recettes créées après la configuration de la politique sont automatiquement incluses dans la prochaine exécution planifiée.

   > **Note**
   >
   > Les versions de recettes créées ou supprimées après le début de l'exécution d'une politique ne sont incluses que lors de la prochaine exécution planifiée.

2. Type de filtre : balises — Pour appliquer des règles de cycle de vie aux ressources d'image en fonction des balises de ressources, entrez une liste de 50 paires clé-valeur maximum sur lesquelles la politique doit correspondre. Image Builder découvre les ressources grâce à l'analyse des balises et applique des règles de cycle de vie aux ressources correspondantes.

   > **Note**
   >
   > Les ressources balisées ou non balisées après le début de l'exécution d'une politique ne sont incluses que lors de la prochaine exécution planifiée.

Configurez les règles d'exclusion pour votre politique (facultatif).

Les règles d'exclusion définissent des exceptions aux règles du cycle de vie. Les ressources qui répondent aux critères spécifiés par les règles d'exclusion sont exclues des actions du cycle de vie.

1. Pour les politiques relatives aux images AMI, vous pouvez configurer les règles d'exclusion d'AMI suivantes :

   - Exclure les AMI publiques : sélectionnez cette option pour exclure les AMI publiques des actions du cycle de vie.
   - Exclure les AMI par région : spécifiez Régions AWS les actions à exclure du cycle de vie.
   - Exclure les AMI partagées avec des comptes — Comptes AWS Spécifiez quelles AMI partagées doivent être exclues des actions du cycle de vie.
   - Exclure les AMI récemment lancées — Spécifiez une période pour exclure les AMI récemment utilisées pour lancer des instances.
   - Exclure les AMI par balises : spécifiez des balises pour les AMI qui doivent être exclues des actions du cycle de vie.
2. Pour les politiques d'image Image Builder basées sur des balises, vous pouvez spécifier des balises pour les ressources d'image Image Builder qui doivent être exclues des actions du cycle de vie.

Supprimer la règle

Pour les images de conteneur, cette règle supprime la ressource d'image de conteneur Image Builder. Vous pouvez éventuellement supprimer les Docker images qui ont été distribuées dans les référentiels ECR pour empêcher qu'elles ne soient utilisées pour exécuter de nouveaux conteneurs.

Lorsque vous configurez la règle de suppression par âge, Image Builder supprime la ressource d'image au bout d'un certain temps. Supprimez, par exemple, les ressources d'images au bout de 6 mois. Lorsque vous configurez par nombre, Image Builder conserve le nombre d'images le plus récent que vous spécifiez, ou le plus proche possible de ce nombre, et supprime les versions précédentes.

- Par âge
  - Nombre d'unités : spécifiez la valeur entière correspondant à la période qui doit s'écouler entre la création d'une ressource d'image et sa suppression.

- Unité — Sélectionnez la plage de temps à utiliser. La plage peut être `Days`, `Weeks`, `Months`, ou `Years`.
- Conserver au moins une image : cochez la case pour ne conserver que la dernière ressource d'image disponible pour chaque version de recette concernée par cette règle.

Par décompte

- Nombre d'images — Spécifiez la valeur entière du nombre de ressources d'images récentes à conserver pour chaque version de recette.

- Supprimer les images de conteneurs ECR : cochez la case pour supprimer les images de conteneurs associées stockées dans un référentiel ECR. Vous ne pouvez plus utiliser l'image du conteneur comme base pour créer de nouvelles images ou pour exécuter de nouveaux conteneurs.
- Conserver les images avec les balises associées : cochez la case pour entrer une liste de balises pour les ressources d'images que vous souhaitez conserver.

Balises (facultatif)

Ajoutez des balises à votre politique de cycle de vie.

AWS CLI

Pour créer une nouvelle politique de cycle de vie d'Image Builder, vous pouvez utiliser la create-lifecycle-policy commande du. AWS CLI

Comment fonctionne l'exécution des politiques de cycle de vie

Image Builder exécute des politiques de cycle de vie activées selon un calendrier récurrent. Chaque run est une exécution. Au cours d'une exécution, Image Builder évalue vos ressources d'image par rapport aux règles de politique et exécute les actions de cycle de vie spécifiées.

Table des matières

- Calendrier d'exécution
- Statuts d'exécution
- Comment les ressources sont évaluées
- Comment fonctionne la rétention pour les règles de suppression

- Gestion manuelle de l'état des ressources
- Annulation d'une exécution
- Comment les politiques de cycle de vie gèrent les ressources modifiées de l'extérieur
- Surveillance des résultats d'exécution

Calendrier d'exécution

Image Builder exécute les politiques de cycle de vie activées une fois par jour.

- Image Builder gère le temps d'exécution exact. Vous ne pouvez pas le configurer.
- Une seule exécution est exécutée par politique à la fois. Une exécution en cours ou en attente se poursuit au lieu d'en démarrer une nouvelle.
- Les politiques désactivées ne s'exécutent pas, et la désactivation d'une politique empêche toute exécution en attente de traiter de nouvelles ressources.

Statuts d'exécution

Surveillez la progression de l'exécution du cycle de vie à l'aide de RunLog l'onglet de la console ou de la commande AWS CLI list-lifecycle-executions. Chaque exécution indique l'un des statuts suivants :

`IN_PROGRESS`

Image Builder évalue activement les ressources et exécute des actions relatives au cycle de vie.

`PENDING`

Le traitement est suspendu. Il reprendra lors de la prochaine exécution planifiée.

`SUCCESS`

Image Builder a traité toutes les ressources applicables avec succès.

`FAILED`

Une erreur a été détectée lors d'une ou de plusieurs actions sur les ressources. Utilisez la commande list-lifecycle-execution-resources pour identifier les ressources qui ont échoué et pourquoi.

`CANCELLING`

Image Builder a reçu une demande d'annulation et termine l'étape de traitement en cours.

`CANCELLED`

L'exécution a été annulée. Image Builder n'annule pas les ressources traitées avant l'annulation.

Comment les ressources sont évaluées

Image Builder évalue les ressources d'images de la plus récente à la plus ancienne (par date de création) en suivant les étapes suivantes :

1. Sélection des ressources : Image Builder identifie toutes les ressources d'image correspondant au champ d'application de la politique, défini soit par un filtre de recettes (y compris les modèles de version), soit par un filtre de balises.
2. Contrôle de compilation actif : Image Builder ignore les images dont la génération est en cours (état de création, de test ou de distribution) et les évalue lors de la prochaine exécution.
3. Évaluation des règles — Image Builder évalue les règles par ordre de priorité : dépréciation, désactivation, puis suppression. Une seule action s'applique à chaque image par exécution. Si une image correspond à une règle obsolète, Image Builder ignore les règles de désactivation et de suppression pour cette image au cours de la même exécution.

   L'évaluation varie selon le type de règle :

   Pour les règles d'obsolescence et de désactivation

   a. Vérification du filtre AGE (si configuré) : Image Builder vérifie si l'image est plus ancienne que le seuil spécifié.
   b. Image-level tag exclusion check (`exclusionRules.tagMap`) : Image Builder ignore les images dotées d'une balise d'exclusion.

   Pour les règles de suppression

   a. Vérification du filtre AGE ou COUNT : pour AGE, Image Builder vérifie si l'image est plus ancienne que le seuil. Pour COUNT, Image Builder vérifie si le nombre de rétention a été atteint.
   b. Vérification échouée ou annulée : les images dont l'état a échoué ou annulé peuvent être supprimées et ne consomment aucun espace de rétention.
   c. Contrôle de rétention : si vous avez configuré une `retainAtLeast` valeur (filtre AGE) ou une valeur de comptage (filtre COUNT), les images les plus récentes remplissent

les emplacements de rétention en premier. Une image remplissant un emplacement de rétention reste sans autre évaluation.

d. Image-level tag exclusion check (`exclusionRules.tagMap`) : pour les images qui passent le contrôle de rétention (images qui seraient sinon supprimées), Image Builder vérifie la présence d'une balise d'exclusion. Le cas échéant, Image Builder ignore l'image.

> **Important**
>
> Image Builder évalue la rétention avant l'exclusion des balises. Une image située dans la fenêtre de rétention reste protégée indépendamment des balises d'exclusion. L'exclusion des balises s'applique uniquement aux images au-delà de la limite de rétention.

Une image dont les balises sont exclues ne consomme pas de créneau de rétention et n'a aucune incidence sur le taux de rétention des autres images.

4. AMI-level contrôles d'exclusion : après avoir déterminé les images éligibles, Image Builder évalue leurs AMI de sortie par rapport aux règles AMI-level d'exclusion (`exclusionRules.amis`). Ces règles incluent l'exclusion des régions, l'exclusion des AMI publiques, l'exclusion des balises AMI, l'exclusion des comptes partagés et l'exclusion lancée pour la dernière fois. Image Builder signale une AMI de sortie qui répond à tous les critères d'exclusion, `SKIPPED` avec une raison indiquant la condition correspondante.

> **Note**
>
> Les exécutions suivantes appliquent des règles de désactivation ou de suppression aux ressources précédemment déconseillées si elles répondent aux critères.

Comment fonctionne la rétention pour les règles de suppression

Les filtres de rétention (AGE et COUNT) évaluent en fonction des ressources d'image Image Builder et ne génèrent pas directement des AMI ou des conteneurs. Image Builder conserve les images les plus récentes par version de recette et marque les anciennes pour suppression.

Pour une logique de calcul détaillée de la rétention, y compris le comportement des filtres basé sur l'âge et le nombre, voir. Comment est calculée la rétention

Gestion manuelle de l'état des ressources

Vous pouvez déclencher des modifications immédiates de l'état du cycle de vie pour une version de création d'image spécifique à l'aide de la commande start-resource-state-update, sans attendre la prochaine exécution planifiée.

Transitions prises en charge :

- Obsolète — Marquer une image comme obsolète.
- Désactiver — Empêche le lancement de nouvelles instances par une image.
- Supprimer — Supprime une image et ses ressources associées (vous devez spécifier au moins un type de ressource associé).
- Disponible — Rétablit l'état disponible d'une image obsolète ou désactivée.

Chaque changement d'état manuel crée une exécution distincte du cycle de vie. Suivez-le à l'aide de la commande list-lifecycle-executions.

Annulation d'une exécution

Annulez une exécution en cours ou en attente à l'aide de la commande cancel-lifecycle-execution ou de la console Image Builder.

Lorsque vous annulez une exécution :

- L'exécution passe à la `CANCELLING` fin de l'action sur les ressources en cours.
- Une fois l'action en cours terminée, l'exécution passe à`CANCELLED`.
- Image Builder ne restaure pas les ressources déjà traitées. Par exemple, une image déconseillée avant son annulation reste obsolète.
- Image Builder ignore les ressources non traitées.

Comment les politiques de cycle de vie gèrent les ressources modifiées de l'extérieur

Vous ou d'autres AWS services pouvez modifier les ressources de votre compte en dehors des règles de cycle de vie d'Image Builder. Exemples de modifications externes :

- Annulation de l'enregistrement des AMI via la console Amazon EC2
- Obsolète de l'AMI à l'aide de l'API Amazon EC2
- Suppression d'une image de conteneur depuis un référentiel Amazon ECR
- Suppression de ressources d'image Image Builder à l'aide de l'API Image Builder

Si une ressource est déjà supprimée ou présente déjà l'état souhaité, Image Builder signale l'action sous la formeSUCCESS. La politique n'échoue pas lorsque les ressources ont été nettoyées par d'autres moyens.

> **Note**
>
> Un SKIPPED statut est différent. Image Builder signale SKIPPED uniquement lorsqu'une ressource existe mais qu'une règle d'exclusion empêche l'action.

Surveillance des résultats d'exécution

Surveillez les résultats d'exécution du cycle de vie à deux niveaux :

Niveau d'exécution

Affichez toutes les exécutions d'une politique, y compris le statut, l'heure de début et l'heure de fin.

- Console : ouvrez la page des détails de la politique de cycle de vie et choisissez l'RunLogonglet. Pour de plus amples informations, veuillez consulter RunLog onglet.
- AWS CLI— Exécutez list-lifecycle-executions avec l'ARN de la politique de cycle de vie.

Niveau de ressource

Affichez les résultats par ressource dans le cadre d'une exécution.

- Console — Dans l'RunLogonglet, choisissez un ID d'exécution pour afficher les ressources traitées.
- AWS CLI— Exécutez list-lifecycle-execution-resources avec l'ID d'exécution.

Chaque entrée de ressource inclut :

- ARN des ressources

- Action entreprise (ou tentative)
- État de la ressource (`SUCCESS`FAILED, ou`SKIPPED`)
- Raison du résultat (particulièrement utile pour les ressources ignorées ou défaillantes)

Comment fonctionnent les règles de gestion du cycle de vie pour les ressources d'image Image Builder

Les règles du cycle de vie permettent de maintenir vos images à jour et de réduire les coûts d'infrastructure, tels que le stockage des instantanés pour les AMI de sortie et le stockage du référentiel Amazon ECR pour les conteneurs.

Les politiques de cycle de vie prennent en charge les types de règles suivants :

Règle obsolète

Définit le statut de la ressource d'image Image Builder sur`Deprecated`. Les pipelines Image Builder fonctionnent toujours pour les images obsolètes. Vous pouvez éventuellement définir le délai d'obsolescence des AMI associées sans affecter votre capacité à lancer de nouvelles instances.

Les AMI obsolètes n'apparaissent pas dans les recherches générales. Par exemple, la describe-images commande Amazon EC2 exclut les AMI obsolètes des résultats. Pour trouver une AMI obsolète, spécifiez directement son ID d'AMI.

> **Note**
>
> Cette règle ne s'applique pas aux images basées sur des conteneurs.

Désactiver la règle

Définit le statut de la ressource d'image Image Builder sur`Disabled`. Cela empêche les pipelines Image Builder de s'exécuter pour cette image. Vous pouvez éventuellement désactiver l'AMI associée pour empêcher le lancement de nouvelles instances.

Une AMI désactivée devient privée et ne lance plus de nouvelles instances. Les comptes, organisations ou unités organisationnelles qui bénéficiaient auparavant d'un accès partagé perdent cet accès.

> **Note**
>
> Cette règle ne s'applique pas aux images basées sur des conteneurs.

Supprimer la règle

Supprime les ressources d'image par âge ou par nombre. Vous définissez le seuil qui répond à vos besoins. Lorsqu'une ressource d'image Image Builder dépasse le seuil, elle est supprimée. Vous pouvez éventuellement désenregistrer les AMI associées ou supprimer les instantanés de ces AMI. Vous pouvez également spécifier des balises pour les ressources que vous souhaitez conserver au-delà du seuil.

Pour les images de conteneur, cette règle supprime la ressource d'image de conteneur Image Builder. Vous pouvez également supprimer des images de conteneurs des référentiels ECR pour empêcher le lancement de nouveaux conteneurs.

Priorité des règles et ordre d'évaluation

Lorsqu'une politique comporte plusieurs types de règles, Image Builder évalue chaque ressource dans cet ordre :

1. Règle obsolète
2. Désactiver la règle
3. Supprimer la règle

> **Important**
>
> Une seule action s'applique à chaque ressource par exécution. Si une ressource correspond à une règle obsolète, Image Builder la déconseille et ignore les règles de désactivation et de suppression pour cette ressource au cours de la même exécution. Lors des exécutions suivantes, Image Builder évalue la ressource par rapport à d'autres règles si elle répond à ses critères.

Cet ordre prioritaire soutient une stratégie de cycle de vie progressive. Par exemple, vous pouvez configurer une politique pour rendre les images obsolètes après 90 jours, les désactiver après

120 jours et les supprimer après 180 jours. Chaque exécution applique uniquement l'action correspondante la plus prioritaire.

États d'image qui ignorent l'évaluation des règles

Image Builder ignore les images dans certains états pour éviter les actions dupliquées :

- Règle de dépréciation : Image Builder ignore les images déjà obsolètes, désactivées, défaillantes ou annulées.
- Règle de désactivation : Image Builder ignore les images déjà désactivées, défaillantes ou annulées.
- Règle de suppression : Image Builder évalue les images, quel que soit leur état, en vue de leur suppression, sous réserve des règles de conservation et des exclusions.

Table des matières

- Règles d'exclusion du cycle de vie des AMI
- Les balises en tant que sélecteurs de ressources ou règles d'exclusion
- Comment est calculée la rétention
- Comment les actions du cycle de vie affectent les ressources associées
- Transitions d'état des ressources autorisées
- Afficher les détails des règles pour une politique de cycle de vie

Règles d'exclusion du cycle de vie des AMI

Les règles d'exclusion protègent des AMI spécifiques contre les actions liées au cycle de vie. Configurez les règles d'exclusion dans Console de gestion AWS, l'API ou le AWS CLI.

Les politiques de cycle de vie prennent en charge les types de règles d'exclusion suivants :

- Tag-based exclusion — Exclut les ressources d'image Image Builder dotées de balises spécifiques (jusqu'à 50 balises). Pour plus d'informations sur le fonctionnement des balises aux différents niveaux de ressources, consultezLes balises en tant que sélecteurs de ressources ou règles d'exclusion.
- Exclusion des AMI publiques : exclut les AMI partagées publiquement.
- Dernière exclusion lancée : exclut les AMI qui ont lancé une instance au cours d'une période spécifiée.

> **Note**
>
> Amazon EC2 indique l'heure du dernier lancement avec un délai de 24 heures. Les lancements effectués au cours des dernières 24 heures peuvent ne pas être pris en compte lors de l'exécution du contrôle d'exclusion.

- Exclusion régionale — Exclut les AMI en particulier Régions AWS.
- Exclusion des comptes partagés : exclut les AMI partagées avec des utilisateurs spécifiques Comptes AWS.

> **Note**
>
> Image Builder ignore les images dont la génération est en cours (état en attente, création, création, importation, test, distribution ou intégration). Image Builder évalue ces images lors de la prochaine exécution planifiée.

Ordre d'évaluation des règles d'exclusion AMI

EC2 Image Builder AMI-level évalue les règles d'exclusion `exclusionRules.amis` () sur l'AMI de sortie après avoir déterminé l'éligibilité à une action du cycle de vie. Ces règles s'appliquent à la ressource AMI Amazon EC2, et non à la ressource image Image Builder. Le comptage de rétention est effectué séparément. Une AMI exclue ne consomme aucun emplacement de rétention. Image Builder vérifie les conditions dans l'ordre et ignore l'AMI lors de la première correspondance :

1. Exclusion de région — Vérifie si l'AMI se trouve dans une région exclue.
2. Exclusion de l'AMI publique : vérifie si l'AMI est publique et si `isPublic` l'exclusion est activée.
3. Exclusion de balises : vérifie si l'AMI possède des balises correspondant à la carte des balises d'exclusion.
4. Exclusion des comptes partagés : vérifie si l'AMI est partagée avec les comptes répertoriés dans l'exclusion.
5. Dernière exclusion lancée — Vérifie si l'AMI a été lancée pendant la période exclue.

Dans les détails des ressources d'exécution, les AMI ignorées affichent un `SKIPPED` statut avec une raison indiquant à quelle condition d'exclusion correspond.

Les balises en tant que sélecteurs de ressources ou règles d'exclusion

Les balises ont deux objectifs distincts dans les politiques de cycle de vie. Chaque mécanisme de balise fonctionne sur un niveau de ressource spécifique :

- Image Builder : ressource Image Builder qui suit une compilation (format ARN :`arn:aws:imagebuilder:`*`region`*`:`*`account`*`:image/`*`name`*`/`*`version`*). Gérez les balises de cette ressource via Image Builder.
- AMI ou image de conteneur de sortie : image de conteneur Amazon EC2 ou Amazon ECR sous-jacente produite par Image Builder. Gérez les balises sur ces ressources via Amazon EC2 ou Amazon ECR respectivement.

Tags en tant que sélecteurs de ressources (champ d'application de la politique)

Tag-based la sélection des ressources évalue les balises de la ressource image Image Builder uniquement, et non des AMI ou des conteneurs de sortie. Image Builder utilise ces balises pour rechercher les ressources auxquelles la politique s'applique.

Configurez la sélection de ressources basée sur des balises `resourceSelection.tagMap` sur le terrain. Chaque entrée est une paire clé-valeur. La politique sélectionne les images Image Builder correspondant à n'importe quelle paire clé-valeur de balise spécifiée (logique OR). Image Builder n'évalue pas les images sans balises correspondantes.

Balises en tant que règles d'exclusion (exceptions au niveau des règles)

Les balises d'exclusion fonctionnent dans le sens inverse. Une fois qu'Image Builder a sélectionné les ressources en fonction du champ d'application de la politique, il vérifie chaque ressource par rapport aux règles d'exclusion avant de prendre des mesures. Configurez les balises d'exclusion à deux niveaux, chacun opérant sur un type de ressource différent :

`exclusionRules.tagMap`

Évalue les balises de la ressource d'image Image Builder uniquement. Ne vérifie pas les balises sur les AMI ou les conteneurs sous-jacents. Si une balise d'image Image Builder correspond à une entrée (correspondance exacte entre la clé et la valeur), Image Builder ignore l'image et toutes les ressources de sortie associées.

`exclusionRules.amis.tagMap`

Évalue les balises sur l'AMI de sortie (la EC2-level ressource Amazon) lors du traitement des ressources associé. Utilisez-le pour exclure des AMI de sortie spécifiques en fonction de leurs

balises Amazon EC2 (par exemple, des balises appliquées via les paramètres de distribution ou directement via Amazon EC2).

> **Important**
>
> Sélectionnez les ressources et `exclusionRules.tagMap` n'évaluez pas les balises sur les AMI de sortie (balises Amazon EC2) ou les conteneurs (balises Amazon ECR). Pour exclure les AMI de sortie basées sur les balises Amazon EC2, utilisez. `exclusionRules.amis.tagMap`

Example Le sélecteur de balises et l'exclusion fonctionnent ensemble

Une politique avec les paramètres suivants :

- Balise de sélection des ressources : `Environment: Production` (évaluée sur les ressources d'image Image Builder)
- Image-level balise d'exclusion (`exclusionRules.tagMap`) : `Retain: True` (évaluée sur les ressources d'image Image Builder)
- AMI-level balise d'exclusion (`exclusionRules.amis.tagMap`) : `SharedWith: Partner` (évaluée sur les AMI de sortie)

Image Builder exécute les étapes suivantes :

1. Sélection des ressources : Image Builder trouve toutes les ressources d'image Image Builder baliséesEnvironment: Production.
2. Image-level exclusion : Image Builder vérifie la présence du tag dans chaque image Image Builder sélectionnéeRetain: True. Image Builder ignore les images comportant cette balise, y compris leurs ressources de sortie.
3. AMI-level exclusion — Image Builder vérifie la présence de la balise dans les AMI de sortie (ressources Amazon EC2) des images restantes. SharedWith: Partner Les AMI comportant cette balise sont ignorées.

Comment est calculée la rétention

Lorsque vous configurez les paramètres de rétention pour les règles de suppression, les filtres AGE et COUNT évaluent en fonction de vos ressources d'image Image Builder, et non directement en fonction des AMI ou des conteneurs de sortie.

> **Important**
>
> Le filtre AGE utilise la date de création de la ressource image Image Builder (définie une fois la génération terminée), et non l'AMI de sortie ou la date de création du conteneur. Le filtre COUNT compte les ressources d'image Image Builder par version de recette, et non les AMI ou les conteneurs en sortie.

Le calcul de la rétention dépend de la manière dont vous spécifiez les versions des recettes dans le champ d'application de votre politique :

Version spécifique (par exemple,`1.0.0`)

Le taux de rétention s'applique aux images Image Builder créées avec cette version de recette exacte. Image Builder conserve d'abord les images les plus récentes (par date de création).

Modèle de version Wildcard (par exemple,`1.x.x`)

Le taux de rétention s'applique indépendamment à chaque version de recette correspondante. Par exemple, `1.x.x` avec un taux de rétention de `5`1.0.0, conserve les 5 images les plus récentes pour`1.1.0`, les 5 plus récentes pour, etc.

Tag-based sélection des ressources

Avec la sélection des ressources basée sur des balises (voirLes balises en tant que sélecteurs de ressources ou règles d'exclusion), la rétention est également calculée par version de recette. Image Builder trie les images les plus récentes en premier dans chaque version et applique les valeurs de rétention indépendamment par groupe de versions.

Age-based par rapport aux filtres de suppression basés sur le nombre

Les règles de suppression utilisent l'un des deux types de filtres suivants pour déterminer l'éligibilité à la suppression :

`AGE`filtre

Conserve les images Image Builder plus récentes que le seuil d'âge spécifié (en jours, semaines, mois ou années à compter de la date de création de l'image Image Builder). Vous pouvez configurer `retainAtLeast` pour conserver un nombre minimum d'images les plus récentes par version de recette, quel que soit l'âge.

`COUNT`filtre

Conserve les images N Image Builder les plus récentes par version de recette. Toutes les images dépassant le nombre spécifié peuvent être supprimées.

D'autres comportements de rétention s'appliquent :

- Les images Image Builder ayant échoué ou annulées ne sont pas prises en compte dans les limites de rétention et peuvent toujours être supprimées.
- Image Builder traite les images les plus récentes en premier et applique le taux de rétention avant d'évaluer l'éligibilité à la suppression.

Comment les actions du cycle de vie affectent les ressources associées

Les actions du cycle de vie d'une image Image Builder affectent les AMI de sortie ou les conteneurs associés, en fonction du type de règle et de la configuration :

Déprécier

Définit le statut de la ressource d'image Image Builder sur`Deprecated`. Vous pouvez également déconseiller les AMI de sortie associées. Les AMI obsolètes n'apparaissent pas dans les recherches générales mais restent utilisables par ID d'AMI. Image Builder ajoute une `DeprecatedBy: EC2 Image Builder` balise à chaque AMI obsolète.

S'applique uniquement aux AMI-based images, pas aux conteneurs.

Désactiver

Définit le statut de la ressource d'image Image Builder sur`Disabled`. Vous pouvez également désactiver les AMI de sortie associées. Les AMI désactivées deviennent privées et ne lancent plus de nouvelles instances.

S'applique uniquement aux AMI-based images, pas aux conteneurs.

Suppression

Supprime la ressource d'image Image Builder. Vous pouvez également supprimer :

- AMI de sortie distribuées à d'autres comptes Régions AWS et
- Instantanés associés à ces AMI
- Conteneurs distribués aux référentiels Amazon ECR

S'applique à la fois aux images AMI-based et aux images basées sur des conteneurs.

> **Note**
>
> Image Builder prend les décisions relatives au cycle de vie au niveau de l'image. Les ressources de sortie associées (AMI, instantanés, conteneurs) ne changent que si vous avez configuré la règle pour les inclure.

Transitions d'état des ressources autorisées

Les politiques de cycle de vie et les ressources d'image de transition de l'`StartResourceStateUpdate`API entre les états suivants :

Disponible → Obsolète

Marque une image disponible comme obsolète. L'image reste utilisable par ID AMI mais n'apparaît pas dans les recherches générales. Image Builder ajoute une `DeprecatedBy: EC2 Image Builder` balise pour le suivi.

Disponible ou obsolète → Désactivé

Rend l'AMI privée et empêche le lancement de nouvelles instances. Si l'AMI était précédemment obsolète, Image Builder supprime la balise obsolète car l'image est passée à un état plus restrictif.

Tout état autre que le bâtiment → Supprimé

Désenregistre l'AMI ou supprime l'image du conteneur de l'ECR. Vous pouvez inclure des instantanés associés ou des conteneurs sous-jacents à supprimer.

Obsolète ou désactivé → Disponible

L'`StartResourceStateUpdate`API renvoie une image obsolète ou désactivée à l'état disponible. Cela annule la dépréciation, réactive l'AMI et supprime la date et le tag

d'obsolescence. DeprecatedBy Cette transition fonctionne uniquement via l'API manuelle, et non via l'exécution automatique des politiques.

> **Note**
>
> Les images en cours de création (en attente, état de création, de construction, d'importation, de test, de distribution ou d'intégration) ne changent pas d'état. Une mise à jour manuelle de l'état d'une image dont la version est active échoue avec une erreur.

Afficher les détails des règles pour une politique de cycle de vie

Dans la console, la page des détails de la politique de cycle de vie contient un Onglet Règles qui affiche les détails des règles relatives à la politique.

Dans le AWS CLI, exécutez la commande get-lifecycle-policy. La réponse inclut toutes les actions configurées (règles) avec leurs paramètres.

Configurer des images personnalisées avec Image Builder

Les ressources de configuration sont les éléments constitutifs des pipelines d'images, ainsi que les images produites par ces pipelines. Ce chapitre traite de la création, de la maintenance et du partage des ressources Image Builder, notamment des composants, des recettes et des images, ainsi que de la configuration de l'infrastructure et des paramètres de distribution.

> **Note**
>
> Pour vous aider à gérer vos ressources Image Builder, vous pouvez attribuer vos propres métadonnées à chaque ressource sous forme de balises. Vous utilisez des balises pour classer vos AWS ressources de différentes manières, par exemple par objectif, propriétaire ou environnement. Cela est utile lorsque vous avez de nombreuses ressources du même type. Vous pouvez identifier plus facilement une ressource spécifique en fonction des balises que vous lui avez attribuées.
> Pour plus d'informations sur le balisage de vos ressources à l'aide des commandes Image Builder dans le AWS CLI, consultez la Balisage des ressources section de ce guide.

Table des matières

- Gérer les recettes dans Image Builder
- Gérer la configuration de l'infrastructure d'Image Builder
- Gérer les paramètres de distribution d'Image Builder

Gérer les recettes dans Image Builder

Une recette EC2 Image Builder constitue le plan directeur pour créer une image de machine personnalisée. Une recette précise trois éléments clés :

1. Image de base : point de départ de votre image (une AMI, une image de conteneur ou une machine virtuelle importée).
2. Composants : étapes de création et de test permettant de personnaliser votre image. Les composants sont facultatifs. Cette conception prend en charge les flux de travail de test et de distribution uniquement.

3. Configuration : volumes de stockage, paramètres d'instance, répertoire de travail et autres paramètres de génération.

Image Builder prend en charge deux types de recettes :

- Les recettes d'images génèrent des Amazon Machine Images (AMI) que vous pouvez utiliser pour lancer des instances Amazon EC2.
- Les recettes de conteneurs produisent des images de Docker conteneurs stockées dans les référentiels Amazon ECR.

Les recettes sont immuables. Une fois que vous avez créé une recette, vous ne pouvez ni la modifier ni la remplacer. Pour mettre à jour des composants ou toute autre configuration, vous devez créer une nouvelle recette ou une nouvelle version de recette. Image Builder conserve toutes les versions précédentes. Cela vous permet de retracer la lignée des images jusqu'à la recette exacte qui a produit chaque sortie. Vous pouvez toujours appliquer des balises à vos recettes existantes. Pour plus d'informations sur le balisage de vos ressources à l'aide des commandes d'Image Builder AWS CLI, consultez la Balisage des ressources section de ce guide.

> **Tip**
>
> Vous pouvez utiliser des composants gérés par Amazon dans vos recettes ou développer vos propres composants personnalisés. Pour de plus amples informations, veuillez consulter Développez des composants personnalisés pour votre image Image Builder. Pour les recettes d'image qui créent des AMI de sortie, vous pouvez également utiliser des produits et des composants d' AWS Marketplace image. Pour plus d'informations sur l'intégration avec AWS Marketplace les produits, consultezAWS Marketplace intégration dans Image Builder.

Utilisez les rubriques de cette section pour gérer vos recettes EC2 Image Builder, notamment pour répertorier les recettes existantes, afficher leurs détails et créer de nouvelles versions de recettes.

Table des matières

- Comparaison des recettes avec images et des recettes en contenants
- Versionnage des recettes
- Contraintes et limites des recettes
- Répertorier et afficher les détails des recettes avec des images

- Répertorier et afficher les détails des recettes des contenants
- Création d'une nouvelle version d'une recette imagée
- Création d'une nouvelle version d'une recette de contenant
- Suivez la lignée des AMI à l'aide de filigranes
- nettoyer des ressources ;

Comparaison des recettes avec images et des recettes en contenants

Le tableau suivant compare les caractéristiques des recettes d'images et des recettes de conteneurs.

Fonctionnalité	Recette imagée	Recette de contenant
Output	AMI	Dockerimage de conteneur dans Amazon ECR
Sources d'images de base	ID AMI, ARN de l'image Image Builder, paramètre SSM, produit AWS Marketplace	Docker Hubimage, image Amazon ECR, Amazon-managed image, ARN d'image Image Builder
Dockerfile	Non applicable	Obligatoire (modèle avec variables contextuelles)
Référentiel cible	Non applicable	Amazon ECR (obligatoire)
Configuration du stockage	Bloquer les mappages des périphériques pour l'AMI de sortie	Bloquer les mappages de périphériques pour l'instance de build
Données de l'utilisateur	Pris en charge (remplace cloud-init par défaut)	Non pris en charge au niveau des recettes
Contrôle des agents Systems Manager	Configurable (désinstallation après la compilation)	Non configurable
AWS Marketplace composants	Pris en charge	Non pris en charge
Remplacement de la plateforme	Non applicable (détecté depuis l'AMI)	Disponible pour Amazon ECR et les images Docker Hub

Versionnage des recettes

<major>Chaque recette utilise un versionnage sémantique dans le format. <minor>. <patch>. Chaque nœud de version accepte des valeurs comprises entre 0 et 1 073 741 823 (2^30 - 1).

Vous pouvez utiliser un caractère générique (x) à une position pour activer l'incrémentation automatique des versions. Lorsque vous créez une recette avec une version générique, Image Builder la remplace x par le prochain numéro disponible pour cette position.

Modèle	Comportement	Exemple de séquence
`1.0.x`	Incrémente le numéro du patch	1.0.1, 1.0.2, 1.0.3,...
`1.x.0`	Incrémente le nombre mineur	1.1.0, 1.2.0, 1.3.0,...
`x.0.0`	Incrémente le nombre principal	1.0.0, 2.0.0, 3.0.0,...

Les règles suivantes s'appliquent aux versions contenant des caractères génériques :

- Un seul x caractère générique est autorisé par chaîne de version.
- Le caractère générique peut apparaître dans n'importe quelle position (majeure, mineure ou patch).
- Les valeurs numériques dans les positions restantes doivent être des entiers explicites.

> **Tip**
>
> Utilisez des versions génériques dans votre CI/CD automatisation afin de ne pas avoir à suivre et à incrémenter les versions des recettes à la main. Chaque fois que vous créez une recette avec le même caractère générique, Image Builder corrige la prochaine version disponible. Par exemple, une recette qui utilise toujours `1.0.x` resolues to`1.0.1`, then`1.0.2`, et ainsi de suite. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

Contraintes et limites des recettes

Avant de créer une recette, vérifiez les contraintes suivantes.

Contrainte	Limite	Remarques
Composants par recette	20 (par défaut)	Comprend à la fois des composants de construction et de test. Vous pouvez demander une augmentation via AWS Support.
Taille maximale de la recette	25 KO	Taille cumulée de toutes les configurations de composants, y compris les valeurs des paramètres.
Codes de produit par image de sortie	9	Combiné à partir de l'image de base et AWS Marketplace des composants.
Unicité des composants	Un par recette	Le même composant (quelle que soit la version) ne peut apparaître qu'une seule fois.
Nœud de version maximum	1 073 741 823	Par nœud (majeur, mineur ou correctif).
Nom de la formule	Ne peut pas être modifié	Le nom est fixé après la création. Créez une nouvelle recette pour un autre nom.

Les composants doivent également répondre aux exigences de compatibilité suivantes :

- Les composants doivent correspondre à la plate-forme de la recette (Linux, Windows ou macOS).
- Les composants doivent prendre en charge la version du système d'exploitation de l'image parente, lorsque les métadonnées de la version du système d'exploitation sont disponibles.
- Les composants ayant le `DISABLED` statut `DEPRECATED` ou ne peuvent pas être ajoutés aux nouvelles recettes.
- Les composants de durcissement CIS nécessitent une image CIS-published parent.

Répertorier et afficher les détails des recettes avec des images

Cette section décrit les différentes manières de rechercher des informations et de visualiser les détails de vos recettes d'images EC2 Image Builder.

Détails de la recette avec image

- Répertorier les recettes d'images depuis la console
- Répertoriez les recettes illustrées du AWS CLI
- Afficher les détails des recettes illustrées depuis la console
- Obtenez des informations détaillées sur les recettes avec des images sur AWS CLI
- Obtenez les détails de la politique relative aux recettes illustrées auprès du AWS CLI

Répertorier les recettes d'images depuis la console

Pour consulter la liste des recettes d'images créées sous votre compte dans la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez des recettes d'images dans le volet de navigation. Cela affiche une liste des recettes d'images créées sous votre compte.
3. Pour afficher les détails ou créer une nouvelle version de recette, cliquez sur le lien Nom de la recette. Cela ouvre la vue détaillée de la recette.

> **Note**
>
> Vous pouvez également cocher la case à côté du nom de la recette, puis choisir Afficher les détails.

Répertoriez les recettes illustrées du AWS CLI

L'exemple suivant montre comment répertorier toutes vos recettes d'images à l'aide du AWS CLI.

```
aws imagebuilder list-image-recipes
```

La commande renvoie un résumé pour chaque recette d'image, comme indiqué dans l'exemple de sortie suivant.

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageRecipeSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-recipe/1.0.0",
            "name": "my-recipe",
            "platform": "Linux",
            "owner": "123456789012",
            "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/amazon-linux-2023-x86/x.x.x",
            "dateCreated": "2024-01-15T10:30:00.000Z",
            "tags": {
                "Environment": "Production"
            }
        }
    ]
}
```

Vous pouvez filtrer les résultats par propriétaire pour contrôler les recettes renvoyées par la commande.

```
aws imagebuilder list-image-recipes --owner Self
```

`Self`Utilisez-le pour répertorier les recettes que vous possédez ou `Shared` pour répertorier les recettes que d'autres comptes ont partagées avec vous.

```
aws imagebuilder list-image-recipes --owner Shared
```

Pour limiter le nombre de résultats renvoyés par la commande sur chaque page, utilisez le `--max-results` paramètre. Si d'autres résultats sont disponibles, la réponse inclut une `nextToken` valeur que vous transmettez à la commande suivante pour récupérer la page suivante.

```
aws imagebuilder list-image-recipes --max-results 10 --next-token "eyJuZXh0VG9rZW4..."
```

Afficher les détails des recettes illustrées depuis la console

Pour afficher les détails d'une recette d'image spécifique à l'aide de la console Image Builder, sélectionnez la recette d'image à consulter en suivant les étapes décrites dansRépertorier les recettes d'images depuis la console.

Sur la page détaillée de la recette, vous pouvez :

- Supprimez la recette. Pour plus d'informations sur la suppression de ressources dans Image Builder, consultezSupprimer les ressources d'Image Builder obsolètes ou inutilisées.
- Créez une nouvelle version.
- Créez un pipeline à partir de la recette. Après avoir choisi Créer un pipeline à partir de cette recette, vous êtes redirigé vers l'assistant de pipeline. Pour plus d'informations sur la création d'un pipeline Image Builder à l'aide de l'assistant de pipeline, voir Tutoriel : Création d'un pipeline d'images avec une AMI de sortie à partir de l'assistant de console Image Builder

> **Note**
>
> Lorsque vous créez un pipeline à partir d'une recette existante, l'option permettant de créer une nouvelle recette n'est pas disponible.

Obtenez des informations détaillées sur les recettes avec des images sur AWS CLI

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour obtenir les détails d'une recette d'image en spécifiant son Amazon Resource Name (ARN).

```
aws imagebuilder get-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-recipe/1.0.0
```

La commande renvoie la recette d'image complète, illustrée dans l'exemple de sortie suivant.

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageRecipe": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
recipe/1.0.0",
        "name": "my-recipe",
        "description": "My Linux image recipe",
```

```
        "platform": "Linux",
        "version": "1.0.0",
        "components": [
            {
                "componentArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-component/1.0.0/1"
            }
        ],
        "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/amazon-linux-2023-x86/
x.x.x",
        "dateCreated": "2024-01-15T10:30:00.000Z",
        "tags": {},
        "owner": "123456789012"
    }
}
```

Obtenez les détails de la politique relative aux recettes illustrées auprès du AWS CLI

Une politique de recette est une politique basée sur les ressources. Il contrôle l'accès de tous les comptes à votre recette. Lorsque vous partagez une recette via AWS Resource Access Manager (AWS RAM), Image Builder crée automatiquement la politique. Vous pouvez également définir une politique personnalisée à l'aide de la `put-image-recipe-policy` commande.

> **Note**
>
> Si vous définissez la politique des ressources directement auprès de`put-image-recipe-policy`, vous devez ensuite promouvoir la ressource vers un partage de AWS RAM ressources afin qu'elle soit visible par les principaux acteurs avec lesquels vous la partagez. Pour de plus amples informations, veuillez consulter Option 2 : appliquer une politique de ressources et promouvoir vers un partage de ressources existant.

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour obtenir les détails d'une politique de recette d'image en spécifiant son ARN. Si aucune politique n'est associée à la recette, la réponse est vide.

```
aws imagebuilder get-image-recipe-policy --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-recipe/1.0.0
```

Répertorier et afficher les détails des recettes des contenants

Cette section décrit comment vous pouvez trouver des informations et afficher les détails de vos recettes de conteneur EC2 Image Builder.

Détails de la recette du contenant

- Répertorier les recettes de conteneurs dans la console
- Répertoriez les recettes de contenants avec AWS CLI
- Afficher les détails des recettes des contenants dans la console
- Obtenez des détails sur les recettes des contenants grâce au AWS CLI
- Obtenez les détails de la politique relative aux recettes relatives aux contenants grâce au AWS CLI

Répertorier les recettes de conteneurs dans la console

Pour consulter la liste des recettes de conteneurs qui ont été créées sous votre compte dans la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez des recettes de conteneurs dans le volet de navigation. Cela affiche une liste des recettes de conteneurs créées sous votre compte.
3. Pour afficher les détails ou créer une nouvelle version de recette, cliquez sur le lien Nom de la recette. Cela ouvre la vue détaillée de la recette.

> **Note**
>
> Vous pouvez également cocher la case à côté du nom de la recette, puis choisir Afficher les détails.

Répertoriez les recettes de contenants avec AWS CLI

L'exemple suivant montre comment répertorier toutes vos recettes de contenants à l'aide du AWS CLI.

```
aws imagebuilder list-container-recipes
```

La commande renvoie un résumé pour chaque recette de contenant, comme indiqué dans l'exemple de sortie suivant.

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "containerRecipeSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-container-recipe/1.0.0",
            "name": "my-container-recipe",
            "containerType": "DOCKER",
            "platform": "Linux",
            "owner": "123456789012",
            "dateCreated": "2024-01-15T10:30:00.000Z",
            "tags": {}
        }
    ]
}
```

Vous pouvez filtrer les résultats par propriétaire pour contrôler les recettes renvoyées par la commande. `Self`Utilisez-le pour répertorier les recettes que vous possédez ou `Shared` pour répertorier les recettes que d'autres comptes ont partagées avec vous.

```
aws imagebuilder list-container-recipes --owner Self
```

Pour limiter le nombre de résultats renvoyés par la commande sur chaque page, utilisez le `--max-results` paramètre. Si d'autres résultats sont disponibles, la réponse inclut une `nextToken` valeur que vous transmettez à la commande suivante pour récupérer la page suivante.

```
aws imagebuilder list-container-recipes --max-results 10 --next-token
 "eyJuZXh0VG9rZW4..."
```

Afficher les détails des recettes des contenants dans la console

Pour afficher les détails d'une recette de conteneur spécifique à l'aide de la console Image Builder, sélectionnez la recette de conteneur à consulter et suivez les étapes décrites dansRépertorier les recettes de conteneurs dans la console.

Sur la page détaillée de la recette, vous pouvez effectuer les opérations suivantes :

- Supprimez la recette. Pour plus d'informations sur la suppression de ressources dans Image Builder, consultezSupprimer les ressources d'Image Builder obsolètes ou inutilisées.
- Créez une nouvelle version.
- Créez un pipeline à partir de la recette. Une fois que vous avez choisi Créer un pipeline à partir de cette recette, la console ouvre l'assistant de pipeline. Pour plus d'informations sur la création d'un pipeline Image Builder à l'aide de l'assistant de pipeline, voir Tutoriel : Création d'un pipeline d'images avec une AMI de sortie à partir de l'assistant de console Image Builder

> **Note**
>
> Lorsque vous créez un pipeline à partir d'une recette existante, l'option permettant de créer une nouvelle recette n'est pas disponible.

Obtenez des détails sur les recettes des contenants grâce au AWS CLI

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour obtenir les détails d'une recette de conteneur en spécifiant son ARN.

```
aws imagebuilder get-container-recipe --container-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:container-recipe/my-container-recipe/1.0.2
```

La commande renvoie la recette complète du conteneur, illustrée dans l'exemple de sortie suivant.

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "containerRecipe": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-
container-recipe/1.0.2",
        "containerType": "DOCKER",
        "name": "my-container-recipe",
        "description": "My Linux Docker container image",
        "platform": "Linux",
        "components": [
            {
                "componentArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-component/1.0.0/1"
            }
        ],
        "parentImage": "amazonlinux:latest",
```

```
        "dockerfileTemplateData": "FROM
 {{{ imagebuilder:parentImage }}}\n{{{ imagebuilder:environments }}}\n{{{ imagebuilder:componen
        "targetRepository": {
            "service": "ECR",
            "repositoryName": "my-repo"
        },
        "dateCreated": "2024-01-15T10:30:00.000Z",
        "owner": "123456789012"
    }
}
```

Obtenez les détails de la politique relative aux recettes relatives aux contenants grâce au AWS CLI

Une politique de recette est une politique basée sur les ressources. Il contrôle l'accès de tous les comptes à votre recette. Lorsque vous partagez une recette via AWS Resource Access Manager (AWS RAM), Image Builder crée automatiquement la politique. Vous pouvez également définir une politique personnalisée à l'aide de la `put-container-recipe-policy` commande.

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour obtenir les détails d'une politique de recette de conteneur en spécifiant son ARN. Si aucune politique n'est associée à la recette, la réponse est vide.

```
aws imagebuilder get-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-container-recipe/1.0.2
```

Création d'une nouvelle version d'une recette imagée

Cette section explique comment créer une recette imagée ou une nouvelle version d'une recette existante. Une recette d'image définit l'AMI de base, les composants de construction et la configuration. Utilisez cette recette pour créer des Amazon Machine Images (AMI) avec Image Builder. Vous pouvez créer des recettes d'images à partir de la console Image Builder ou à l'aide du AWS CLI.

Table des matières

- Créez une nouvelle version d'image de la recette depuis la console
- Créez une recette imagée à l'aide du AWS CLI
- Importez une machine virtuelle comme image de base dans la console

Créez une nouvelle version d'image de la recette depuis la console

Lorsque vous créez une nouvelle version de recette, c'est pratiquement la même chose que de créer une nouvelle recette. La différence est que certains détails sont présélectionnés pour correspondre à la recette de base, dans la plupart des cas. La liste suivante décrit les différences entre la création d'une nouvelle recette et la création d'une nouvelle version d'une recette existante.

Détails de la recette de base dans la nouvelle version

- Nom : non modifiable.
- Version — Obligatoire. Entrez le numéro de version que vous souhaitez créer dans le format <major>. <minor>. <patch>. Image Builder prend en charge l'incrémentation automatique des versions pour les recettes, ce qui vous permet d'utiliser des caractères génériques dans les versions de vos recettes. Lorsque vous créez une recette avec une version générique`1.0.x`, telle que Image Builder incrémente automatiquement la version (par exemple,`1.0.1`,`1.0.2`,`1.0.3`, etc.). Il n'est donc plus nécessaire de suivre et d'incrémenter manuellement les versions des recettes.
- L'option Sélectionner une image — Pre-selected, mais vous pouvez la modifier. Si vous modifiez le choix de la source de votre image de base, vous risquez de perdre d'autres détails qui dépendent de l'option d'origine que vous avez choisie.

  Pour voir les détails associés à votre sélection d'images de base, choisissez l'onglet correspondant à votre sélection.

  Managed image

  - Système d'exploitation (OS) d'image : non modifiable.
  - Nom de l'image — Pre-selected, en fonction de la combinaison des choix d'images de base que vous avez effectués pour la recette existante. Toutefois, si vous modifiez l'option Sélectionner une image, vous perdez le nom de l'image présélectionnée.
  - Auto-versioning options — Ne correspond pas à votre recette de base. Cette option d'image est définie par défaut sur l'option Utiliser la version du système d'exploitation sélectionnée.

    > **Important**
    >
    > Si vous utilisez le versionnage sémantique pour lancer la création de pipelines, assurez-vous de modifier cette valeur pour Utiliser la dernière version du système d'exploitation disponible. Pour en savoir plus sur la gestion sémantique des versions

pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

AWS Marketplace image

- Abonnements — Cet onglet doit être ouvert et l'image à laquelle vous êtes abonné AWS Marketplace doit être présélectionnée pour correspondre à votre recette de base. Si vous modifiez l'image que votre recette utilise comme image de base, vous risquez de perdre d'autres détails qui dépendent de l'image d'origine que vous avez choisie.

Pour plus d'informations sur les AWS Marketplace produits, consultez la section Acheter des produits dans le Guide de AWS Marketplace l'acheteur.

Custom AMI

Source AMI (obligatoire) - Entrez l'ID AMI ou un paramètre de magasin de paramètres AWS Systems Manager (SSM) contenant un ID AMI à utiliser comme image de base. L'agent SSM doit être préinstallé dans l'AMI sélectionnée.

- ID AMI — Ce paramètre n'est pas prérempli avec votre saisie d'origine. Entrez l'ID AMI de votre image de base. Exemple: *ami-1234567890abcdef1*.
- Paramètre SSM — Entrez le nom ou l'ARN du paramètre SSM Parameter Store qui contient l'ID AMI de votre image de base. Exemple : */ib/test/param* ou `arn:aws:ssm:`*`us-east-1`*`:`*`111122223333`*`:parameter`*`/ib/test/param`*.

- Configuration de l'instance : les paramètres sont présélectionnés, mais vous pouvez les modifier.
  - Agent Systems Manager : vous pouvez activer ou désactiver cette case à cocher pour contrôler l'installation de l'agent Systems Manager sur la nouvelle image. Par défaut, cette case à cocher reste désactivée et Image Builder conserve l'agent Systems Manager dans votre AMI finale. Pour supprimer l'agent Systems Manager de l'image finale, cochez la case.
  - Données utilisateur : vous pouvez utiliser cette zone pour fournir des commandes, ou un script de commande à exécuter, lorsque vous lancez votre instance de build. Toutefois, cette valeur remplace toutes les commandes qu'Image Builder aurait pu ajouter pour s'assurer que Systems Manager est installé. Ces commandes incluent le script de nettoyage qu'Image Builder exécute normalement pour les images Linux avant de créer la nouvelle image.

    Lorsqu'Image Builder lance une instance, les scripts de données utilisateur s'exécutent pendant la phase d'initialisation du cloud, avant le début de l'exécution des composants. Cette étape est enregistrée dans le fichier suivant de l'instance :`var/log/cloud-init.log`.

> **Note**
>
> - Si vous saisissez des données utilisateur, assurez-vous que l'agent Systems Manager est préinstallé sur votre image de base ou que vous incluez l'installation dans vos données utilisateur.
> - Pour les images Linux, assurez-vous que les étapes de nettoyage sont exécutées en incluant une commande pour créer un fichier vide nommé `perform_cleanup` dans votre script de données utilisateur. Image Builder détecte ce fichier et exécute le script de nettoyage avant de créer la nouvelle image. Pour plus d'informations et un exemple de script, consultezMeilleures pratiques de sécurité pour Image Builder.

- Répertoire de travail — Pre-selected, mais vous pouvez le modifier.
- Composants : les composants déjà inclus dans la recette sont affichés dans la section Composants sélectionnés à la fin de chacune des listes de composants (construction et test). Vous pouvez supprimer ou réorganiser les composants sélectionnés en fonction de vos besoins.

  Les composants de renforcement CIS ne suivent pas les règles d'ordre standard des composants dans les recettes d'Image Builder. Les composants de durcissement CIS sont toujours exécutés en dernier pour garantir que les tests de référence s'exécutent par rapport à votre image de sortie.

  > **Note**
  >
  > Les listes de composants de création et de test affichent les composants disponibles en fonction du type de propriétaire du composant. Pour ajouter un composant, choisissez Ajouter des composants de construction, puis sélectionnez le filtre de propriété qui s'applique. Par exemple, pour ajouter un composant de construction associé à un AWS Marketplace produit, sélectionnez`AWS Marketplace`. Cela ouvre un panneau de sélection sur le côté droit de l'interface de la console qui répertorie les AWS Marketplace composants.
  > Pour le composant CIS, sélectionnez`Third party managed`.

  Vous pouvez configurer les paramètres suivants pour le composant que vous avez sélectionné :

  - Options de gestion des versions — Pre-selected, mais vous pouvez les modifier. Nous vous recommandons de choisir l'option Utiliser la dernière version disponible du composant pour vous assurer que les builds de votre image utilisent toujours la dernière version du composant. Si

vous devez utiliser une version de composant spécifique dans votre recette, vous pouvez choisir Spécifier la version du composant et saisir la version dans la zone Version du composant qui apparaît.

- Paramètres d'entrée : affiche les paramètres d'entrée acceptés par le composant. La valeur est pré-remplie avec la valeur de la version précédente de la recette. Si vous utilisez ce composant pour la première fois dans cette recette et qu'une valeur par défaut a été définie pour le paramètre d'entrée, la valeur par défaut apparaît dans la zone Valeur avec du texte grisé. Si aucune autre valeur n'est saisie, Image Builder utilise la valeur par défaut.

  Si un paramètre d'entrée est obligatoire, mais qu'aucune valeur par défaut n'est définie dans le composant, vous devez fournir une valeur. Image Builder ne créera pas la version de la recette s'il manque des paramètres requis et si aucune valeur par défaut n'est définie.

  > **Important**
  >
  > Les paramètres des composants sont des valeurs en texte brut et sont connectés AWS CloudTrail. Nous vous recommandons d'utiliser AWS Secrets Manager ou le AWS Systems Manager Parameter Store pour stocker vos secrets. Pour plus d'informations sur Secrets Manager, voir Qu'est-ce que Secrets Manager ? dans le Guide de AWS Secrets Manager l'utilisateur. Pour plus d'informations sur le stockage des AWS Systems Manager paramètres, consultez la section AWS Systems Manager Parameter Store dans le Guide de AWS Systems Manager l'utilisateur.

Pour développer les paramètres des options de version ou des paramètres d'entrée, vous pouvez choisir la flèche à côté du nom du paramètre. Pour étendre tous les paramètres de tous les composants sélectionnés, vous pouvez activer et désactiver le bouton Tout étendre.

- Stockage (volumes) — sont préremplis. Le nom de l'appareil, le snapshot et les sélections IOPS du volume racine ne sont pas modifiables. Vous pouvez toutefois modifier tous les paramètres restants, tels que la taille. Vous pouvez également ajouter de nouveaux volumes et chiffrer des volumes nouveaux ou existants.

  Pour chiffrer les volumes des images créées par Image Builder sous votre compte dans la région source (où s'exécute la génération), vous devez utiliser le chiffrement du volume de stockage dans la recette d'image. Le cryptage qui s'exécute pendant la phase de distribution de la version concerne uniquement les images distribuées vers d'autres comptes ou régions.

> **Note**
>
> Si vous utilisez le chiffrement pour vos volumes, vous devez sélectionner la clé pour chaque volume séparément, même si la clé est la même que celle utilisée pour le volume racine.

> **Note**
>
> Si vous n'avez pas de recette existante, utilisez l'assistant de pipeline Image Builder ou choisissez Recettes d'images dans le volet de navigation, puis choisissez Créer une recette d'image. Les champs décrits dans cette section s'appliquent à la fois aux nouvelles recettes et aux nouvelles versions.

Pour créer une nouvelle version d'une recette imagée, procédez comme suit :

1. En haut de la page des détails de la recette, choisissez Créer une nouvelle version. Cela ouvre la page Créer une recette avec une image.
2. Pour créer la nouvelle version, apportez vos modifications, puis choisissez Créer une recette.

   Votre image finale peut contenir jusqu'à neuf codes de produits issus de produits et de composants AWS Marketplace d'image. Image Builder renvoie une erreur lors de la création de la recette si votre image de base et ses composants dépassent cette limite.

Passez en revue les contraintes suivantes avant de créer une recette d'image :

- Votre image finale peut contenir au maximum neuf codes de AWS Marketplace produit, combinés à partir de l'image de base et des composants. Un code produit est un identifiant associé à une AMI payante ou prise en charge à des fins de facturation et de licence. AWS Marketplace

  > **Tip**
  >
  > Les composants publiés via AWS Marketplace peuvent également porter des codes de produit. Image Builder lit les codes de produit à partir des métadonnées de chaque composant. Il compte les identifiants de code de produit distincts, ainsi que les codes de produit de l'AMI de base, dans la limite.

Pour vérifier les codes de produit sur une AMI de base, exécutez la commande Amazon EC2 describe-images et interrogez le `ProductCodes` champ, comme illustré dans l'exemple suivant :

```
aws ec2 describe-images --image-ids ami-example12345 --query
 "Images[].ProductCodes"
```

- Vous pouvez inclure jusqu'à 20 composants (build et test combinés). Il s'agit de la limite par défaut, que vous pouvez demander à augmenter AWS Support.
- Le même composant ne peut pas apparaître plus d'une fois dans une recette.
- Tous les composants doivent correspondre à la plate-forme (Linux, Windows ou macOS) de votre image de base. Les composants doivent également prendre en charge la version du système d'exploitation lorsque ces métadonnées sont disponibles.

Pour plus d'informations sur la création d'une recette d'image lorsque vous créez un pipeline d'images, consultez Étape 2 : Choisissez la recette la section Démarrage de ce guide.

Créez une recette imagée à l'aide du AWS CLI

Pour créer une recette d'image à l'aide de la `create-image-recipe` commande Image Builder dans le AWS CLI, procédez comme suit :

Conditions préalables

Avant d'exécuter les commandes Image Builder de cette section pour créer une recette d'image à partir de AWS CLI, vous pouvez éventuellement créer des composants que la recette utilise. L'exemple de recette d'image présenté à l'étape suivante fait référence à des exemples de composants créés dans la Créez un composant personnalisé à partir du AWS CLI section de ce guide.

Si vous souhaitez inclure des composants dans votre recette, notez les ARN que vous souhaitez inclure. Vous pouvez également créer des recettes sans aucun composant pour tester les AMI existantes ou les flux de travail réservés à la distribution.

1. Créer un fichier JSON d'entrée CLI

   Vous pouvez fournir toutes les entrées de la create-image-recipe commande avec des paramètres de commande en ligne. Cependant, la commande qui en résulte peut être assez

longue. Pour rationaliser la commande, vous pouvez plutôt fournir un fichier JSON contenant tous les paramètres de la recette.

> **Note**
>
> La convention de dénomination des valeurs de données du fichier JSON suit le modèle spécifié pour les paramètres de demande d'opération de l'API Image Builder. Pour passer en revue les paramètres de demande d'opération d'API, consultez la CreateImageRecipe commande dans le document de référence de l'API EC2 Image Builder.
> Pour fournir les valeurs de données sous forme de paramètres de ligne de commande, reportez-vous aux noms de paramètres spécifiés dans la référence des AWS CLI commandes.

Voici un résumé des paramètres spécifiés dans ces exemples :

- name (chaîne, obligatoire) — Le nom de la recette imagée.
- description (chaîne) — Description de la recette imagée.
- parentImage (chaîne, obligatoire) : image que la recette d'image utilise comme base pour votre image personnalisée. Vous pouvez spécifier l'image parente à l'aide de l'une des options suivantes :
  - ID AMI : à utiliser lorsque vous avez une AMI spécifique (par exemple`ami-1234567890abcdef1`) que vous souhaitez personnaliser. Comme un ID AMI est statique, nous vous recommandons plutôt d'utiliser un ARN d'image Image Builder ou un paramètre SSM si vous souhaitez que votre recette récupère automatiquement les nouvelles images de base.
  - ARN de la ressource image Image Builder : à utiliser lorsque vous souhaitez créer à partir d'une image Image Builder. Cela inclut Amazon-managed les images fournies par Image Builder pour les systèmes d'exploitation les plus courants, ainsi que les images que vous avez créées précédemment. Spécifiez l'ARN de l'image avec une version sémantique ou utilisez un caractère générique de version pour toujours obtenir la dernière version.
  - AWS Systems Manager Paramètre de stockage des paramètres (SSM) : à utiliser lorsque vous souhaitez que votre recette récupère automatiquement de nouveaux ID AMI sans créer de nouvelle version de recette. Préfixez le paramètre avec`ssm:`, suivi du nom du paramètre

ou de l'ARN. C'est idéal pour les pipelines qui doivent toujours utiliser la dernière image de base Amazon Linux ou Windows.

- AWS Marketplace ID de produit : à utiliser lorsque vous vous abonnez à un produit AMI AWS Marketplace et que vous souhaitez le personnaliser davantage. Vous pouvez spécifier l'identifiant du produit dans l'un des formats suivants :
  - Un identifiant unique universel (UUID), par exemple. `722e1255-be34-4bbe-9ce6-80634c9d01f7`
  - Un identifiant de produit avec le `prod-` préfixe, par exemple`prod-example12345`.

  Abonnez-vous au produit avant de l'utiliser comme image mère. Pour de plus amples informations, veuillez consulter AWS Marketplace intégration dans Image Builder.

  > **Note**
  >
  > - Lorsque vous utilisez un ID AMI ou un paramètre SSM qui se résout en AMI, vous devez avoir accès à cette AMI dans votre compte. L'AMI doit également exister dans la même région que celle où Image Builder exécute la génération.
  > - Les exemples Linux et macOS spécifient un ID AMI, tandis que l'exemple Windows utilise un ARN d'image Image Builder.
  > - Le `parentImage` paramètre accepte un ID AMI, un ARN d'image Image Builder, une référence de `ssm:` paramètre ou un identifiant de AWS Marketplace produit. Si la valeur ne correspond à aucun de ces formats, Image Builder renvoie un `InvalidParameterValueException` pour le `parentImage` paramètre.

- <major>SemanticVersion (chaîne, obligatoire) — Entrez le numéro de version que vous souhaitez créer au format. <minor>. <patch>. Image Builder prend en charge l'incrémentation automatique des versions pour les recettes, ce qui vous permet d'utiliser des caractères génériques dans vos versions de recettes. Lorsque vous créez une recette avec une version générique`1.0.x`, telle que Image Builder incrémente automatiquement la version (par exemple,`1.0.1`,`1.0.2`,`1.0.3`, etc.). Il n'est donc plus nécessaire de suivre et d'incrémenter manuellement les versions des recettes. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder
- components (tableau, facultatif) — Contient un tableau d'`ComponentConfiguration`objets. Les composants sont facultatifs : vous pouvez créer des recettes sans aucun composant pour les flux de test ou de distribution :

> **Note**
>
> Image Builder installe les composants dans l'ordre que vous avez indiqué dans la recette. Cependant, les composants de renforcement CIS sont toujours exécutés en dernier pour garantir que les tests de référence s'exécutent par rapport à votre image de sortie.

- componentArn (chaîne, obligatoire) — L'ARN du composant.

  > **Tip**
  >
  > Pour utiliser l'un des exemples afin de créer votre propre recette d'image, vous devez remplacer les exemples d'ARN par les ARN des composants que vous utilisez pour votre recette.

- paramètres (tableau d'objets) — Contient un tableau d'`ComponentParameter`objets. Si un paramètre d'entrée est obligatoire, mais qu'aucune valeur par défaut n'est définie dans le composant, vous devez fournir une valeur. Image Builder ne créera pas la version de la recette s'il manque des paramètres requis et si aucune valeur par défaut n'est définie.

  > **Important**
  >
  > Les paramètres des composants sont des valeurs en texte brut et sont connectés AWS CloudTrail. Nous vous recommandons d'utiliser AWS Secrets Manager ou le AWS Systems Manager Parameter Store pour stocker vos secrets. Pour plus d'informations sur Secrets Manager, voir Qu'est-ce que Secrets Manager ? dans le Guide de AWS Secrets Manager l'utilisateur. Pour plus d'informations sur AWS Systems Manager Parameter Store, voir AWS Systems Manager Parameter Store dans le Guide de AWS Systems Manager l'utilisateur.

  - name (chaîne, obligatoire) — Nom du paramètre de composant à définir.
  - valeur (tableau de chaînes, obligatoire) — Contient un tableau de chaînes pour définir la valeur du paramètre de composant nommé. Si une valeur par défaut est définie pour le composant et qu'aucune autre valeur n'est fournie, AWSTOE utilise la valeur par défaut.

- AMIWatermarks (tableau de chaînes, facultatif) : liste de noms de filigranes à associer aux AMI de sortie créées à partir de cette recette. Les filigranes sont des marqueurs de lignage qui vous permettent de retracer l'histoire de vos images. Spécifiez de 1 à 5 noms. Chaque nom doit comporter de 3 à 128 caractères et peut inclure des lettres, des chiffres, des espaces et les caractères spéciaux suivants :`( ) [ ] . / - ' @ _`. Pour de plus amples informations, veuillez consulter Suivez la lignée des AMI à l'aide de filigranes.
- supplémentaire InstanceConfiguration (objet) : spécifiez des paramètres supplémentaires et lancez des scripts pour vos instances de build.
  - systems ManagerAgent (objet) : contient les paramètres de l'agent Systems Manager sur votre instance de build.

    > **Note**
    >
    > Les recettes Windows ne prennent pas en charge `systemsManagerAgent` cette configuration. Si vous l'incluez avec une image parent Windows, la demande échoue.

  - uninstall AfterBuild (booléen) : contrôle si l'agent Systems Manager est supprimé de votre image de build finale avant de créer la nouvelle AMI. Si cette option est définie sur`true`, l'agent est supprimé de l'image finale. Si l'option est définie sur`false`, l'agent reste actif afin qu'il soit inclus dans la nouvelle AMI. La valeur par défaut est `false`.

    > **Note**
    >
    > Si l'`uninstallAfterBuild`attribut n'est pas inclus dans le fichier JSON et que les conditions suivantes sont remplies, Image Builder supprime l'agent Systems Manager de l'image finale afin qu'il ne soit pas disponible dans l'AMI :
    >
    > - Le `userDataOverride` champ est vide ou a été omis dans le fichier JSON.
    > - Image Builder a automatiquement installé l'agent Systems Manager sur l'instance de génération pour un système d'exploitation sur lequel l'agent n'était pas préinstallé sur l'image de base.

  - user DataOverride (chaîne) — Fournissez des commandes ou un script de commande à exécuter lorsque vous lancez votre instance de build.

> **Note**
>
> Les données utilisateur sont toujours codées en base 64.
> Par exemple, les commandes suivantes sont codées comme `IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg==` :
>
> ```
> #!/bin/bash
> mkdir -p /var/bb/
> touch /var
> ```
>
> L'exemple Linux utilise cette valeur codée.

Linux

L'image de base (`parentImage`propriété) de l'exemple suivant est une AMI. Lorsque vous utilisez une AMI, vous devez avoir accès à l'AMI et celle-ci doit se trouver dans la région source (la même région où Image Builder exécute la commande). Enregistrez le fichier sous `create-image-recipe.json` et utilisez-le dans la create-image-recipe commande.

```
{
"name": "BB Ubuntu Image recipe",
"description": "Hello World image recipe for Linux.",
"parentImage": "ami-1234567890abcdef1",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:111122223333:component/bb$"
 }
],
"additionalInstanceConfiguration": {
 "systemsManagerAgent": {
   "uninstallAfterBuild": true
 },
 "userDataOverride": "IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg=="
}
}
```

Windows

L'exemple suivant fait référence à la dernière version de l'image de base complète en anglais de Windows Server 2016. Dans cet exemple, l'ARN fait référence à la dernière image en fonction des filtres de version sémantiques que vous avez spécifiés :`arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x`.

```
{
"name": "MyBasicRecipe",
"description": "This example image recipe creates a Windows 2016 image.",
"parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:111122223333:component/my-example-component/2019.12.02/1"
 },
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:111122223333:component/my-imported-component/1.0.0/1"
 }
]
}
```

> **Note**
>
> Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

macOS

L'image de base (`parentImage`propriété) de l'exemple suivant est une AMI. Lorsque vous utilisez une AMI, vous devez avoir accès à l'AMI et celle-ci doit se trouver dans la région source (la même région où Image Builder exécute la commande). Enregistrez le fichier sous `create-image-recipe.json` et utilisez-le dans la create-image-recipe commande.

```
{
"name": "macOS Catalina Image recipe",
"description": "Hello World image recipe for macOS.",
"parentImage": "ami-1234567890abcdef1",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-
west-2:111122223333:component/catalina$"
 }
],
"additionalInstanceConfiguration": {
 "systemsManagerAgent": {
   "uninstallAfterBuild": true
 },
 "userDataOverride": "IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg=="
}
}
```

Exemple : recette sans composants

Vous pouvez créer des recettes sans aucun composant pour tester les AMI existantes ou les flux de travail réservés à la distribution. L'exemple suivant montre une recette qui utilise une AMI existante sans appliquer de composants supplémentaires :

```
{
 "name": "Test Distribution Recipe",
 "description": "Recipe for testing and distributing existing AMI without
 modifications.",
 "parentImage": "ami-1234567890abcdef1",
 "semanticVersion": "1.0.0",
 "additionalInstanceConfiguration": {
  "systemsManagerAgent": {
    "uninstallAfterBuild": true
  }
 }
}
```

2. Créez la recette

   Utilisez la commande suivante pour créer la recette. Entrez le nom du fichier JSON que vous avez créé à l'étape précédente dans le `--cli-input-json` paramètre :

   ```
   aws imagebuilder create-image-recipe --cli-input-json file://create-image-recipe.json
   ```

   > **Note**
   >
   > - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
   > - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

   Votre image finale peut contenir jusqu'à neuf codes de produits issus de produits et de composants AWS Marketplace d'image. Image Builder renvoie une erreur lors de la création de la recette si votre image de base et ses composants dépassent cette limite lorsque vous exécutez la `create-image-recipe` commande.

   La commande renvoie l'ARN de la nouvelle recette d'image, comme indiqué dans l'exemple de sortie suivant.

   ```
   {
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-recipe/1.0.1"
   }
   ```

3. Vérifiez votre recette

   Pour vérifier qu'Image Builder a créé votre recette comme prévu, exécutez la get-image-recipe commande avec l'ARN renvoyé à l'étape précédente.

```
aws imagebuilder get-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-recipe/1.0.1
```

Répertoire de travail

Le répertoire de travail spécifie le chemin du système de fichiers sur l'instance de génération où Image Builder exécute les opérations de génération et de test. Image Builder transmet ce chemin en AWS Systems Manager tant que répertoire de travail pour l'exécution des commandes.

Si vous ne spécifiez pas de répertoire de travail, Image Builder utilise les valeurs par défaut suivantes :

- Linux : `/tmp`
- macOS : `/tmp`
- Windows: `C:/`

Pour Windows, le chemin ne peut pas contenir de guillemets doubles (").

> **Tip**
>
> Définissez le répertoire de travail sur un chemin situé sur un volume attaché plus important lorsque vos composants ont besoin d'un espace disque temporaire supérieur à celui fourni par le répertoire par défaut lors de l'installation.

Balises AMI

Vous pouvez appliquer des balises à l'AMI créée par Image Builder pendant la phase de création, avant la distribution de l'image. Image Builder applique ces balises à l'AMI, séparément des balises de ressources figurant sur la recette elle-même. Spécifiez les balises AMI à l'aide du `amiTags` paramètre de votre fichier d'entrée JSON.

> **Note**
>
> Vous ne pouvez pas utiliser les clés de tag réservées `CreatedBy` ou`Ec2ImageBuilderArn`. Image Builder gère ces clés automatiquement.

```
{
 "name": "tagged-recipe",
 "semanticVersion": "1.0.0",
 "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/amazon-linux-2023-x86/x.x.x",
 "components": [
  {
   "componentArn": "arn:aws:imagebuilder:us-west-2:aws:component/update-linux/x.x.x"
  }
 ],
 "amiTags": {
  "Environment": "Production",
  "Team": "Platform",
  "CostCenter": "12345"
 }
}
```

Erreurs courantes et résolution des problèmes

Le tableau suivant répertorie les erreurs courantes que vous pouvez rencontrer lors de la création d'une recette d'image, ainsi que la manière de les résoudre.

Erreur	Cause	Résolution
« Un composant peut être spécifié dans une recette d'image au plus une fois »	Un composant dupliqué (même avec une version différent e).	Supprimez le doublon. N'utilisez qu'une seule version de chaque composant.
« L'ARN du composant ne correspond pas à la plate-forme d'image parent »	Une inadéquation de plateforme.	Vérifiez votre plateforme d'image de base. Utilisez des composants Linux avec des AMI Linux, etc.
« L'ARN du composant ne prend pas en charge la version du système d'exploitation de l'image parent »	Une incompatibilité entre les versions du système d'exploit ation.	Vérifiez les versions du système d'exploitation prises en charge par le composant. Utilisez une version de composant compatible.
« Image Builder ne prend pas en charge la configuration de l'agent SSM sous Windows »	Vous avez spécifié `systemsMa nagerAgent` avec	Supprimez le `additiona lInstanceConfigura`

Erreur	Cause	Résolution
	une image parent Windows.	`tion.systemsManage rAgent` bloc.
« Vous avez dépassé la taille cumulée maximale des composant s de 25 Ko »	Paramètres trop nombreux ou trop grands.	Réduisez le nombre de paramètre s ou raccourcissez les valeurs des paramètres.
« La recette peut contenir au maximum 9 produits du marché »	Trop de codes AWS Marketplace produits.	Réduisez le nombre de AWS Marketplace composant s ou utilisez une image de base contenant moins de codes produits.
« Les recettes contenant des composants de la place de marché doivent contenir au moins un composant de compilation »	La recette fait référence à AWS Marketplace des composants mais ne comporte aucun composant de construction.	Ajoutez au moins un composant de construction à la recette.
« La version sémantique fournie ne suit pas le format requis »	Chaîne de version non valide.	Utilisez le format major.min or.patch avec un caractère générique facultatif. x
« Le composant ARN... est obsolète et ne peut pas être inclus dans les nouvelles recettes » (ou « est désactivé »)	Un composant référencé possède un `DISABLED` statut `DEPRECATED` ou.	Effectuez une mise à jour vers la version actuelle du composant.

Importez une machine virtuelle comme image de base dans la console

Cette section explique comment importer une machine virtuelle (VM) comme image de base pour votre recette d'image. Les autres étapes liées à la création d'une recette ou d'une version de recette ne sont pas abordées ici. Pour des étapes supplémentaires permettant de créer une nouvelle recette d'image à l'aide de l'assistant de création de pipeline de la console Image Builder, consultezAssistant

de pipeline : création d'une AMI. Pour plus d'informations sur la création d'une nouvelle recette imagée ou d'une nouvelle version de recette, consultezCréation d'une nouvelle version d'une recette imagée.

Pour importer une machine virtuelle en tant qu'image de base de votre recette d'image dans la console Image Builder, procédez comme suit, ainsi que toutes les autres étapes requises, pour créer votre recette ou votre version de recette.

1. Dans la section Sélectionner une image pour l'image de base, sélectionnez l'option Importer l'image de base.
2. Choisissez le système d'exploitation (OS) d'image et la version du système d'exploitation comme vous le feriez normalement.

Configuration de l'importation de machines virtuelles

Lorsque vous exportez votre machine virtuelle depuis son environnement de virtualisation, ce processus crée un ensemble d'un ou de plusieurs fichiers conteneurs de disque qui agissent comme des instantanés de l'environnement, des paramètres et des données de votre machine virtuelle. Vous pouvez utiliser ces fichiers pour importer votre machine virtuelle en tant qu'image de base pour votre recette d'image. Pour plus d'informations sur l'importation de machines virtuelles dans Image Builder, consultez Importation et exportation d'images de machines virtuelles

Pour spécifier l'emplacement de votre source d'importation, procédez comme suit :

Source d'importation

Spécifiez la source du premier conteneur de disque ou du premier instantané d'image de machine virtuelle à importer dans la section Conteneur de disque 1.

1. Source : il peut s'agir d'un compartiment S3 ou d'un instantané EBS.
2. Sélectionnez l'emplacement S3 du disque — Entrez l'emplacement dans Amazon S3 où vos images de disque sont stockées. Pour rechercher l'emplacement, choisissez Parcourir S3.
3. Pour ajouter un conteneur de disque, choisissez Ajouter un conteneur de disque.

Rôle IAM

Pour associer un rôle IAM à la configuration d'importation de votre machine virtuelle, sélectionnez le rôle dans la liste déroulante des rôles IAM ou choisissez Créer un nouveau rôle pour en créer

un nouveau. Si vous créez un nouveau rôle, la page de console IAM Roles s'ouvre dans un onglet distinct.

Paramètres avancés — facultatif

Les paramètres suivants sont facultatifs. Ces paramètres vous permettent de configurer le chiffrement, les licences, les balises, etc. pour l'image de base créée par l'importation.

Général

1. Spécifiez un nom unique pour l'image de base. Si vous ne saisissez aucune valeur, l'image de base hérite du nom de la recette.
2. Spécifiez une version pour l'image de base. Utilisez le format suivant : *<major>.<minor>.<patch>*. Si vous ne saisissez aucune valeur, l'image de base hérite de la version de la recette.
3. Vous pouvez également saisir une description pour l'image de base.

Architecture d'image de base

Pour spécifier l'architecture de la source d'importation de votre machine virtuelle, sélectionnez une valeur dans la liste Architecture.

Chiffrement

Si les images de disque de votre machine virtuelle sont chiffrées, vous devez fournir une clé à utiliser pour le processus d'importation. Pour spécifier et AWS KMS key pour l'importation, sélectionnez une valeur dans la liste Chiffrement (clé KMS). La liste contient les clés KMS auxquelles votre compte a accès dans la région actuelle.

Gestion des licences

Lorsque vous importez une machine virtuelle, le processus d'importation détecte automatiquement le système d'exploitation de la machine virtuelle et applique la licence appropriée à l'image de base. En fonction de la plate-forme de votre système d'exploitation, les types de licences sont les suivants :

- Licence incluse — Une AWS licence adaptée à votre plateforme est appliquée à votre image de base.
- Apportez votre propre licence (BYOL) : conserve la licence de votre machine virtuelle, le cas échéant.

Pour joindre des configurations de licence créées avec AWS License Manager à votre image de base, sélectionnez-les dans la liste des noms de configuration de licence. Pour plus d'informations sur License Manager, consultez la section Utilisation des configurations de licence dans le Guide de l'utilisateur du License Manager.

> **Note**
>
> - Les configurations de licence contiennent des règles de licence basées sur les termes de vos contrats d'entreprise.
> - Linux ne prend en charge que les licences BYOL.

Tags (image de base)

Les balises utilisent des paires clé-valeur pour attribuer du texte consultable à votre ressource Image Builder. Pour spécifier des balises pour l'image de base importée, entrez des paires clé-valeur dans les zones Clé et Valeur.

Pour ajouter une balise, choisissez Ajouter une balise. Pour supprimer une balise, choisissez Remove tag (Supprimer une balise).

Création d'une nouvelle version d'une recette de contenant

Cette section explique comment créer une recette en contenant ou une nouvelle version d'une recette existante. Une recette de conteneur définit trois éléments clés : l'image de base, les composants de construction et le référentiel cible. Utilisez cette recette pour produire des images de Docker conteneur qu'Image Builder stocke dans Amazon ECR.

Table des matières

- Créez une nouvelle version de recette de conteneur à l'aide de la console
- Créez une recette de contenant à l'aide du AWS CLI

Créez une nouvelle version de recette de conteneur à l'aide de la console

La création d'une nouvelle version d'une recette en contenant est pratiquement la même chose que la création d'une nouvelle recette. La différence est que certains détails sont présélectionnés pour correspondre à la recette de base, dans la plupart des cas. La liste suivante décrit les différences entre la création d'une nouvelle recette et la création d'une nouvelle version d'une recette existante.

Détails de la recette

- Nom : non modifiable.
- Version — Obligatoire. Ce détail n'est pas pré-rempli avec la version actuelle ou aucune sorte de séquence. Entrez le numéro de version que vous souhaitez créer au format major.minor.patch. Si la version existe déjà, Image Builder renvoie une erreur.

Image de base

- Sélectionnez l'option d'image — Pre-selected, mais modifiable. Si vous modifiez le choix de la source de votre image de base, vous risquez de perdre d'autres détails qui dépendent de l'option d'origine que vous avez choisie.

  Pour les images de Docker conteneur, vous pouvez choisir entre des images publiques hébergées surDockerHub, des images de conteneur existantes dans Amazon ECR ou des images de Amazon-managed conteneur. Pour voir les détails associés à votre sélection d'images de base, choisissez l'onglet correspondant à votre sélection.

  Managed images

  - Système d'exploitation (OS) d'image : non modifiable.
  - Nom de l'image — Pre-selected, en fonction de la combinaison des choix d'images de base que vous avez effectués pour la recette existante. Toutefois, si vous modifiez l'option Sélectionner une image, vous perdez le nom de l'image présélectionnée.
  - Auto-versioning options — Ne correspond pas à votre recette de base. Auto-versioning options utilise par défaut l'option Utiliser la version du système d'exploitation sélectionnée.

    > **Important**
    >
    > Si vous utilisez le versionnage sémantique pour lancer la création de pipelines, assurez-vous de modifier cette valeur pour Utiliser la dernière version du système d'exploitation disponible. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder

  ECR image

  - Système d'exploitation (OS) d'image —Pre-selected, mais modifiable.
  - Version du système d'exploitation — Pre-selected, mais modifiable.

- ID d'image ECR — Pre-filled, mais modifiable.

Docker Hub image

- Système d'exploitation (OS) d'image : non modifiable.
- Version du système d'exploitation — Pre-selected, mais modifiable.
- DockerID de l'image — Pre-filled, mais modifiable.

Configuration de l'instance

- Source AMI (obligatoire) : identifiez une AMI personnalisée à utiliser comme image de base pour votre instance de création et de test de conteneur. Il peut s'agir d'un ID AMI ou d'un paramètre de magasin de paramètres AWS Systems Manager (SSM) contenant un ID AMI.
  - ID AMI — Ce paramètre n'est pas prérempli avec votre saisie d'origine. Entrez l'ID AMI de votre image de base. Exemple: *ami-1234567890abcdef1*.
  - Paramètre SSM — Entrez le nom ou l'ARN du paramètre SSM Parameter Store qui contient l'ID AMI de votre image de base. Exemple : */ib/test/param* ou `arn:aws:ssm:`*us-east-1*`:`*111122223333*`:parameter`*/ib/test/param*.
- Stockage (volumes)

  EBS volume 1 (racine AMI) — Pre-filled. Vous ne pouvez pas modifier le nom de l'appareil, l'instantané ou les sélections IOPS du volume racine. Vous pouvez toutefois modifier tous les paramètres restants, tels que la taille. Vous pouvez également ajouter de nouveaux volumes.

  > **Note**
  >
  > Si vous avez spécifié une AMI de base qui a été partagée avec vous depuis un autre compte, les instantanés de tous les volumes secondaires spécifiés doivent également être partagés avec votre compte.

Répertoire de travail

- Chemin du répertoire de travail — Pre-filled, mais modifiable.

Éléments

- Composants : les composants déjà inclus dans la recette sont affichés dans la section Composants sélectionnés à la fin de chacune des listes de composants (construction et test). Vous pouvez supprimer ou réorganiser les composants sélectionnés en fonction de vos besoins.

  Les composants de renforcement CIS ne suivent pas les règles d'ordre standard des composants dans les recettes d'Image Builder. Les composants de renforcement CIS sont toujours exécutés en dernier pour garantir que les tests de référence s'exécutent par rapport à votre image de sortie.

  > **Note**
  >
  > Les listes de composants de création et de test affichent les composants disponibles en fonction du type de propriétaire du composant. Pour ajouter un composant, choisissez Ajouter des composants de construction, puis sélectionnez le filtre de propriété qui s'applique. Par exemple, pour ajouter un composant de construction associé à un AWS Marketplace produit, sélectionnez`AWS Marketplace`. Cela ouvre un panneau de sélection sur le côté droit de l'interface de la console qui répertorie les AWS Marketplace composants.
  > Pour le composant CIS, sélectionnez`Third party managed`.

  Vous pouvez configurer les paramètres suivants pour le composant que vous avez sélectionné :

  - Options de gestion des versions — Pre-selected, mais vous pouvez les modifier. Nous vous recommandons de choisir l'option Utiliser la dernière version disponible du composant pour vous assurer que les builds de votre image utilisent toujours la dernière version du composant. Si vous devez utiliser une version de composant spécifique dans votre recette, vous pouvez choisir Spécifier la version du composant et saisir la version dans la zone Version du composant qui apparaît.
  - Paramètres d'entrée : affiche les paramètres d'entrée acceptés par le composant. La valeur est pré-remplie avec la valeur de la version précédente de la recette. Si vous utilisez ce composant pour la première fois dans cette recette et qu'une valeur par défaut a été définie pour le paramètre d'entrée, la valeur par défaut apparaît dans la zone Valeur avec du texte grisé. Si aucune autre valeur n'est saisie, Image Builder utilise la valeur par défaut.

Si un paramètre d'entrée est obligatoire, mais qu'aucune valeur par défaut n'est définie dans le composant, vous devez fournir une valeur. Image Builder ne créera pas la version de la recette s'il manque des paramètres requis et si aucune valeur par défaut n'est définie.

> **Important**
>
> Les paramètres des composants sont des valeurs en texte brut et sont connectés AWS CloudTrail. Nous vous recommandons d'utiliser AWS Secrets Manager ou le AWS Systems Manager Parameter Store pour stocker vos secrets. Pour plus d'informations sur Secrets Manager, voir Qu'est-ce que Secrets Manager ? dans le Guide de AWS Secrets Manager l'utilisateur. Pour plus d'informations sur le stockage des AWS Systems Manager paramètres, consultez la section AWS Systems Manager Parameter Store dans le Guide de AWS Systems Manager l'utilisateur.

Pour développer les paramètres des options de version ou des paramètres d'entrée, vous pouvez choisir la flèche à côté du nom du paramètre. Pour étendre tous les paramètres de tous les composants sélectionnés, vous pouvez activer et désactiver le bouton Tout étendre.

Modèle Dockerfile

- Modèle Dockerfile — Pre-filled, mais modifiable. Vous pouvez spécifier l'une des variables contextuelles suivantes qu'Image Builder remplace par les informations de génération lors de l'exécution.

  Image du parent (obligatoire)

  Au moment de la création, cette variable correspond à l'image de base de votre recette.

  Exemple :

  ```
  FROM
  {{{ imagebuilder:parentImage }}}
  ```

environnements (obligatoire si des composants sont spécifiés)

Au moment de la génération, cette variable est résolue en fonction des directives Dockerfile qui placent vos composants dans le contexte de génération. Ces directives copient les scripts des composants dans l'image et définissent les variables d'environnement que ces scripts utilisent. Il prépare l'environnement mais n'exécute pas les composants que vous spécifiez. Placez cette variable avant la `components` variable.

Exemple :

```
{{{ imagebuilder:environments }}}
```

composants (en option)

Au moment de la génération, cette variable correspond à la commande qui exécute les scripts de génération et de test des composants pour les composants inclus dans la recette du conteneur, puis supprime ces scripts de l'image. Il s'agit de la variable qui exécute vos composants. Placez-le après la `environments` variable.

Exemple :

```
{{{ imagebuilder:components }}}
```

Référentiel cible

- Nom du référentiel cible : référentiel Amazon ECR dans lequel est stockée votre image de sortie si aucun autre référentiel n'est spécifié dans la configuration de distribution de votre pipeline pour la région où le pipeline s'exécute (Région 1).

Pour créer une nouvelle version de recette de conteneur :

1. Sur la page des détails de la recette du contenant, choisissez Créer une nouvelle version en haut de la page. La console ouvre la page Créer une recette pour les recettes en conteneur.
2. Pour créer la nouvelle version, apportez vos modifications, puis choisissez Créer une recette.

Pour plus d'informations sur la création d'une recette de conteneur lorsque vous créez un pipeline d'images, consultez Étape 2 : Choisissez la recette la section Démarrage de ce guide.

Créez une recette de contenant à l'aide du AWS CLI

Pour créer une recette de conteneur Image Builder à l'aide de la `imagebuilder create-container-recipe` commande contenue dans le AWS CLI, procédez comme suit :

Conditions préalables

Avant d'exécuter les commandes Image Builder de cette section pour créer une recette de conteneur avec le AWS CLI, vous devez créer les composants que la recette utilisera. L'exemple de recette de contenant présenté à l'étape suivante fait référence à des exemples de composants créés dans la Créez un composant personnalisé à partir du AWS CLI section de ce guide.

Après avoir créé vos composants, ou si vous utilisez des composants existants, notez les ARN que vous souhaitez inclure dans la recette.

1. Créer un fichier JSON d'entrée CLI

   Vous pouvez fournir toutes les entrées de la create-container-recipe commande avec des paramètres de commande en ligne. Cependant, la commande qui en résulte peut être assez longue. Pour rationaliser la commande, vous pouvez plutôt fournir un fichier JSON contenant tous les paramètres de recette du conteneur

   > **Note**
   >
   > La convention de dénomination des valeurs de données du fichier JSON suit le modèle spécifié pour les paramètres de demande d'opération de l'API Image Builder. Pour passer en revue les paramètres de demande de fonctionnement de l'API, consultez la CreateContainerRecipe commande dans le document de référence de l'API EC2 Image Builder.
   > Pour fournir les valeurs de données sous forme de paramètres de ligne de commande, reportez-vous aux noms de paramètres spécifiés dans la référence des AWS CLI commandes.

   Voici un résumé des paramètres de cet exemple :

   - components (tableau d'objets, facultatif) — Contient un tableau d'`ComponentConfiguration`objets. Les composants sont facultatifs pour les recettes de contenants. Vous pouvez personnaliser l'image de sortie avec les composants BUILD, avec

des instructions dans votre modèle Dockerfile (par exemple, `RUN` ou `COPY` des instructions), ou les deux. Les composants TEST valident l'image mais ne la modifient pas.

> **Note**
>
> Image Builder installe les composants dans l'ordre que vous avez indiqué dans la recette. Cependant, les composants de renforcement CIS sont toujours exécutés en dernier pour garantir que les tests de référence s'exécutent par rapport à votre image de sortie.

- componentArn (chaîne, obligatoire) — L'ARN du composant.

  > **Tip**
  >
  > Pour utiliser l'exemple afin de créer votre propre recette de contenant, remplacez les exemples d'ARN par les ARN des composants que vous utilisez pour votre recette,. Il s'agit Région AWS notamment du nom et du numéro de version de chacun.

- paramètres (tableau d'objets) — Contient un tableau d'`ComponentParameter`objets. Si un paramètre d'entrée est obligatoire, mais qu'aucune valeur par défaut n'est définie dans le composant, vous devez fournir une valeur. Image Builder ne créera pas la version de la recette s'il manque des paramètres requis et si aucune valeur par défaut n'est définie.

  > **Important**
  >
  > Les paramètres des composants sont des valeurs en texte brut et sont connectés AWS CloudTrail. Nous vous recommandons d'utiliser AWS Secrets Manager ou le AWS Systems Manager Parameter Store pour stocker vos secrets. Pour plus d'informations sur Secrets Manager, voir Qu'est-ce que Secrets Manager ? dans le Guide de AWS Secrets Manager l'utilisateur. Pour plus d'informations sur le stockage des AWS Systems Manager paramètres, consultez la section AWS Systems Manager Parameter Store dans le Guide de AWS Systems Manager l'utilisateur.

  - name (chaîne, obligatoire) — Nom du paramètre de composant à définir.

    - valeur (tableau de chaînes, obligatoire) — Contient un tableau de chaînes pour définir la valeur du paramètre de composant nommé. Si une valeur par défaut est définie pour le composant et qu'aucune autre valeur n'est fournie, AWSTOE utilise la valeur par défaut.
- ContainerType (chaîne, obligatoire) — Type de conteneur à créer. Les valeurs valides incluentDOCKER.
- dockerfile TemplateData (string) — Le modèle Dockerfile en ligne pour créer votre image. Les données en ligne sont limitées à 16 000 caractères. Vous devez spécifier `dockerfileTemplateData` ou `dockerfileTemplateUri`, mais pas les deux.

  Votre modèle Dockerfile peut inclure des variables contextuelles (espaces réservés qu'Image Builder remplace par des informations de génération lors de l'exécution). Incluez au moins la `parentImage` variable. Si votre recette comprend des composants, incluez également la `environments` variable et la `components` variable.

  Image du parent (obligatoire)

  Au moment de la création, cette variable correspond à l'image de base de votre recette.

  Exemple :

  ```
  FROM
  {{{ imagebuilder:parentImage }}}
  ```

  environnements (obligatoire si des composants sont spécifiés)

  Au moment de la génération, cette variable est résolue en fonction des directives Dockerfile qui placent vos composants dans le contexte de génération. Ces directives copient les scripts des composants dans l'image et définissent les variables d'environnement que ces scripts utilisent. Il prépare l'environnement mais n'exécute pas les composants que vous spécifiez. Placez cette variable avant la `components` variable.

  Exemple :

  ```
  {{{ imagebuilder:environments }}}
  ```

  composants (en option)

  Au moment de la génération, cette variable correspond à la commande qui exécute les scripts de génération et de test des composants pour les composants inclus dans la recette

du conteneur, puis supprime ces scripts de l'image. Il s'agit de la variable qui exécute vos composants. Placez-le après la `environments` variable.

Exemple :

```
{{{ imagebuilder:components }}}
```

Pour stocker le modèle dans Amazon S3, utilisez plutôt le TemplateUri paramètre dockerfile avec un URI Amazon S3 valide (par exemple,`s3://my-bucket/templates/Dockerfile`).

- name (chaîne, obligatoire) — Le nom de la recette du contenant.
- description (chaîne) — Description de la recette du contenant.
- parentImage (chaîne, obligatoire) — L'image du Docker conteneur à utiliser dans la recette du conteneur comme référence pour votre image personnalisée.
  - Images publiques hébergées sur DockerHub
  - Images de conteneurs existantes dans Amazon ECR
  - Amazon-managed images de conteneurs
- PlatformOverride (chaîne) — Spécifie la plate-forme du système d'exploitation lorsque vous utilisez une image de base personnalisée. Nous vous recommandons de définir cette valeur lorsque vous utilisez une image Docker Hub ou une image Amazon ECR. Si vous l'omettez, Image Builder tente de détecter la plate-forme à partir du manifeste du conteneur, qui n'est peut-être pas toujours disponible. Vous ne pouvez pas spécifier `platformOverride` quand votre image parente est un ARN d'image Image Builder. Les valeurs valides sont `Linux` et `Windows`. Les recettes de conteneurs ne sont pas compatibles avec la plate-forme macOS.
- <major>SemanticVersion (chaîne, obligatoire) — La version sémantique de la recette de conteneur spécifiée dans le format suivant, avec des valeurs numériques à chaque position pour indiquer une version spécifique :. <minor>. <patch>. Un exemple serait `1.0.0`. Pour en savoir plus sur la gestion sémantique des versions pour les ressources Image Builder, consultez. Versionnage sémantique dans Image Builder
- tags (string map) — Tags attachés à la recette du contenant.
- InstanceConfiguration (objet) : groupe d'options qui peuvent être utilisées pour configurer une instance afin de créer et de tester des images de conteneurs.
  - image (string) — L'image de base d'une instance de création et de test de conteneur. Il peut contenir un ID AMI ou spécifier un paramètre AWS Systems Manager (SSM) Parameter Store, préfixé par`ssm:`, suivi du nom du paramètre ou de l'ARN. Si vous utilisez un paramètre SSM, la valeur du paramètre doit contenir un ID AMI. Si vous ne spécifiez

pas d'image de base, Image Builder utilise l'AMI optimisée Amazon ECS appropriée comme image de base.

- block DeviceMappings (tableau d'objets) : définit les périphériques de type bloc à associer pour créer une instance à partir de l'AMI Image Builder spécifiée dans le image paramètre.
  - DeviceName (chaîne) — L'appareil auquel ces mappages s'appliquent.
  - ebs (object) — Utilisé pour gérer la configuration spécifique d'Amazon EBS pour ce mappage.
    - delete OnTermination (booléen) — Utilisé pour configurer la suppression lors de la fermeture de l'appareil associé.
    - chiffré (booléen) — Utilisé pour configurer le chiffrement de l'appareil.
    - VolumeSize (entier) — Utilisé pour modifier la taille du volume de l'appareil.
    - VolumeType (chaîne) — Utilisé pour remplacer le type de volume de l'appareil.
- TargetRepository (object, obligatoire) — Le référentiel de destination pour l'image du conteneur s'il n'y a aucun autre référentiel spécifié dans la configuration de distribution de votre pipeline pour la région où le pipeline s'exécute (Région 1).
  - repositoryName (chaîne, obligatoire) — Nom du référentiel de conteneurs dans lequel l'image de conteneur en sortie est stockée. Ce nom est doté du préfixe de l'emplacement du référentiel.
  - service (chaîne, obligatoire) — Spécifie le service de référentiel qui stocke l'image de sortie. La valeur valide est ECR. Le référentiel cible doit déjà exister.
- WorkingDirectory (chaîne) — Le répertoire de travail à utiliser lors des flux de travail de génération et de test.

L'exemple suivant montre un fichier d'entrée de recette de conteneur complet qui définit un Docker conteneur Linux avec des composants de construction. Enregistrez-le sous `create-container-recipe.json` et utilisez-le dans la create-container-recipe commande.

```
{
 "components": [
   {
   "componentArn": "arn:aws:imagebuilder:us-west-2:111122223333:component/
helloworldal2/x.x.x"
   }
 ],
 "containerType": "DOCKER",
```

```
 "description": "My Linux Docker container image",
 "dockerfileTemplateData": "FROM
 {{{ imagebuilder:parentImage }}}\n{{{ imagebuilder:environments }}}\n{{{ imagebuilder:comp
 "name": "amazonlinux-container-recipe",
 "parentImage": "amazonlinux:latest",
 "platformOverride": "Linux",
 "semanticVersion": "1.0.2",
 "tags": {
   "sometag" : "Tag detail"
 },
 "instanceConfiguration": {
   "image": "ami-1234567890abcdef1",
   "blockDeviceMappings": [
   {
   "deviceName": "/dev/xvda",
   "ebs": {
    "deleteOnTermination": true,
    "encrypted": false,
    "volumeSize": 8,
    "volumeType": "gp2"
    }
    }
   ]
 },
 "targetRepository": {
   "repositoryName": "myrepo",
   "service": "ECR"
 },
 "workingDirectory": "/tmp"
}
```

2. Créez la recette

   Utilisez la commande suivante pour créer la recette. Entrez le nom du fichier JSON que vous avez créé à l'étape précédente dans le `--cli-input-json` paramètre :

   ```
   aws imagebuilder create-container-recipe --cli-input-json file://create-container-recipe.json
   ```

   > **Note**
   >
   > - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.

- Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

L'exemple de sortie suivant montre l'ARN de la recette de conteneur nouvellement créée, ainsi que l'ID de demande et le jeton client pour la demande.

```
{
 "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-container-recipe/1.0.2"
}
```

Contraintes relatives aux recettes des contenants

Passez en revue les contraintes suivantes avant de créer une recette de contenant :

- Vous devez spécifier un référentiel Amazon ECR cible qui existe déjà.
- AWS Marketplace les composants ne sont pas pris en charge dans les recettes de conteneurs.
- Les composants sont facultatifs. Vous pouvez personnaliser l'image de sortie avec les composants BUILD, avec des instructions dans votre modèle Dockerfile (par exemple, `RUN` ou `COPY` des instructions), ou les deux. Les composants TEST valident l'image mais ne la modifient pas.
- Les recettes de conteneurs ne sont pas compatibles avec la plate-forme macOS.
- Pour le modèle Dockerfile, fournissez soit des données en ligne (16 000 caractères maximum), soit un URI Amazon S3. Vous ne pouvez pas spécifier les deux.
- La taille cumulée de la recette ne peut pas dépasser 25 Ko. Si vous recevez une erreur de taille, réduisez le nombre de paramètres des composants ou réduisez les valeurs des paramètres. Cette limite est distincte des limites de taille du modèle Dockerfile.
- Lorsque vous définissez `platformOverride` sur`Windows`, vous devez également spécifier`imageOsVersionOverride`. Cela s'applique à toute image parent autre qu'Image Builder, telle qu'une image Amazon ECR ou une Docker Hub image.

Erreurs courantes et résolution des problèmes

Le tableau suivant répertorie les erreurs courantes que vous pouvez rencontrer lors de la création d'une recette de contenant, ainsi que la manière de les résoudre.

Erreur	Cause	Résolution
« Les recettes de contenants ne doivent pas contenir de composants du marché »	Un AWS Marketplace composant d'une recette en contenant.	N'utilisez que des AWS Marketplace composants non constitutifs.
« Les recettes de conteneurs ne sont actuellement pas compatibles avec la plateforme macOS »	`platform0verride` est réglé sur macOS.	Utiliser `Linux` ou `Windows`.
« Vous devez spécifier soit un dockerfile, TemplateData soit un dockerfile » TemplateUri	Vous avez spécifié les deux ou aucun des deux.	Fournissez-en exactement un.
« URI S3 non valide »	Un chemin Amazon S3 mal formé dans. `dockerfileTemplateUri`	Utilisez le format `s3://`*`bucket`*`/`*`key`*.
« Vous ne pouvez pas spécifier de remplacement de plate-forme lorsque vous utilisez une image Image Builder comme parent »	`platform0verride` avec un ARN d'image Image Builder.	Supprimez `platform0verride` . Image Builder détecte la plateforme à partir des métadonnées de l'image parente.
« Vous devez spécifier un remplacement d'image OsVersion lorsque vous utilisez les référentiels ECR comme image parente pour la plate-forme Windows »	Une version Windows `platform0verride` sans système d'exploitation.	Ajoutez le `image0sVersion0verride` champ.

Suivez la lignée des AMI à l'aide de filigranes

Les filigranes AMI sont de simples étiquettes de texte qui retracent l'historique des images de votre machine. Vous ajoutez des noms de filigranes à votre recette d'image, et Image Builder les attache à l'AMI de sortie pendant le processus de création.

Lorsque vous copiez ou distribuez une AMI vers d'autres régions ou comptes, les filigranes sont automatiquement copiés. Toutes les copies de votre AMI portent les mêmes filigranes. Cela vous permet de retracer l'historique de vos images.

Vous ne pouvez pas rendre publiques les AMI filigranées. Si votre recette inclut des filigranes ou si votre AMI source contient des filigranes, Image Builder rejette les configurations de distribution qui définissent les autorisations de lancement comme publiques.

Principales caractéristiques des filigranes AMI :

- Objectif — Suivez le lignage des AMI à travers les copies et les distributions.
- Champ d'application : métadonnées uniquement. Les filigranes n'affectent pas les performances ou le comportement d'exécution de l'instance.
- Restriction de visibilité — Les AMI filigranées ne peuvent pas être rendues publiques.

Table des matières

- Contraintes relatives aux filigranes
- Conditions préalables
- Ajouter des filigranes à une recette illustrée
- Afficher les filigranes d'une recette existante
- Considérations

Contraintes relatives aux filigranes

Les contraintes suivantes s'appliquent aux filigranes AMI dans les recettes d'images Image Builder.

Contrainte	Value
Filigranes par recette	1 à 5
Longueur du nom	3 à 128 caractères

Contrainte	Value
Caractères autorisés	Lettres, chiffres, espaces et caractères spéciaux suivants : `( ) [ ] . / - ' @ _`
Personnages Edge	Les noms ne peuvent pas commencer ou se terminer par un espace
Sensibilité à la casse	Les noms des filigranes distinguent les majuscules et minuscules
Duplicates (doublons)	Les noms dupliqués dans la même recette ne sont pas autorisés
Total par AMI de sortie	Les filigranes de l'AMI source et les filigranes de la recette ne peuvent pas dépasser 5

Conditions préalables

Avant d'utiliser les filigranes AMI, vérifiez les exigences suivantes.

- Les filigranes AMI ne sont pris en charge que pour les recettes d'images, et non pour les recettes de conteneurs.
- Votre rôle d'exécution d'Image Builder doit inclure l'`ec2:AttachImageWatermark`autorisation sur`arn:aws:ec2:*::image/*`. Si vous utilisez le rôle lié au service Image Builder, cette autorisation est automatiquement incluse.

Ajouter des filigranes à une recette illustrée

Vous pouvez ajouter des filigranes lorsque vous créez une recette illustrée ou une nouvelle version d'une recette existante. Choisissez l'un des onglets suivants pour obtenir des instructions.

Console

Pour ajouter des filigranes à une recette illustrée

1. Lorsque vous créez ou modifiez une recette d'image, dans la section de configuration de la recette, développez les filigranes AMI.

2. Entrez un nom en filigrane (3 à 128 caractères). Choisissez Ajouter un filigrane pour ajouter d'autres noms (jusqu'à 5 au total).
3. Effectuez les étapes de création de recettes restantes, puis choisissez Créer une recette.

AWS CLI

Pour ajouter des filigranes avec le AWS CLI, incluez le `amiWatermarks` paramètre dans votre fichier d'entrée JSON pour la create-image-recipe commande.

L'exemple suivant montre un fichier JSON de recette avec des filigranes configurés.

```
{
    "name": "MyWatermarkedRecipe",
    "description": "Recipe with AMI watermarks for lineage tracking.",
    "parentImage": "ami-1234567890abcdef1",
    "semanticVersion": "1.0.0",
    "components": [
        {
            "componentArn": "arn:aws:imagebuilder:us-
west-2:111122223333:component/my-component/1.0.0/1"
        }
    ],
    "amiWatermarks": [
        "ProductionV1",
        "team-alpha"
    ]
}
```

Exécutez la commande suivante pour créer la recette.

```
aws imagebuilder create-image-recipe --cli-input-json file://create-watermarked-
recipe.json
```

La liste suivante décrit le `amiWatermarks` paramètre.

- AMIWatermarks (tableau de chaînes, facultatif) : liste de noms de filigranes à associer aux AMI de sortie créées à partir de cette recette. Spécifiez 1 à 5 noms. Chaque nom doit comporter de 3 à 128 caractères et peut inclure des lettres, des chiffres, des espaces et les caractères spéciaux suivants : `( ) [ ] . / - ' @ _`

CloudFormation

Vous pouvez spécifier des filigranes AMI dans une `AWS::ImageBuilder::ImageRecipe` ressource à l'aide de la `AmiWatermarks` propriété.

L'exemple suivant montre un modèle de recette d'image minimal avec des filigranes configurés.

```
Resources:
  WatermarkedRecipe:
    Type: AWS::ImageBuilder::ImageRecipe
    Properties:
      Name: MyWatermarkedRecipe
      Version: 1.0.0
      ParentImage: arn:aws:imagebuilder:us-west-2:aws:image/amazon-linux-2023-x86/
x.x.x
      Components:
        - ComponentArn: arn:aws:imagebuilder:us-west-2:111122223333:component/my-
component/1.0.0/1
      AmiWatermarks:
        - ProductionV1
        - team-alpha
```

Vous ne pouvez définir la `AmiWatermarks` propriété qu'au moment de sa création. Les recettes d'images étant immuables, la modification des filigranes nécessite une nouvelle version de recette.

Pour plus d'informations sur la `AmiWatermarks` propriété, voir AWS::ImageBuilder: : ImageRecipe dans la référence du AWS CloudFormation modèle.

Afficher les filigranes d'une recette existante

Vous pouvez afficher les filigranes configurés pour une recette en appelant la get-image-recipe commande dans le AWS CLI.

```
aws imagebuilder get-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:111122223333:image-recipe/my-watermarked-recipe/1.0.0
```

La réponse inclut le `amiWatermarks` tableau si des filigranes ont été configurés pour cette version de recette.

Considérations

Gardez les points suivants à l'esprit lorsque vous utilisez des filigranes AMI.

- Les filigranes d'une AMI source et les filigranes définis dans la recette se combinent sur l'AMI de sortie. Le total ne peut pas dépasser 5. Si le nombre combiné est supérieur à 5, le build échoue.
- Si vous distribuez une AMI filigranée à d'autres régions ou comptes, les filigranes se propagent automatiquement.
- Image Builder vérifie que les AMI filigranées ne sont pas définies comme publiques au moment de la création de l'image. Si votre recette comporte des filigranes (ou si l'AMI source contient des filigranes) et que la configuration de distribution définit les autorisations de lancement comme publiques, Image Builder renvoie une `InvalidParameterCombinationException` erreur.

nettoyer des ressources ;

Pour éviter des frais imprévus, veillez à nettoyer les ressources et les pipelines que vous avez créés à partir des exemples de ce guide. Pour plus d'informations sur la suppression de ressources dans Image Builder, consultezSupprimer les ressources d'Image Builder obsolètes ou inutilisées.

Gérer la configuration de l'infrastructure d'Image Builder

Une configuration d'infrastructure est une ressource Image Builder réutilisable qui définit l'environnement Amazon EC2 dans lequel Image Builder crée et teste votre image. Lorsque vous créez un pipeline d'images, vous y associez une configuration d'infrastructure. Image Builder utilise ensuite ces paramètres chaque fois que le pipeline exécute une génération.

La configuration de l'infrastructure étant une ressource distincte, vous pouvez réutiliser le même environnement de génération sur de nombreux pipelines. Vous pouvez également mettre à jour l'environnement en un seul endroit sans modifier les pipelines individuels. Lorsque vous mettez à jour une configuration d'infrastructure, vos modifications prennent effet lors de la prochaine version. Vos modifications n'affectent pas les constructions déjà en cours ni les images déjà créées par Image Builder.

> **Note**
>
> La configuration d'une infrastructure est indépendante des images qu'elle permet de créer. La suppression ou la modification d'une configuration d'infrastructure ne modifie pas les images déjà créées par Image Builder. Si un pipeline fait référence à une configuration

d'infrastructure, vous ne pouvez pas supprimer cette configuration tant que vous n'avez pas supprimé la référence.

Utilisez une configuration d'infrastructure pour spécifier les paramètres suivants pour votre environnement de génération et de test. Pour les noms des champs d'API, les paramètres requis et les contraintes, consultez CreateInfrastructureConfiguration la référence de l'API EC2 Image Builder.

- Types d'instances pour vos instances de génération et de test. Nous vous recommandons de spécifier plusieurs types d'instance, car cela permet à Image Builder de lancer une instance à partir d'un pool doté d'une capacité suffisante. Cela peut réduire vos échecs de construction transitoires. Si vous spécifiez plusieurs types, Image Builder lance une instance à partir du premier type disposant de la capacité disponible. Si aucun de vos types d'instance n'est disponible dans le, la compilation échoue.

  Pour les images Mac, choisissez des types d'instances compatibles nativement avec macOS. Pour plus d'informations, consultez la section Instances Amazon EC2 Mac dans le guide de l'utilisateur Amazon EC2.
- Un profil d'instance qui donne à vos instances de génération et de test les autorisations nécessaires pour exécuter vos composants. Par exemple, si vous avez un composant qui extrait des ressources d'Amazon S3, le profil d'instance nécessite des autorisations pour accéder à ces fichiers. Le profil d'instance nécessite également un ensemble minimal d'autorisations pour qu'EC2 Image Builder puisse communiquer correctement avec l'instance. Pour de plus amples informations, veuillez consulter Préparez-vous à créer des images personnalisées avec Image Builder.
- Paramètres de placement des instances qui contrôlent la zone de disponibilité, la location et l'hôte dédié sur lequel Image Builder lance vos instances.
- Réseau : sous-réseau VPC et groupes de sécurité pour vos instances de génération et de test. Si vous spécifiez un sous-réseau, vous devez également spécifier au moins un groupe de sécurité.
- Journalisation : emplacement Amazon S3 où Image Builder stocke les journaux des AWSTOE applications relatifs à vos phases de création et de test. Pour utiliser la journalisation Amazon S3, le profil d'instance doit disposer `s3:PutObject` d'autorisations pour le compartiment cible (`arn:aws:s3:::BucketName/*`). Si vous ne configurez pas la journalisation Amazon S3, Image Builder conserve les journaux de l' AWSTOE application uniquement sur les instances de génération et de test.

- Paramètres de résolution des problèmes : une paire de clés Amazon EC2 et le `terminateInstanceOnFailure` paramètre. Grâce à ces paramètres, vous pouvez continuer à exécuter une instance de génération défaillante et vous y connecter pour effectuer une enquête.
- Options de métadonnées d'instance qui contrôlent si les instances nécessitent des jetons IMDSv2 et la limite de sauts de réponse des métadonnées.
- Rubrique SNS dans laquelle Image Builder publie des messages concernant l'état de création de votre image, par exemple lorsqu'une image devient disponible ou qu'une génération échoue. Pour de plus amples informations, veuillez consulter Intégration d'Amazon SNS dans Image Builder.

> **Note**
>
> Si votre rubrique SNS est chiffrée, la clé qui chiffre cette rubrique doit résider dans le compte sur lequel s'exécute le service Image Builder. Image Builder ne peut pas envoyer de notifications à des sujets SNS chiffrés à l'aide de clés provenant d'autres comptes.

Table des matières

- Placement et location d'instances
- Répertorier et afficher les détails de configuration de l'infrastructure
- Créer une configuration d'infrastructure
- Mettre à jour une configuration d'infrastructure
- Supprime une configuration d'infrastructure
- Image Builder et AWS PrivateLink Points de terminaison de VPC d'interface

> **Tip**
>
> Lorsque vous disposez de plusieurs ressources du même type, le balisage vous permet d'identifier une ressource spécifique en fonction des balises que vous lui avez attribuées. Pour plus d'informations sur le balisage de vos ressources à l'aide des commandes d'Image Builder AWS CLI, consultez la Balisage des ressources section de ce guide.

Placement et location d'instances

Utilisez les paramètres de placement pour contrôler l'endroit où Image Builder lance vos instances de génération et de test. Par défaut, les instances Amazon EC2 s'exécutent sur du matériel de location partagé, ce qui signifie que plusieurs instances Comptes AWS peuvent partager le même serveur physique. Vous pouvez modifier la location pour qu'elle fonctionne sur du matériel à locataire unique ou sur un hôte dédié. Vous pouvez également épingler des instances dans une zone de disponibilité spécifique.

Pour les noms des champs de placement, leurs valeurs valides et leurs contraintes, voir Placement dans la référence d'API EC2 Image Builder.

> **Note**
>
> Les instances Mac nécessitent un Hôte dédié. Réglez `tenancy` sur `host` pour les images macOS. Si le placement automatique est activé pour votre hôte dédié et que vous ne spécifiez pas de `hostId` ou`hostResourceGroupArn`, Amazon EC2 trouve un hôte disponible pour vous. Pour plus d'informations, consultez Auto-placement le guide de l'utilisateur Amazon EC2.

Répertorier et afficher les détails de configuration de l'infrastructure

Cette section décrit les différentes manières de trouver des informations et de visualiser les détails de vos configurations d'infrastructure EC2 Image Builder.

Détails de configuration de l'infrastructure

- Liste des configurations d'infrastructure
- Obtenir des informations sur la configuration de l'infrastructure

Liste des configurations d'infrastructure

Console

Pour consulter la liste des configurations d'infrastructure créées sous votre compte dans la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez Configuration de l'infrastructure dans le volet de navigation. Cela affiche la liste des configurations d'infrastructure créées sous votre compte.

AWS CLI

L'exemple suivant montre comment répertorier toutes les configurations de votre infrastructure à l'aide de la list-infrastructure-configurations commande du AWS CLI.

```
aws imagebuilder list-infrastructure-configurations
```

Pour affiner les résultats, utilisez l'`--filters`option. Vous pouvez filtrer par`name`. L'exemple suivant renvoie des configurations d'infrastructure dont le nom correspond à la valeur que vous spécifiez.

```
aws imagebuilder list-infrastructure-configurations --filters name=name,values=my-example-infrastructure-configuration
```

Si vous avez de nombreuses configurations d'infrastructure, Image Builder renvoie les résultats page par page. Permet `--max-results` de contrôler la taille de la page. Si la réponse inclut une `nextToken` valeur, transmettez-la `--next-token` pour récupérer la page suivante.

```
aws imagebuilder list-infrastructure-configurations --max-results 10
```

Obtenir des informations sur la configuration de l'infrastructure

Console

Pour afficher les détails d'une configuration d'infrastructure spécifique dans la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez Configuration de l'infrastructure dans le volet de navigation.

3. Cliquez sur le lien Nom de la configuration pour la configuration d'infrastructure que vous souhaitez afficher. Cela ouvre la page détaillée, dans laquelle vous pouvez consulter ses paramètres.

AWS CLI

L'exemple suivant montre comment utiliser la get-infrastructure-configuration commande du AWS CLI pour obtenir les détails d'une configuration d'infrastructure en spécifiant son Amazon Resource Name (ARN).

```
aws imagebuilder get-infrastructure-configuration --infrastructure-configuration-
arn arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-
example-infrastructure-configuration
```

Créer une configuration d'infrastructure

Cette section décrit comment utiliser la console ou les imagebuilder commandes d'Image Builder AWS CLI pour créer une configuration d'infrastructure.

Console

Pour créer une ressource de configuration d'infrastructure à partir de la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Dans le volet de navigation, choisissez Configuration de l'infrastructure.
3. Choisissez Créer une configuration d'infrastructure.
4. Dans la section Général, entrez les informations requises suivantes :
   - Entrez le nom de votre ressource de configuration d'infrastructure.
   - Sélectionnez le rôle IAM que vous souhaitez associer au profil d'instance pour les autorisations des composants sur vos instances de génération et de test. Image Builder utilise ces autorisations pour télécharger et exécuter vos composants, charger les journaux et effectuer toutes les actions supplémentaires spécifiées par les composants de votre recette. CloudWatch

5. Dans le panneau AWS Infrastructure, configurez les autres paramètres d'infrastructure. Saisissez les informations requises suivantes :

   - Type d'instance : vous pouvez spécifier un ou plusieurs types d'instance à utiliser pour cette version. Image Builder sélectionne l'un de ces types d'instance en fonction de sa disponibilité.

     > **Note**
     >
     > Les instances Mac s'exécutent sur des types d'`.metal`instances sur un hôte dédié. Votre type d'instance doit correspondre à l'un des types définis pour l'hôte sur lequel elle s'exécute. Pour plus d'informations sur les instances Mac et une liste des types d'instances qui prennent en charge nativement le système d'exploitation macOS, consultez les instances Mac Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

   - Rubrique SNS (facultatif) : sélectionnez une rubrique SNS pour recevoir des notifications et des alertes de la part d'EC2 Image Builder.

   Si vous ne fournissez pas de valeurs pour les paramètres suivants, ils utilisent les valeurs par défaut spécifiques au service, le cas échéant.

   - VPC, sous-réseau et groupes de sécurité : Image Builder utilise votre VPC et votre sous-réseau par défaut. Pour plus d'informations sur la configuration des points de terminaison de l'interface VPC, consultez. Image Builder et AWS PrivateLink Points de terminaison de VPC d'interface
   - Dans la section Paramètres de résolution des problèmes, configurez les valeurs suivantes :
     - Par défaut, la case à cocher Terminer l'instance en cas d'échec est sélectionnée. Toutefois, lorsqu'une compilation échoue, vous pouvez vous connecter à l'instance Amazon EC2 pour résoudre les problèmes. Si vous souhaitez que votre instance continue de fonctionner après un échec de compilation, désactivez la case à cocher.
     - Paire de clés : si votre instance Amazon EC2 continue de fonctionner après un échec de génération, vous pouvez créer une paire de clés ou utiliser une paire de clés existante pour vous connecter à l'instance et résoudre les problèmes.

- Journaux : vous pouvez spécifier un compartiment Amazon S3 dans lequel Image Builder peut écrire des journaux d'application pour vous aider à résoudre les problèmes liés à votre build et à vos tests. Si vous ne spécifiez pas de compartiment Amazon S3, Image Builder écrit les journaux de l'application dans l'instance.

- Dans la section Paramètres des métadonnées de l'instance, configurez les valeurs suivantes à appliquer aux instances Amazon EC2 qu'Image Builder utilise pour créer et tester votre image :
  - Sélectionnez la version des métadonnées pour déterminer si Amazon EC2 nécessite un en-tête de jeton signé pour les demandes de récupération de métadonnées d'instance.
    - V1 et V2 (jeton facultatif) : valeur par défaut si vous ne sélectionnez rien.
    - V2 (jeton requis)

    > **Note**
    >
    > Nous vous recommandons de configurer toutes les instances EC2 qu'Image Builder lance à partir d'une génération de pipeline pour utiliser IMDSv2 afin que les demandes de récupération des métadonnées d'instance nécessitent un en-tête de jeton signé.

  - Limite de sauts de réponse du jeton de métadonnées : nombre de sauts réseau que le jeton de métadonnées peut parcourir. Plage valide : 1 à 64. Si vous ne définissez aucune valeur, Image Builder applique une limite de sauts de 2.

- Dans la section Paramètres de placement des instances, configurez les valeurs suivantes à appliquer aux instances Amazon EC2 qu'Image Builder utilise pour créer et tester votre image :
  - Vous pouvez sélectionner la zone de disponibilité dans laquelle Image Builder lance les instances lors de la création de l'image.
  - Vous pouvez également sélectionner Tenancy pour les serveurs qui exécutent les instances que vous lancez. Par défaut, les instances Amazon EC2 s'exécutent sur du matériel de location partagée. Cela signifie que plusieurs Comptes AWS personnes peuvent partager le même matériel physique. Une instance avec `dedicated` location s'exécute sur du matériel à locataire unique. Une instance avec `host` location s'exécute sur un hôte dédié.

    Les instances Mac nécessitent la création d'un hôte dédié comme condition préalable à la création d'une image personnalisée. Sélectionnez `host` pour votre image macOS.

Vous pouvez ensuite sélectionner un hôte cible ou un groupe de ressources d'hôtes pour lancer des instances, mais cela n'est pas obligatoire si le placement automatique est activé sur votre hôte dédié. Pour plus d'informations, consultez Auto-placement le guide de l'utilisateur Amazon EC2.

- ID d'hôte de location : ID de l'hôte dédié sur lequel les instances s'exécutent.
- Groupe de ressources hôte Tenancy : nom de ressource Amazon (ARN) du groupe de ressources hôte dans lequel lancer les instances.

6. Dans la section Balises d'infrastructure (facultatif), vous pouvez attribuer des balises de métadonnées à l'instance Amazon EC2 qu'Image Builder lance pendant le processus de création. Entrez les balises sous forme de paires clé-valeur.
7. Dans la section Tags (facultatif), vous pouvez attribuer des balises de métadonnées à la ressource de configuration d'infrastructure créée par Image Builder en sortie. Entrez les balises sous forme de paires clé-valeur.

AWS CLI

La procédure suivante montre comment configurer l'infrastructure de votre image à l'aide de la create-infrastructure-configuration commande Image Builder du AWS CLI. La commande de l'étape 2 intègre le fichier que vous avez créé à l'étape 1. Pour ces exemples, le fichier de l'étape 1 est appelé`create-infrastructure-configuration.json`.

L'exemple suivant définit les champs d'infrastructure courants. Pour exiger des jetons IMDSv2 sur les instances de build et de test, ajoutez un `instanceMetadataOptions` bloc avec `httpTokens` set to. `required`

> **Note**
>
> Nous vous recommandons de configurer toutes les instances EC2 qu'Image Builder lance à partir d'une génération de pipeline pour utiliser IMDSv2 afin que les demandes de récupération des métadonnées d'instance nécessitent un en-tête de jeton signé.

1. Créer un fichier JSON d'entrée CLI

   Les exemples suivants présentent des variantes du fichier JSON que vous pouvez créer pour la configuration de l'infrastructure. Utilisez un outil d'édition de fichiers pour créer votre propre fichier JSON.

Exemple 1 : Configuration pour conserver une instance d'une compilation ayant échoué

Cet exemple spécifie deux types d'instances, `m7i.large` et`m7i.xlarge`. Nous vous recommandons de spécifier plusieurs types d'instance, car cela permet à Image Builder de lancer une instance à partir d'un pool doté d'une capacité suffisante. Cela peut réduire vos échecs de construction transitoires.

`instanceProfileName`Spécifie le profil d'instance qui fournit à l'instance les autorisations dont elle a besoin pour effectuer des activités de personnalisation. Par exemple, si vous avez un composant qui extrait des ressources d'Amazon S3, le profil d'instance nécessite des autorisations pour accéder à ces fichiers. Le profil d'instance nécessite également un ensemble minimal d'autorisations pour qu'EC2 Image Builder puisse communiquer correctement avec l'instance. Pour de plus amples informations, veuillez consulter Préparez-vous à créer des images personnalisées avec Image Builder.

```
{
    "name": "ExampleInfraConfigDontTerminate",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
        "m7i.large", "m7i.xlarge"
    ],
    "instanceProfileName": "myIAMInstanceProfileName",
    "securityGroupIds": [
        "sg-12345678"
    ],
    "subnetId": "subnet-12345678",
    "logging": {
        "s3Logs": {
            "s3BucketName": "my-logging-bucket",
            "s3KeyPrefix": "my-path"
        }
    },
    "keyPair": "myKeyPairName",
    "terminateInstanceOnFailure": false,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

Exemple 2 : configuration macOS avec placement automatique

Cet exemple spécifie les types d'instances et leur emplacement pour une instance Mac où le placement automatique est activé sur l'hôte dédié.

```
{
   "name": "macOSInfraConfigAutoPlacement",
   "description": "An example infrastructure configuration for macOS.",
   "instanceProfileName": "EC2InstanceProfileForImageBuilder",
   "instanceTypes": ["mac1.metal", "mac2.metal"],
   "terminateInstanceOnFailure": false,
   "placement": {
      "tenancy": "host"
   }
}
```

Exemple 3 : configuration macOS avec l'ID d'hôte spécifié

Cet exemple spécifie le type d'instance et l'emplacement d'une instance Mac qui cible un hôte dédié spécifique.

```
{
   "name": "macOSInfraConfigHostPlacement",
   "description": "An example infrastructure configuration for macOS.",
   "instanceProfileName": "EC2InstanceProfileForImageBuilder",
   "instanceTypes": ["mac2-m1ultra.metal"],
   "terminateInstanceOnFailure": false,
   "placement": {
      "tenancy": "host",
      "hostId" : "h-1234567890abcdef0"
   }
}
```

Exemple 4 : VPC personnalisé avec IMDSv2 et journalisation Amazon S3

Cet exemple lance des instances dans un sous-réseau et un groupe de sécurité spécifiques, nécessite des jetons IMDSv2 et écrit les journaux des applications sur Amazon S3. Parce que l'exemple définit`subnetId`, il doit également définir`securityGroupIds`.

```
{
    "name": "VpcInfraConfigWithLogging",
    "description": "Build in a custom VPC with IMDSv2 and S3 logging",
```

```
    "instanceTypes": ["m7i.large", "m8i.large"],
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "subnetId": "subnet-12345678",
    "securityGroupIds": ["sg-12345678"],
    "instanceMetadataOptions": {
        "httpTokens": "required",
        "httpPutResponseHopLimit": 2
    },
    "logging": {
        "s3Logs": {
            "s3BucketName": "my-logging-bucket",
            "s3KeyPrefix": "image-builder-logs"
        }
    },
    "resourceTags": {
        "Project": "golden-ami"
    },
    "terminateInstanceOnFailure": true
}
```

Exemple 5 : placement d'un hôte dédié auprès d'un groupe de ressources d'hôtes

Cet exemple cible un groupe de ressources hôtes au lieu d'un ID d'hôte spécifique. Parce `tenancy` que oui`host`, vous pouvez spécifier l'ARN d'un groupe de ressources hôtes. Ne spécifiez pas également un`hostId`.

```
{
    "name": "HostResourceGroupInfraConfig",
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "instanceTypes": ["mac2.metal"],
    "terminateInstanceOnFailure": false,
    "placement": {
        "tenancy": "host",
        "hostResourceGroupArn": "arn:aws:resource-groups:us-
west-2:123456789012:group/my-host-group"
    }
}
```

2. Utilisez le fichier que vous avez créé comme entrée lorsque vous exécutez la commande suivante.

```
aws imagebuilder create-infrastructure-configuration --cli-input-json
 file://create-infrastructure-configuration.json
```

Mettre à jour une configuration d'infrastructure

Cette section explique comment utiliser la console Image Builder ou imagebuilder les commandes de la ressource de configuration AWS CLI pour mettre à jour une infrastructure. Pour suivre vos ressources, vous pouvez appliquer des balises comme suit. Entrez les balises sous forme de paires clé-valeur.

- Les balises de ressources attribuent des balises de métadonnées à l'instance Amazon EC2 qu'Image Builder lance pendant le processus de création.
- Les balises attribuent des balises de métadonnées à la ressource de configuration de l'infrastructure qu'Image Builder crée en sortie.

Console

Vous pouvez modifier les détails de configuration d'infrastructure suivants depuis la console Image Builder :

- Description de la configuration de votre infrastructure.
- Le rôle IAM à associer au profil d'instance.
- AWS infrastructure, y compris le type d'instance et une rubrique SNS pour les notifications.
- VPC, sous-réseau et groupes de sécurité.
- Paramètres de résolution des problèmes, notamment l'arrêt de l'instance en cas de panne, la paire de clés pour la connexion et l'emplacement facultatif du compartiment S3 pour les journaux d'instance.

Pour mettre à jour une ressource de configuration d'infrastructure depuis la console Image Builder, procédez comme suit :

Choisissez une configuration d'infrastructure Image Builder existante

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Pour consulter la liste des ressources de configuration de l'infrastructure associées à votre compte, choisissez Configuration de l'infrastructure dans le volet de navigation.
3. Pour afficher les détails ou modifier une configuration d'infrastructure, cliquez sur le lien Nom de la configuration. Cela ouvre la vue détaillée de la configuration de l'infrastructure.

> **Note**
>
> Vous pouvez également cocher la case à côté du nom de la configuration, puis choisir Afficher les détails.

4. Dans le coin supérieur droit du panneau Détails de l'infrastructure, choisissez Modifier.
5. Lorsque vous êtes prêt à enregistrer les mises à jour que vous avez apportées à la configuration de votre infrastructure, choisissez Enregistrer les modifications.

AWS CLI

L'exemple suivant montre comment mettre à jour la configuration de l'infrastructure de votre image à l'aide de la update-infrastructure-configuration commande Image Builder du AWS CLI.

1. Créer un fichier JSON d'entrée CLI

   Dans cet exemple, les paramètres correspondent à ceux de l'exemple de création, sauf que `terminateInstanceOnFailure` celui-ci est défini sur`true`. Après avoir exécuté la update-infrastructure-configuration commande, les pipelines qui utilisent cette configuration d'infrastructure mettent fin à la génération et testent les instances en cas d'échec d'une génération.

   L'exemple suivant définit les champs d'infrastructure courants. Pour exiger des jetons IMDSv2 sur les instances de build et de test, ajoutez un `instanceMetadataOptions` bloc avec `httpTokens` set to. `required`

> **Note**
>
> Nous vous recommandons de configurer toutes les instances EC2 qu'Image Builder lance à partir d'une génération de pipeline pour utiliser IMDSv2 afin que les demandes de récupération des métadonnées d'instance nécessitent un en-tête de jeton signé.

Utilisez un outil d'édition de fichier pour créer un fichier JSON avec les clés illustrées dans l'exemple suivant, ainsi que des valeurs valides pour votre environnement. Cet exemple utilise un fichier nomméupdate-infrastructure-configuration.json :

```
{
"infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
"description": "An example that will terminate instances of failed builds",
"instanceTypes": [
    "m7i.large", "m7i.2xlarge"
],
"instanceProfileName": "myIAMInstanceProfileName",
"securityGroupIds": [
    "sg-12345678"
],
"subnetId": "subnet-12345678",
"logging": {
    "s3Logs": {
        "s3BucketName": "my-logging-bucket",
        "s3KeyPrefix": "my-path"
    }
},
"terminateInstanceOnFailure": true,
"snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

2. Utilisez le fichier que vous avez créé comme entrée lorsque vous exécutez la commande suivante.

```
aws imagebuilder update-infrastructure-configuration --cli-input-json
 file://update-infrastructure-configuration.json
```

Supprime une configuration d'infrastructure

Vous pouvez supprimer une configuration d'infrastructure dont vous n'avez plus besoin. Vous ne pouvez pas supprimer une configuration d'infrastructure alors qu'un pipeline d'images y fait référence. Mettez d'abord à jour ou supprimez le pipeline, puis supprimez la configuration de l'infrastructure.

Console

Pour supprimer une configuration d'infrastructure depuis la console Image Builder, procédez comme suit :

Supprimer la configuration de l'infrastructure

1. Pour consulter la liste des configurations d'infrastructure créées sous votre compte, choisissez Configuration de l'infrastructure dans le volet de navigation.
2. Cochez la case à côté de Nom de la configuration pour sélectionner la configuration d'infrastructure que vous souhaitez supprimer.
3. En haut du panneau Configurations de l'infrastructure, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans la case et choisissez Supprimer.

AWS CLI

L'exemple suivant montre comment supprimer une configuration d'infrastructure à l'aide de la delete-infrastructure-configuration commande du AWS CLI.

```
aws imagebuilder delete-infrastructure-configuration --infrastructure-configuration-
arn arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-
example-infrastructure-configuration
```

Image Builder et AWS PrivateLink Points de terminaison de VPC d'interface

Vous pouvez établir une connexion privée entre votre VPC et EC2 Image Builder en créant un point de terminaison VPC d'interface. Les points de terminaison de l'interface sont alimentés par AWS PrivateLink une technologie qui vous permet d'accéder en privé aux API d'Image Builder sans passerelle Internet, périphérique NAT, connexion VPN ou Direct Connect connexion. Les instances de votre VPC n'ont pas besoin d'adresses IP publiques pour communiquer avec les API Image Builder. Le trafic entre votre VPC et Image Builder ne quitte pas le réseau Amazon.

Chaque point de terminaison d'interface est représenté par une ou plusieurs interfaces réseau Elastic dans vos sous-réseaux. Lorsque vous créez une nouvelle image, vous pouvez spécifier l'ID de sous-réseau VPC dans la configuration de votre infrastructure.

> **Note**
>
> Chaque service auquel vous accédez depuis un VPC possède son propre point de terminaison d'interface, avec sa propre politique de point de terminaison. Image Builder télécharge l'application de gestion des AWSTOE composants et accède aux ressources gérées à partir de compartiments S3 pour créer des images personnalisées. Pour autoriser l'accès à ces compartiments, vous devez mettre à jour la politique relative aux points de terminaison S3 afin de l'autoriser. Pour de plus amples informations, veuillez consulter Politiques personnalisées pour l'accès au compartiment S3.

> **Tip**
>
> Pour plus d'informations sur un exemple de build de VPC privé déployable, y compris les points de terminaison et les politiques de terminaux requis, consultez l'exemple de versions de VPC privé sur. GitHub Pour de plus amples informations, veuillez consulter Découvrez des exemples de projets Image Builder sur GitHub.

Pour plus d'informations sur les points de terminaison d'un VPC, consultez Points de terminaison de VPC d'interface (AWS PrivateLink) dans le Guide de l'utilisateur Amazon VPC.

Considérations relatives aux points de terminaison VPC d'Image Builder

Avant de configurer un point de terminaison VPC d'interface pour Image Builder, assurez-vous de consulter les propriétés et les limites du point de terminaison d'interface dans le guide de l'utilisateur Amazon VPC.

Image Builder prend en charge les appels vers toutes ses actions d'API depuis votre VPC.

Création d'un point de terminaison VPC d'interface pour Image Builder

Pour créer un point de terminaison VPC pour le service Image Builder, vous pouvez utiliser la console Amazon VPC ou le AWS Command Line Interface ()AWS CLI. Pour plus d'informations, consultez Création d'un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

Créez un point de terminaison VPC pour Image Builder en utilisant le nom de service suivant :

- com.amazonaws. *region*.créateur d'images

Si vous activez le DNS privé pour le point de terminaison, vous pouvez envoyer des requêtes d'API à Image Builder en utilisant son nom DNS par défaut pour la région, par exemple :`imagebuilder.us-east-1.amazonaws.com`. Pour rechercher le point de terminaison qui s'applique à votre région cible, consultez la section Points de terminaison et quotas d'EC2 Image Builder dans le. Référence générale d'Amazon Web Services

Pour plus d'informations, consultez Accès à un service via un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

Création d'une politique de point de terminaison VPC pour Image Builder

Vous pouvez associer une politique de point de terminaison à votre point de terminaison VPC qui contrôle l'accès à Image Builder. 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.

Si vous utilisez des Amazon-managed composants dans votre recette, le point de terminaison VPC pour Image Builder doit autoriser l'accès à la bibliothèque de composants appartenant au service suivante :

`arn:aws:imagebuilder:region:aws:component/*`

> **Important**
>
> Lorsqu'une politique autre que celle par défaut est appliquée à un point de terminaison VPC d'interface pour EC2 Image Builder, certaines demandes d'API échouées, telles que celles émanant de`RequestLimitExceeded`, peuvent ne pas être connectées ou ne pas être enregistrées sur Amazon. AWS CloudTrail CloudWatch

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.

Politiques personnalisées pour l'accès au compartiment S3

Image Builder utilise un compartiment S3 accessible au public pour stocker et accéder aux ressources gérées, telles que les composants. Il télécharge également l'application de gestion des AWSTOE composants à partir d'un compartiment S3 distinct. Si vous utilisez un point de terminaison VPC pour Amazon S3 dans votre environnement, vous devez vous assurer que votre politique de point de terminaison VPC S3 autorise Image Builder à accéder aux compartiments S3 suivants. Les noms des compartiments sont uniques par AWS région (*region*) et par environnement d'application (*environment*). Image Builder et AWSTOE prennent en charge les environnements d'application suivants : `prod`, `preprod`, et `beta`.

- Le compartiment AWSTOE du gestionnaire de composants :

  ```
  s3://ec2imagebuilder-toe-region-environment
  ```

  Exemple : s3://ec2imagebuilder-toe-us-west-2-prod/ *
- Le compartiment de ressources gérées par Image Builder :

  ```
  s3://ec2imagebuilder-managed-resources-region-environment/components
  ```

  Exemple : s3://ec2imagebuilder-managed-resources-us-west-2 - prod/components /*

Exemples de politique de point de terminaison de VPC

Cette section inclut des exemples de politiques personnalisées relatives aux terminaux VPC.

Politique générale relative aux terminaux VPC pour les actions d'Image Builder

L'exemple de politique de point de terminaison suivant pour Image Builder refuse l'autorisation de supprimer des images et des composants d'Image Builder. L'exemple de politique accorde également l'autorisation d'effectuer toutes les autres actions d'EC2 Image Builder.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Action": "imagebuilder:*",
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": "imagebuilder:DeleteImage",
            "Effect": "Deny",
            "Resource": "*"
        },
        {
            "Action": "imagebuilder:DeleteComponent",
            "Effect": "Deny",
            "Resource": "*"
        }
    ]
}
```

Restreindre l'accès par organisation, autoriser l'accès aux composants gérés

L'exemple de politique de point de terminaison suivant montre comment restreindre l'accès aux identités et aux ressources appartenant à votre organisation et fournir un accès aux composants Amazon-managed d'Image Builder. Remplacez *regionprincipal-org-id*, et *resource-org-id* par les valeurs de votre organisation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsByOrgsIdentitiesToOrgsResources",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "principal-org-id",
                    "aws:ResourceOrgID": "resource-org-id"
```

```
                }
            }
        },
        {
            "Sid": "AllowAccessToEC2ImageBuilderComponents",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": [
                "imagebuilder:GetComponent"
            ],
            "Resource": [
                "arn:aws:imagebuilder:us-east-1:aws:component/*"
            ]
        }
    ]
}
```

Politique de point de terminaison VPC pour l'accès au compartiment Amazon S3

L'exemple de politique de point de terminaison S3 suivant montre comment fournir un accès aux compartiments S3 qu'Image Builder utilise pour créer des images personnalisées. Remplacez *region* et *environment* par les valeurs de votre organisation. Ajoutez toutes les autres autorisations requises à la politique en fonction des exigences de votre application.

> **Note**
>
> Pour les images Linux, si vous ne spécifiez pas de données utilisateur dans votre recette d'image, Image Builder ajoute un script pour télécharger et installer l'agent Systems Manager sur les instances de génération et de test de votre image. Pour télécharger l'agent, Image Builder accède au compartiment S3 de votre région de build.
> Pour vous assurer qu'Image Builder peut démarrer les instances de génération et de test, ajoutez la ressource supplémentaire suivante à votre politique de point de terminaison S3 : `"arn:aws:s3:::amazon-ssm-region/*"`

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowImageBuilderAccessToAppAndComponentBuckets",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::ec2imagebuilder-toe-region-environment/*",
        "arn:aws:s3:::ec2imagebuilder-managed-resources-region-environment/
components/*"
      ]
    }
  ]
}
```

Gérer les paramètres de distribution d'Image Builder

Avant de configurer les paramètres de distribution de vos images de sortie, nous vous recommandons de vérifier la disponibilité de toute infrastructure sous-jacente ou d'autres exigences pour les instances lancées à partir de votre image de sortie dans les régions cibles de distribution. Par exemple, toutes les régions ne prennent pas en charge les hôtes dédiés EC2 pour Mac, qui sont nécessaires pour lancer des instances à partir d'une image macOS. Pour plus d'informations sur les types d'instances et la tarification des hôtes dédiés, consultez la rubrique Tarification des hôtes dédiés Amazon EC2.

Après avoir créé les paramètres de distribution avec Image Builder, vous pouvez les gérer à l'aide de la console Image Builder, de l'API Image Builder ou imagebuilder des commandes du AWS CLI. Les paramètres de distribution vous permettent d'effectuer les actions suivantes :

Diffusion de l'AMI

- Spécifiez le nom et la description de votre AMI de sortie.

- Autorisez Comptes AWS d'autres organisations et unités d'organisation à lancer l'AMI depuis le compte du propriétaire. Le compte propriétaire est facturé pour les frais associés à l'AMI.

  > **Note**
  >
  > Pour rendre une AMI publique, définissez les comptes autorisés pour l'autorisation de lancement sur `all`. Pour plus d'informations et des exemples, consultez ModifyImageAttribute le manuel Amazon EC2 API Reference.

  > **Important**
  >
  > Vous ne pouvez pas rendre publiques les AMI comportant des filigranes. Si votre recette d'image inclut des filigranes, ou si des filigranes sont attachés à l'AMI source, vous ne pouvez pas définir les autorisations de lancement comme publiques. Image Builder rejette la configuration de distribution. Les filigranes se propagent automatiquement lorsque vous copiez ou distribuez une AMI entre les régions ou les comptes. Pour de plus amples informations, veuillez consulter Suivez la lignée des AMI à l'aide de filigranes.

- Créez une copie de l'AMI de sortie pour chacun des comptes, organisations et unités d'organisation cibles spécifiés dans la région de destination. Les comptes, organisations et unités d'organisation cibles sont propriétaires de leurs copies AMI et sont facturés pour tous les frais associés. Pour plus d'informations sur la distribution de votre AMI aux unités d'organisation AWS Organizations et aux unités d'organisation, voir Partager une AMI avec des organisations ou des unités d'organisation.
- Copiez l'AMI sur le compte du propriétaire dans un autre compte Régions AWS.
- Exportez des disques d'image de machine virtuelle vers Amazon Simple Storage Service (Amazon S3). Pour de plus amples informations, veuillez consulter Exemple : créez des paramètres de distribution pour les disques de sortie des machines virtuelles à partir du AWS CLI.

Distribution d'images de conteneurs

- Spécifiez le référentiel ECR dans lequel Image Builder stocke l'image de sortie dans la région de distribution.

Vous pouvez utiliser vos paramètres de distribution de la manière suivante pour diffuser des images aux régions, comptes AWS Organizations et unités organisationnelles (UO) cibles en une seule fois ou à chaque création de pipeline :

- Pour diffuser automatiquement des images mises à jour à des régions, des comptes, des organisations et des unités d'organisation spécifiques, utilisez les paramètres de distribution avec un pipeline Image Builder qui s'exécute selon un calendrier.
- Pour créer une nouvelle image et la diffuser aux régions, comptes, organisations et unités d'organisation spécifiés, utilisez les paramètres de distribution d'un pipeline Image Builder que vous exécutez une seule fois depuis la console Image Builder, en utilisant Exécuter le pipeline dans le menu Actions.
- Pour créer une nouvelle image et la diffuser aux régions, comptes, organisations et unités d'organisation spécifiés, utilisez les paramètres de distribution à l'aide de l'action d'API ou de la commande Image Builder suivante dans le AWS CLI :
  - L'CreateImageaction dans l'API Image Builder.
  - La create-image commande contenue dans le AWS CLI.
- Exporter des disques d'image de machine virtuelle (VM) vers des compartiments S3 dans les régions cibles dans le cadre de votre processus de création d'image habituel.

> **Tip**
>
> Lorsque vous disposez de plusieurs ressources du même type, le balisage vous permet d'identifier une ressource spécifique en fonction des balises que vous lui avez attribuées. Pour plus d'informations sur le balisage de vos ressources à l'aide des commandes d'Image Builder AWS CLI, consultez la Balisage des ressources section de ce guide.

Table des matières

- Répertorier et afficher les détails de la configuration de distribution
- Création et mise à jour des configurations de distribution AMI
- Création et mise à jour des paramètres de distribution pour les images de conteneurs
- Configurer la distribution d'AMI entre comptes avec Image Builder
- Configuration de la distribution d'AMI avec un modèle de lancement EC2
- Utilisez des fonctionnalités de distribution d'AMI améliorées

Répertorier et afficher les détails de la configuration de distribution

Cette section décrit les différentes manières de trouver des informations et de visualiser les détails de la configuration de distribution de votre EC2 Image Builder.

Détail de la configuration de distribution

- Répertorier les configurations de distribution depuis la console
- Afficher les détails de configuration de distribution depuis la console
- Répertorier les distributions provenant du AWS CLI
- Obtenez des informations détaillées sur la configuration de la distribution à partir du AWS CLI

Répertorier les configurations de distribution depuis la console

Pour consulter la liste des configurations de distribution créées sous votre compte dans la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez les paramètres de distribution dans le volet de navigation. Cela affiche la liste des configurations de distribution créées sous votre compte.
3. Pour afficher les détails ou créer une nouvelle configuration de distribution, cliquez sur le lien Nom de la configuration. Cela ouvre la vue détaillée des paramètres de distribution.

> **Note**
>
> Vous pouvez également cocher la case à côté du nom de la configuration, puis choisir Afficher les détails.

Afficher les détails de configuration de distribution depuis la console

Pour afficher les détails d'une configuration de distribution spécifique à l'aide de la console Image Builder, sélectionnez la configuration à consulter en suivant les étapes décrites dansRépertorier les configurations de distribution depuis la console.

Sur la page détaillée de la distribution, vous pouvez :

- Supprimez la configuration de distribution. Pour plus d'informations sur la suppression de ressources dans Image Builder, consultezSupprimer les ressources d'Image Builder obsolètes ou inutilisées.
- Modifiez les détails de distribution.

Répertorier les distributions provenant du AWS CLI

L'exemple suivant montre comment utiliser la list-distribution-configurations commande du AWS CLI pour répertorier toutes vos distributions.

```
aws imagebuilder list-distribution-configurations
```

Obtenez des informations détaillées sur la configuration de la distribution à partir du AWS CLI

L'exemple suivant montre comment utiliser la get-distribution-configuration commande du AWS CLI pour obtenir les détails d'une configuration de distribution en spécifiant son Amazon Resource Name (ARN).

```
aws imagebuilder get-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

Création et mise à jour des configurations de distribution AMI

Cette section couvre la création et la mise à jour de configurations de distribution pour une AMI Image Builder.

Table des matières

- Prérequis pour la distribution de l'AMI
- Création d'une configuration de distribution AMI
- Mettre à jour une configuration de distribution AMI
- Exemple : activer EC2 Fast Launch avec un modèle de lancement pour les AMI de sortie
- Exemple : créez des paramètres de distribution pour les disques de sortie des machines virtuelles à partir du AWS CLI

Prérequis pour la distribution de l'AMI

Certains paramètres de distribution sont soumis à des prérequis, comme suit :

Rubriques

- Conditions requises pour les paramètres de sortie SSM
- Prérequis pour le lancement rapide d'EC2

Conditions requises pour les paramètres de sortie SSM

Avant de créer une nouvelle configuration de distribution AMI qui définit un AWS Systems Manager paramètre Parameter Store (paramètre SSM), assurez-vous que vous remplissez les conditions préalables suivantes.

Rôle d'exécution

Lorsque vous créez un pipeline ou que vous utilisez la commande create-image dans le AWS CLI, vous ne pouvez spécifier qu'un seul rôle d'exécution d'Image Builder. Si vous avez défini un rôle d'exécution du flux de travail Image Builder, vous devez ajouter toutes les autorisations de fonctionnalité supplémentaires à ce rôle. Sinon, vous créeriez un nouveau rôle personnalisé qui inclut les autorisations requises.

> **Important**
>
> Nous vous recommandons de ne pas transmettre le rôle AWSServiceRoleForImageBuilder lié à un service en tant que rôle d'exécution. Créez plutôt un rôle IAM personnalisé et associez la politique EC2ImageBuilderExecutionPolicy AWS gérée. Cette politique accorde les mêmes autorisations dont Image Builder a besoin pour appeler Services AWS en votre nom. L'utilisation d'un rôle personnalisé vous permet de contrôler totalement les autorisations utilisées par Image Builder. Il permet également de maintenir vos politiques de contrôle des services (SCP) et vos politiques de contrôle des ressources (RCP) en vigueur pour les opérations qu'Image Builder effectue en votre nom.

- Pour stocker l'ID AMI de sortie dans un paramètre SSM pendant la distribution, vous devez spécifier l'`ssm:PutParameter`action dans votre rôle d'exécution d'Image Builder, le paramètre étant répertorié en tant que ressource.

- Lorsque vous définissez le type de données du paramètre `AWS EC2 Image` pour indiquer à Systems Manager de valider la valeur du paramètre en tant qu'ID AMI, vous devez également ajouter l'`ec2:DescribeImages`action.

Opt-in Région

Vous devez activer une région opt-in (région désactivée par défaut) dans le compte source. Cette exigence s'applique lorsque votre configuration de distribution cible cette région`ssmParameterConfigurations`, que vous utilisiez le rôle lié au service Image Builder ou un rôle d'exécution personnalisé.

> **Note**
>
> La copie de l'AMI vers une région optionnelle peut réussir même si le compte source n'a pas activé la région. Seule l'écriture du paramètre de sortie SSM échoue.

Sans cette condition préalable, la compilation échoue avec une erreur similaire à la suivante :

```
In region 'af-south-1' - 'STS Client Error: The provided execution role does not
 exist or does not have sufficient permissions.'
```

Pour résoudre ce problème, procédez de l'une des manières suivantes :

- Activez les régions optionnelles dans le compte source. Pour obtenir des instructions sur l'activation d'une région, voir Activer ou désactiver AWS des régions dans votre compte dans le Guide de référence de gestion des AWS comptes.
- Omettre `ssmParameterConfigurations` pour les régions optionnelles. Utilisez plutôt une EventBridge règle Amazon pour réagir à l'événement EC2 Image Builder Image Builder Image State Change et écrivez vous-même les paramètres.

Prérequis pour le lancement rapide d'EC2

Avant de créer une nouvelle configuration de distribution pour les AMI EC2 Fast Launch pour Windows, assurez-vous que vous remplissez les conditions préalables suivantes.

- Si vous fournissez un modèle de lancement personnalisé lorsque vous configurez EC2 Fast Launch, le service utilise le VPC et les autres paramètres de configuration que vous avez définis dans le modèle de lancement. Pour plus d'informations, voir Utiliser un modèle de lancement lorsque vous configurez EC2 Fast Launch.

- Si vous n'utilisez pas de modèle de lancement personnalisé pour configurer vos paramètres, vous devez associer la EC2FastLaunchFullAccess politique au rôle IAM qu'Image Builder utilise pour créer votre image. Lorsque vous créez un pipeline ou que vous utilisez la commande create-image dans le AWS CLI, vous ne pouvez spécifier qu'un seul rôle d'exécution d'Image Builder. Si vous avez défini un rôle d'exécution du flux de travail Image Builder, vous devez ajouter toutes les autorisations de fonctionnalité supplémentaires à ce rôle. Sinon, vous créeriez un nouveau rôle personnalisé qui inclut les autorisations requises.

  Cette fonctionnalité nécessite des autorisations supplémentaires en plus de celles prévues dans la politique d'exécution de base. Vous devez donc utiliser un rôle personnalisé. Nous vous recommandons de créer un rôle personnalisé avec la stratégie EC2ImageBuilderExecutionPolicy AWS gérée associée, puis d'ajouter la EC2FastLaunchFullAccess stratégie à ce rôle.

  Ensuite, lorsqu'Image Builder copie votre image, EC2 Fast Launch crée automatiquement une CloudFormation pile contenant les ressources suivantes dans votre Compte AWS.

  - Un cloud privé virtuel (VPC)
  - Des sous-réseaux privés sur plusieurs zones de disponibilité
  - Un modèle de lancement configuré avec le service des métadonnées d'instance version 2 (IMDSv2)
  - Un groupe de sécurité sans règles entrantes ou sortantes

> **Note**
>
> Image Builder ne prend pas en charge la distribution entre comptes pour les AMI sur lesquelles EC2 Fast Launch est préactivé. EC2 Fast Launch doit être activé depuis le compte de destination.

Création d'une configuration de distribution AMI

Les configurations de distribution incluent le nom de l'AMI de sortie, les paramètres de région spécifiques pour le chiffrement, les autorisations de lancement Comptes AWS, les organisations et les unités organisationnelles (UO) qui peuvent lancer l'AMI de sortie, ainsi que les configurations de licence.

Une configuration de distribution vous permet de spécifier le nom et la description de votre AMI de sortie, d'autoriser d'autres Comptes AWS personnes à lancer l'AMI, de copier l'AMI sur d'autres

comptes et de répliquer l'AMI vers d'autres AWS régions. Il vous permet également d'exporter l'AMI vers Amazon Simple Storage Service (Amazon S3) ou de configurer EC2 Fast Launch pour les AMI Windows de sortie. Pour rendre une AMI publique, définissez les comptes autorisés pour l'autorisation de lancement sur `all`. Consultez les exemples de création d'une AMI publique sur EC2. ModifyImageAttribute

Console

Procédez comme suit pour créer une nouvelle configuration de distribution AMI dans Console de gestion AWS :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez les paramètres de distribution dans le volet de navigation. Cela affiche la liste des configurations de distribution créées sous votre compte.
3. Choisissez Créer des paramètres de distribution en haut du panneau des paramètres de distribution.
4. Dans la section Type d'image, choisissez le type de sortie Amazon Machine Image (AMI).
5. Dans la section Général, entrez un nom pour votre configuration de distribution, ainsi qu'une description facultative.
6. Dans la section Paramètres régionaux, entrez les informations suivantes pour chaque région dans laquelle vous distribuez votre AMI :

   a. L'AMI est distribuée dans la région actuelle (région 1), par défaut. La région 1 est la source de la distribution. Certains paramètres de la Région 1 ne peuvent pas être modifiés. Pour toutes les régions que vous ajoutez, vous pouvez choisir une région dans la liste déroulante des régions.

      La clé Kms identifie celle AWS KMS key qui est utilisée pour chiffrer les volumes EBS de votre image dans la région cible. Il est important de noter que cela ne s'applique pas à l'AMI d'origine créée par le build sous votre compte dans la région source (région 1). Le cryptage qui s'exécute pendant la phase de distribution de la version concerne uniquement les images distribuées vers d'autres comptes ou régions.

      Pour chiffrer les volumes EBS pour l'AMI créée dans la région source de votre compte, vous devez définir la clé KMS dans l'image, la recette, le bloc, le mappage des appareils (Stockage (volumes) dans la console).

Image Builder copie l'AMI vers les comptes cibles que vous spécifiez pour la région.

> **Prérequis**
>
> Pour copier une image sur plusieurs comptes, vous devez créer le `EC2ImageBuilderDistributionCrossAccountRole` rôle dans tous les comptes cibles de distribution et associer la politique Ec2ImageBuilderCrossAccountDistributionAccess policy gérée au rôle.

Le nom de l'AMI de sortie est facultatif. Si vous fournissez un nom, le nom final de l'AMI en sortie inclut un horodatage ajouté indiquant la date de création de l'AMI. Si vous ne spécifiez pas de nom, Image Builder ajoute l'horodatage de génération au nom de la recette. Cela garantit des noms d'AMI uniques pour chaque build.

i. Avec le partage d'AMI, vous pouvez accorder l'accès à AWS des principaux spécifiques pour lancer des instances à partir de votre AMI. Si vous développez la section Partage d'AMI, vous pouvez saisir les informations suivantes :

   - Autorisations de lancement : sélectionnez Privé si vous souhaitez préserver la confidentialité de votre AMI et autoriser l'accès à des AWS principaux spécifiques pour lancer une instance à partir de votre AMI privée. Sélectionnez Public si vous souhaitez rendre votre AMI publique. N'importe quel AWS principal peut lancer une instance depuis votre AMI publique.
   - Principaux : vous pouvez accorder l'accès aux types de AWS principaux suivants pour lancer des instances :
     - AWS compte — Autoriser l'accès à un AWS compte spécifique
     - Unité organisationnelle (UO) : accordez l'accès à une UO et à toutes ses entités enfants. Les entités enfants incluent les unités d'organisation et AWS les comptes.
     - Organisation : accordez l'accès à vos AWS Organizations entités et à toutes ses entités enfants. Les entités enfants incluent les unités d'organisation et AWS les comptes.

Sélectionnez d'abord le type principal. Entrez ensuite l'ID du AWS principal auquel vous souhaitez accorder l'accès dans la case située à droite de la liste déroulante. Vous pouvez saisir plusieurs identifiants de différents types.

ii. Vous pouvez développer la section Configuration des licences pour joindre les configurations de licence créées avec AWS License Manager à vos images Image Builder. Les configurations de licence contiennent des règles de licence basées sur les termes de vos contrats d'entreprise. Image Builder inclut automatiquement les configurations de licence associées à votre AMI de base.

iii. Vous pouvez développer la section Configuration du modèle de lancement pour spécifier un modèle de lancement EC2 à utiliser pour lancer des instances à partir de l'AMI que vous créez.

Si vous utilisez un modèle de lancement EC2, vous pouvez demander à Image Builder de créer une nouvelle version de votre modèle de lancement qui inclut le dernier ID AMI une fois la génération terminée. Pour mettre à jour le modèle de lancement, configurez les paramètres comme suit :

- Nom du modèle de lancement : sélectionnez le nom du modèle de lancement que vous souhaitez qu'Image Builder mette à jour.
- Définir la version par défaut : cochez cette case pour mettre à jour la version par défaut du modèle de lancement vers la nouvelle version.

Pour ajouter une autre configuration de modèle de lancement, choisissez Ajouter une configuration de modèle de lancement. Vous pouvez avoir jusqu'à cinq configurations de modèles de lancement par région.

iv. Vous pouvez développer la section Configurations des paramètres SSM pour configurer un paramètre SSM qui stockera l'ID AMI de sortie pour l'image distribuée dans la région de destination. Vous pouvez éventuellement spécifier un compte de distribution dans la région.

Nom du paramètre — Entrez le nom de votre paramètre. Par exemple `/output/image/param`.

Type de données — Conservez la valeur par défaut (`AWS EC2 Image`). Cela indique à Systems Manager de valider la valeur du paramètre pour s'assurer qu'il s'agit d'un ID AMI valide.

b. Pour ajouter des paramètres de distribution pour une autre région, choisissez Ajouter une région.

7. Choisissez Créer des paramètres lorsque vous avez terminé.

AWS CLI

L'exemple suivant montre comment utiliser la create-distribution-configuration commande pour créer une nouvelle configuration de distribution pour votre AMI, à l'aide du AWS CLI.

1. Créer un fichier JSON d'entrée CLI

   Utilisez un outil d'édition de fichiers pour créer un fichier JSON avec les clés illustrées dans l'un des exemples suivants et des valeurs valides pour votre environnement. Ces exemples définissent quelles Comptes AWS unités organisationnelles (AWS Organizations OU) sont autorisées à lancer l'AMI que vous distribuez dans les régions spécifiées. Donnez un nom au fichier`create-ami-distribution-configuration.json`, à utiliser à l'étape suivante :

   Exemple 1 : distribuer à Comptes AWS

   Cet exemple distribue une AMI à deux régions et Comptes AWS indique quelles régions disposent d'autorisations de lancement dans chaque région.

```
{
 "name": "MyExampleAccountDistribution",
 "description": "Copies AMI to eu-west-1, and specifies accounts that can launch
 instances in each Region.",
 "distributions": [
  {
   "region": "us-west-2",
   "amiDistributionConfiguration": {
    "name": "Name {{imagebuilder:buildDate}}",
    "description": "An example image name with parameter references",
    "amiTags": {
     "KeyName": "Some Value"
    },
    "launchPermission": {
     "userIds": [
      "987654321012"
     ]
    }
   }
```

```
  },
  {
   "region": "eu-west-1",
   "amiDistributionConfiguration": {
    "name": "My {{imagebuilder:buildVersion}} image {{imagebuilder:buildDate}}",
    "amiTags": {
     "KeyName": "Some value"
    },
    "launchPermission": {
     "userIds": [
      "100000000001"
     ]
    }
   }
  }
 ]
}
```

Exemple 2 : Distribution aux organisations et aux unités d'organisation

Cet exemple distribue une AMI à la région source et spécifie les autorisations de lancement de l'organisation et de l'unité d'organisation.

```
{
 "name": "MyExampleAWSOrganizationDistribution",
 "description": "Shares AMI with the Organization and OU",
 "distributions": [
  {
   "region": "us-west-2",
   "amiDistributionConfiguration": {
    "name": "Name {{ imagebuilder:buildDate }}",
    "launchPermission": {
     "organizationArns": [
      "arn:aws:organizations::123456789012:organization/o-myorganization123"
     ],
     "organizationalUnitArns": [
      "arn:aws:organizations::123456789012:ou/o-123example/ou-1234-
myorganizationalunit"
     ]
    }
   }
  }
 ]
```

```
}
```

Exemple 3 : stocker l'ID AMI de sortie dans un paramètre SSM

Cet exemple enregistre l'ID AMI en sortie dans un AWS Systems Manager paramètre de magasin de paramètres dans la région de distribution.

```
{
 "name": "SSMParameterOutputAMI",
 "description": "Updates an SSM parameter with the output AMI ID for the
 distribution.",
 "distributions": [
  {
   "region": "us-west-2",
   "amiDistributionConfiguration": {
    "name": "Name {{ imagebuilder:buildDate }}"
   },
   "ssmParameterConfigurations": [
    {
     "amiAccountId": "111122223333",
     "parameterName": "/output/image/param",
     "dataType": "aws:ec2:image"
    }
   ]
  }
 ]
}
```

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder create-distribution-configuration --cli-input-json
 file://create-ami-distribution-configuration.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par

exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Pour plus d'informations, consultez create-distribution-configuration la référence des AWS CLI commandes.

Mettre à jour une configuration de distribution AMI

Vous pouvez modifier la configuration de distribution de votre AMI. Toutefois, les modifications que vous apportez ne s'appliquent pas aux ressources qu'Image Builder a déjà distribuées. Par exemple, si vous avez distribué une AMI dans une région que vous supprimez ultérieurement de votre distribution, l'AMI qui était déjà distribuée reste dans cette région jusqu'à ce que vous la supprimiez manuellement.

Console de gestion AWS

Procédez comme suit pour configurer une distribution AMI dans le Console de gestion AWS :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez les paramètres de distribution dans le volet de navigation. Cela affiche la liste des configurations de distribution créées sous votre compte.
3. Pour afficher les détails ou mettre à jour une configuration de distribution, cliquez sur le lien Nom de la configuration. Cela ouvre la vue détaillée des paramètres de distribution.

> **Note**
>
> Vous pouvez également cocher la case à côté du nom de la configuration, puis choisir Afficher les détails.

4. Pour modifier la configuration de distribution, choisissez Modifier dans le coin supérieur droit de la section Détails de distribution. Certains champs sont verrouillés, tels que le nom de la configuration de distribution et la région par défaut qui s'affiche en tant que région 1. Pour plus d'informations sur les paramètres de configuration de distribution, consultezCréation d'une configuration de distribution AMI.
5. Lorsque vous avez terminé, choisissez Save changes (Enregistrer les modifications).

AWS CLI

L'exemple suivant montre comment utiliser la update-distribution-configuration commande pour mettre à jour les paramètres de distribution de votre AMI, à l'aide du AWS CLI.

1. Créer un fichier JSON d'entrée CLI

   Utilisez un outil d'édition de fichiers pour créer un fichier JSON avec les clés illustrées dans l'exemple suivant et des valeurs valides pour votre environnement. Cet exemple utilise un fichier nomméupdate-ami-distribution-configuration.json.

```
{
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-ami-distribution-
configuration.json",
 "description": "Copies AMI to eu-west-2, and specifies accounts that can launch
 instances in each Region.",
 "distributions": [
   {
   "region": "us-west-2",
   "amiDistributionConfiguration": {
    "name": "Name {{imagebuilder:buildDate}}",
    "description": "An example image name with parameter references",
    "launchPermissions": {
     "userIds": [
      "987654321012"
     ]
    }
   }
  },
  {
   "region": "eu-west-2",
   "amiDistributionConfiguration": {
    "name": "My {{imagebuilder:buildVersion}} image {{imagebuilder:buildDate}}",
    "tags": {
     "KeyName": "Some value"
    },
    "launchPermissions": {
     "userIds": [
      "100000000001"
     ]
    }
   }
```

```
  }
 ]
}
```

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder update-distribution-configuration --cli-input-json
 file://update-ami-distribution-configuration.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Pour plus d'informations, consultez update-distribution-configuration la référence des AWS CLI commandes. Pour mettre à jour les balises de votre ressource de configuration de distribution, consultez la Balisage des ressources section.

Exemple : activer EC2 Fast Launch avec un modèle de lancement pour les AMI de sortie

L'exemple suivant montre comment utiliser la create-distribution-configuration commande avec un modèle de lancement pour créer des paramètres de distribution dans lesquels EC2 Fast Launch est configuré pour votre AMI, à partir du AWS CLI.

Pour configurer les paramètres de lancement rapide d'EC2 sans modèle de lancement, assurez-vous de remplir toutes les conditions Conditions préalables au lancement rapide d'EC2 avant de créer votre configuration de distribution.

1. Créer un fichier JSON d'entrée CLI

   Utilisez un outil d'édition de fichiers pour créer un fichier JSON contenant des clés, comme indiqué dans l'exemple suivant, ainsi que des valeurs valides pour votre environnement.

Cet exemple lance des instances pour toutes ses ressources cibles simultanément, car le nombre maximum de lancements parallèles est supérieur au nombre de ressources cibles. Ce fichier est nommé `ami-dist-config-win-fast-launch.json` dans l'exemple de commande présenté à l'étape suivante.

```
{
"name": "WinFastLaunchDistribution",
"description": "An example of Windows AMI EC2 Fast Launch settings in the
 distribution configuration.",
"distributions": [
 {
  "region": "us-west-2",
  "amiDistributionConfiguration": {
   "name": "Name {{imagebuilder:buildDate}}",
   "description": "Includes Windows AMI EC2 Fast Launch settings.",
   "amiTags": {
    "KeyName": "Some Value"
   }
  },
  "fastLaunchConfigurations": [{
   "enabled": true,
   "snapshotConfiguration": {
    "targetResourceCount": 5
   },
   "maxParallelLaunches": 6,
   "launchTemplate": {
    "launchTemplateId": "lt-0ab1234c56d789012",
    "launchTemplateVersion": "1"
    }
  }],
  "launchTemplateConfigurations": [{
       "launchTemplateId": "lt-0ab1234c56d789012",
       "setDefaultVersion": true
    }]
 }]
}
```

> **Note**
>
> Vous pouvez spécifier le `launchTemplateName` au lieu de le `launchTemplateId` dans la `launchTemplate` section, mais vous ne pouvez pas spécifier à la fois le nom et l'identifiant.

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://ami-dist-config-win-fast-launch.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Pour plus d'informations, consultez create-distribution-configuration la référence des AWS CLI commandes.

Exemple : créez des paramètres de distribution pour les disques de sortie des machines virtuelles à partir du AWS CLI

L'exemple suivant montre comment utiliser la create-distribution-configuration commande pour créer des paramètres de distribution qui exporteront des disques d'image de machine virtuelle vers Amazon S3 à chaque création d'image.

1. Créer un fichier JSON d'entrée CLI

   Vous pouvez rationaliser la create-distribution-configuration commande que vous utilisez dans le AWS CLI. Pour ce faire, créez un fichier JSON contenant toutes les configurations d'exportation que vous souhaitez transmettre à la commande.

> **Note**
>
> La convention de dénomination des valeurs de données du fichier JSON suit le modèle spécifié pour les paramètres de demande d'opération de l'API Image Builder. Pour passer en revue les paramètres de demande d'opération d'API, consultez la CreateDistributionConfiguration commande dans le document de référence de l'API EC2 Image Builder.
> Pour fournir les valeurs de données sous forme de paramètres de ligne de commande, reportez-vous aux noms de paramètres spécifiés dans la référence des AWS CLI commandes. À la create-distribution-configuration commande en tant qu'options.

Voici un résumé des paramètres que nous spécifions dans l'objet `s3ExportConfiguration` JSON pour cet exemple :

- roleName (chaîne, obligatoire) : nom du rôle qui Import/Export autorise la machine virtuelle à exporter des images vers votre compartiment S3.
- disque ImageFormat (chaîne, obligatoire) — Exportez l'image disque mise à jour vers l'un des formats pris en charge suivants :
  - Disque dur virtuel (VHD) : compatible avec les produits de Hyper-V virtualisation Citrix Xen et Microsoft.
  - Stream-optimized Disque de machine virtuelle ESX (VMDK) : compatible avec VMware ESX et VMware vSphere versions 4, 5 et 6.
  - Raw — Format brut.
- S3bucket (chaîne, obligatoire) — Le compartiment S3 dans lequel stocker les images de disque de sortie pour votre machine virtuelle.

Enregistrez le fichier sous le nom `export-vm-disks.json`. Utilisez le nom de fichier dans la create-distribution-configuration commande.

```
{
 "name": "example-distribution-configuration-with-vm-export",
 "description": "example",
 "distributions": [
  {
   "region": "us-west-2",
```

```
   "amiDistributionConfiguration": {
    "description": "example-with-vm-export"

   },
   "s3ExportConfiguration": {
    "roleName": "vmimport",
    "diskImageFormat": "RAW",
    "s3Bucket": "vm-bucket-export"
   }
  }],
 "clientToken": "abc123def4567ab"
}
```

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://export-vm-disks.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Pour plus d'informations, consultez create-distribution-configuration la référence des AWS CLI commandes.

Création et mise à jour des paramètres de distribution pour les images de conteneurs

Cette section couvre la création et la mise à jour des paramètres de distribution pour les images de conteneur Image Builder.

Table des matières

- Créez des paramètres de distribution pour les images de conteneurs Image Builder à partir du AWS CLI
- Mettez à jour les paramètres de distribution de l'image de votre conteneur à partir du AWS CLI

Créez des paramètres de distribution pour les images de conteneurs Image Builder à partir du AWS CLI

Une configuration de distribution vous permet de spécifier le nom et la description de votre image de conteneur en sortie et de répliquer l'image de conteneur dans d'autres AWS régions. Vous pouvez également appliquer des balises distinctes à la ressource de configuration de distribution et aux images de conteneur dans chaque région.

1. Créer un fichier JSON d'entrée CLI

   Utilisez votre outil d'édition de fichiers préféré pour créer un fichier JSON avec les clés illustrées dans l'exemple suivant, ainsi que des valeurs valides pour votre environnement. Cet exemple utilise un fichier nommé`create-container-distribution-configuration.json` :

```
{
 "name": "distribution-configuration-name",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
  {
   "region": "us-west-2",
   "containerDistributionConfiguration": {
    "description": "My test image.",
    "targetRepository": {
     "service": "ECR",
     "repositoryName": "testrepo"
    },
    "containerTags": ["west2", "image1"]
   }
  },
  {
   "region": "us-east-1",
   "containerDistributionConfiguration": {
    "description": "My test image.",
    "targetRepository": {
     "service": "ECR",
     "repositoryName": "testrepo"
```

```
    },
      "containerTags": ["east1", "imagedist"]
   }
  }
 ],
 "tags": {
    "DistributionConfigurationTestTagKey1":
 "DistributionConfigurationTestTagValue1",
    "DistributionConfigurationTestTagKey2":
 "DistributionConfigurationTestTagValue2"
 }
}
```

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://create-container-distribution-configuration.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Pour plus d'informations, consultez create-distribution-configuration la référence des AWS CLI commandes.

Mettez à jour les paramètres de distribution de l'image de votre conteneur à partir du AWS CLI

L'exemple suivant montre comment utiliser la update-distribution-configuration commande pour mettre à jour les paramètres de distribution de votre image de conteneur, à l'aide du AWS CLI. Vous pouvez également mettre à jour les balises pour les images de conteneurs dans chaque région.

1. Créer un fichier JSON d'entrée CLI

   Utilisez votre outil d'édition de fichiers préféré pour créer un fichier JSON avec les clés illustrées dans l'exemple suivant, ainsi que des valeurs valides pour votre environnement. Cet exemple utilise un fichier nomméupdate-container-distribution-configuration.json :

```
{
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-container-distribution-
configuration.json",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
  {
   "region": "us-west-2",
   "containerDistributionConfiguration": {
    "description": "My test image.",
    "targetRepository": {
     "service": "ECR",
     "repositoryName": "testrepo"
    },
    "containerTags": ["west2", "image1"]
   }
  },
  {
   "region": "us-east-2",
   "containerDistributionConfiguration": {
    "description": "My test image.",
    "targetRepository": {
     "service": "ECR",
     "repositoryName": "testrepo"
    },
      "containerTags": ["east2", "imagedist"]
   }
  }
 ]
}
```

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé comme entrée :

```
aws imagebuilder update-distribution-configuration --cli-input-json file://update-
container-distribution-configuration.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Pour plus d'informations, consultez update-distribution-configuration la référence des AWS CLI commandes. Pour mettre à jour les balises de votre ressource de configuration de distribution, consultez la Balisage des ressources section.

Configurer la distribution d'AMI entre comptes avec Image Builder

Cette section explique comment configurer les paramètres de distribution pour fournir une AMI Image Builder aux autres comptes que vous spécifiez.

Le compte de destination peut ensuite lancer ou modifier l'AMI, selon les besoins.

> **Tip**
>
> Pour plus d'informations sur un exemple déployable qui inclut la politique AWS KMS clé et le rôle du compte cible, consultez l'exemple d'AMI multicomptes sur. GitHub Le même référentiel inclut une variante Terraform. Pour de plus amples informations, veuillez consulter Découvrez des exemples de projets Image Builder sur GitHub.

> **Note**
>
> AWS CLI les exemples de commandes de cette section supposent que vous avez déjà créé des fichiers JSON de recette d'image et de configuration d'infrastructure. Pour créer le fichier JSON pour une recette d'image, consultezCréez une recette imagée à l'aide du AWS CLI. Pour créer le fichier JSON pour une configuration d'infrastructure, consultezCréer une configuration d'infrastructure.

Conditions requises pour la distribution d'AMI entre comptes

Pour vous assurer que les comptes cibles peuvent lancer des instances avec succès à partir de votre image Image Builder, vous devez configurer les autorisations appropriées pour tous les comptes de destination dans toutes les régions.

Si vous chiffrez votre AMI à l'aide de AWS Key Management Service (AWS KMS), vous devez configurer un AWS KMS key pour votre compte qui est utilisé pour chiffrer la nouvelle image.

Lorsqu'Image Builder effectue une distribution entre comptes pour les AMI chiffrées, l'image du compte source est déchiffrée et transmise à la région cible, où elle est rechiffrée à l'aide de la clé désignée pour cette région. Comme Image Builder agit pour le compte cible et utilise un rôle IAM que vous créez dans la région de destination, ce compte doit avoir accès aux clés des régions source et de destination.

Clés de chiffrement

Les conditions préalables suivantes sont requises si votre image est cryptée à l'aide AWS KMS de. Les prérequis IAM sont abordés dans la section suivante.

Exigences relatives au compte source

- Créez une clé KMS dans votre compte dans toutes les régions où vous créez et distribuez votre AMI. Vous pouvez également utiliser une clé existante.
- Mettez à jour la politique relative aux clés pour toutes ces clés afin de permettre aux comptes de destination d'utiliser votre clé.

Exigences relatives au compte de destination

- Ajoutez une politique en ligne `EC2ImageBuilderDistributionCrossAccountRole` qui permet au rôle d'effectuer les actions requises pour distribuer une AMI chiffrée. Pour les étapes de configuration IAM, consultez la section politiques IAM Prérequis.

Pour plus d'informations sur l'utilisation de l'accès entre comptes AWS KMS, voir Autoriser les utilisateurs d'autres comptes à utiliser une clé KMS dans le Guide du AWS Key Management Service développeur.

Spécifiez votre clé de chiffrement dans la recette de l'image, comme suit :

- Si vous utilisez la console Image Builder, choisissez votre clé de chiffrement dans la liste déroulante Chiffrement (alias KMS) de la section Stockage (volumes) de votre recette.
- Si vous utilisez l'action d'CreateImageRecipeAPI ou la create-image-recipe commande contenue dans le AWS CLI, configurez votre clé dans la `ebs` section `blockDeviceMappings` ci-dessous de votre entrée JSON.

  L'extrait JSON suivant indique les paramètres de chiffrement d'une recette d'image. En plus de fournir votre clé de chiffrement, vous devez également définir le `encrypted` drapeau sur`true`.

```
{
 ...
 "blockDeviceMappings": [
 {
  "deviceName": "Example root volume",
  "ebs": {
   "deleteOnTermination": true,
   "encrypted": true,
   "iops": 100,
   "kmsKeyId": "image-owner-key-id",
   ...
  },
  ...
 }],
 ...
}
```

politiques IAM

Pour configurer les autorisations de distribution entre comptes dans Gestion des identités et des accès AWS (IAM), procédez comme suit :

1. Pour utiliser les AMI Image Builder distribuées entre les comptes, le propriétaire du compte de destination doit créer un nouveau rôle IAM dans son compte appelé`EC2ImageBuilderDistributionCrossAccountRole`.
2. Ils doivent associer le Ec2ImageBuilderCrossAccountDistributionAccess policy au rôle pour permettre la distribution entre comptes. Pour plus d'informations sur les politiques gérées, voir Stratégies gérées et politiques intégrées dans le Guide de l'Gestion des identités et des accès AWS utilisateur.

3. Vérifiez que l'ID du compte source est ajouté à la politique de confiance associée au rôle IAM du compte de destination. L'exemple suivant montre une politique de confiance dans le compte de destination qui spécifie l'ID du compte source.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [{
           "Effect": "Allow",
           "Principal": {
               "AWS": "arn:aws:iam::444455556666:root"
           },
           "Action": "sts:AssumeRole"
       }]
   }
   ```

   Pour plus d'informations sur les politiques de confiance, consultez la section Resource-Based Politiques du Guide de Gestion des identités et des accès AWS l'utilisateur.

4. Si l'AMI que vous distribuez est cryptée, le propriétaire du compte de destination doit ajouter la politique `EC2ImageBuilderDistributionCrossAccountRole` en ligne suivante à son compte afin de pouvoir utiliser vos clés KMS. La `Principal` section contient leur numéro de compte. Cela permet à Image Builder d'agir en leur nom lorsqu'il AWS KMS crypte et déchiffre l'AMI avec les clés appropriées pour chaque région.

   JSON

   ```
   {
    "Version":"2012-10-17",
    "Statement": [
     {
      "Sid": "AllowRoleToPerformKMSOperationsOnBehalfOfTheDestinationAccount",
      "Effect": "Allow",
      "Action": [
       "kms:Encrypt",
       "kms:Decrypt",
       "kms:ReEncrypt*",
       "kms:GenerateDataKey*",
   ```

```
    "kms:DescribeKey",
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
   ],
   "Resource": "*"
  }
 ]
}
```

Pour plus d'informations sur les politiques intégrées, consultez la section Politiques intégrées dans le Guide de l'Gestion des identités et des accès AWS utilisateur.

5. Si vous utilisez `launchTemplateConfigurations` pour spécifier un modèle de lancement Amazon EC2, vous devez également ajouter la politique suivante `EC2ImageBuilderDistributionCrossAccountRole` à votre compte de destination.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplateVersion",
                "ec2:ModifyLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeLaunchTemplates"
            ],
            "Resource": "*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:launch-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        }
    ]
}
```

6. Si vous utilisez un AWS Systems Manager paramètre Parameter Store pour stocker l'ID AMI de l'AMI de sortie pour le compte de distribution et la région, vous devez ajouter la politique suivante `EC2ImageBuilderDistributionCrossAccountRole` à votre compte de destination.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter"
            ],
            "Resource": "arn:aws:ssm:*:111122223333:parameter/ImageBuilder-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages"
            ],
            "Resource": "*"
        }
    ]
}
```

Limites applicables à la distribution entre comptes

La distribution d'images Image Builder sur plusieurs comptes est soumise à certaines limites :

- Le compte de destination est limité à 50 copies AMI simultanées pour chaque région de destination.
- Si vous souhaitez copier une AMI de virtualisation paravirtuelle (PV) vers une autre région, la région de destination doit prendre en charge les AMI de virtualisation PV. Pour plus d'informations, consultez Types de virtualisations AMI Linux.
- Vous ne pouvez pas créer de copie non chiffrée d'un instantané chiffré. Si vous ne spécifiez pas de AWS Key Management Service (AWS KMS) clé gérée par le client pour le `KmsKeyId` paramètre, Image Builder utilise la clé par défaut pour Amazon Elastic Block Store (Amazon EBS). Pour de plus amples informations, veuillez consulter Chiffrement Amazon EBS dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

Pour plus d'informations, consultez CreateDistributionConfiguration la référence de l'API EC2 Image Builder.

Configurer la distribution entre comptes pour une AMI Image Builder depuis la console

Cette section explique comment créer et configurer les paramètres de distribution pour la distribution entre comptes de vos AMI Image Builder à l'aide du Console de gestion AWS. La configuration de la distribution entre comptes nécessite des autorisations IAM spécifiques. Vous devez remplir le Conditions requises pour la distribution d'AMI entre comptes formulaire pour cette section avant de continuer.

Pour créer des paramètres de distribution dans la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez les paramètres de distribution dans le volet de navigation. Cela affiche une liste des paramètres de distribution créés sous votre compte.
3. En haut de la page Paramètres de distribution, choisissez Créer des paramètres de distribution. Cela vous amène à la page Créer des paramètres de distribution.
4. Dans la section Type d'image, choisissez Amazon Machine Image (AMI) comme type de sortie. Il s'agit du paramètre par défaut.

5. Dans la section Général, entrez le nom de la ressource de paramètres de distribution que vous souhaitez créer (obligatoire).
6. Dans la section Paramètres régionaux, entrez un identifiant de compte à 12 chiffres auquel vous souhaitez distribuer votre AMI dans les comptes Target de la région sélectionnée, puis appuyez sur Entrée. Cela vérifie le formatage correct, puis affiche l'identifiant de compte que vous avez saisi sous la case. Répétez le processus pour ajouter d'autres comptes.

   Pour supprimer un compte que vous avez saisi, choisissez le X affiché à droite de l'identifiant du compte.

   Entrez le nom de l'AMI de sortie pour chaque région.
7. Continuez à spécifier les paramètres supplémentaires dont vous avez besoin, puis choisissez Créer des paramètres pour créer votre nouvelle ressource de paramètres de distribution.

Configurez la distribution entre comptes pour une AMI Image Builder à partir du AWS CLI

Cette section explique comment configurer un fichier de paramètres de distribution et utiliser la create-image commande du AWS CLI pour créer et distribuer une AMI Image Builder entre les comptes.

La configuration de la distribution entre comptes nécessite des autorisations IAM spécifiques. Vous devez compléter cette section avant d'exécuter la create-image commande. Conditions requises pour la distribution d'AMI entre comptes

1. Configuration d'un fichier de paramètres de distribution

   Avant d'utiliser la create-image commande du AWS CLI pour créer une AMI Image Builder distribuée à un autre compte, vous devez créer une structure `DistributionConfiguration` JSON qui spécifie les ID de compte cible dans les `AmiDistributionConfiguration` paramètres. Vous devez en spécifier au moins un `AmiDistributionConfiguration` dans la région source.

   L'exemple de fichier suivant, intitulé`create-distribution-configuration.json`, montre la configuration de la distribution d'images entre comptes dans la région source.

```
{
 "name": "cross-account-distribution-example",
```

```
 "description": "Cross Account Distribution Configuration Example",
 "distributions": [
  {
   "amiDistributionConfiguration": {
    "targetAccountIds": ["123456789012", "987654321098"],
    "name": "Name {{ imagebuilder:buildDate }}",
    "description": "ImageCopy Ami Copy Configuration"
   },
   "region": "us-west-2"
  }
 ]
}
```

2. Création des paramètres de distribution

   Pour créer une ressource de paramètres de distribution d'Image Builder à l'aide de la commande create-distribution-configuration du AWS CLI, indiquez les paramètres suivants dans la commande :

   - Entrez le nom de la distribution dans le `--name` paramètre.
   - Joignez le fichier JSON de configuration de distribution que vous avez créé dans le `--cli-input-json` paramètre.

```
aws imagebuilder create-distribution-configuration --name my distribution name --
cli-input-json file://create-distribution-configuration.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Vous pouvez également fournir du JSON directement dans la commande, à l'aide du `--distributions` paramètre.

Configuration de la distribution d'AMI avec un modèle de lancement EC2

Pour garantir une expérience de lancement cohérente pour votre AMI Image Builder dans les comptes cibles et les régions, vous pouvez spécifier un modèle de lancement Amazon EC2 dans vos paramètres de distribution, en utilisant. `launchTemplateConfigurations` Lorsqu'ils `launchTemplateConfigurations` sont présents pendant le processus de distribution, Image Builder crée une nouvelle version du modèle de lancement qui inclut tous les paramètres d'origine du modèle et le nouvel ID d'AMI issu de la compilation. Pour plus d'informations sur le lancement d'une instance EC2 à l'aide d'un modèle de lancement, consultez l'un des liens suivants, en fonction de votre système d'exploitation cible.

- Lancer une instance Linux à partir d'un modèle de lancement
- Lancer une instance Windows à partir d'un modèle de lancement

> **Note**
>
> Lorsque vous incluez un modèle de lancement pour activer Windows Fast Launch dans votre image, le modèle de lancement doit inclure la balise suivante afin qu'Image Builder puisse activer Windows Fast Launch en votre nom.
> `CreatedBy: EC2 Image Builder`

Ajouter un modèle de lancement EC2 aux paramètres de distribution de l'AMI depuis la console

Pour fournir un modèle de lancement avec votre AMI de sortie, procédez comme suit dans la console :

1. Ouvrez la console https://console.aws.amazon.com/imagebuilder/EC2 Image Builder à l'adresse.
2. Choisissez Paramètres de distribution dans le volet de navigation. Cela affiche la liste des paramètres de distribution créés sous votre compte.
3. En haut de la page des paramètres de distribution, choisissez Créer des paramètres de distribution. Cela ouvre la page Créer des paramètres de distribution.
4. Dans la section Type d'image, choisissez le type de sortie Amazon Machine Image (AMI). Il s'agit du paramètre par défaut.

5. Dans la section Général, entrez le nom de la ressource de paramètres de distribution que vous souhaitez créer (obligatoire).
6. Dans la section Paramètres régionaux, sélectionnez le nom d'un modèle de lancement EC2 dans la liste. Si votre compte ne contient aucun modèle de lancement, choisissez Créer un nouveau modèle de lancement, qui ouvre les modèles de lancement dans le tableau de bord EC2.

   Cochez la case Définir la version par défaut pour mettre à jour la version par défaut du modèle de lancement vers la nouvelle version créée par Image Builder avec votre AMI de sortie.

   Pour ajouter un autre modèle de lancement à la région sélectionnée, choisissez Ajouter une configuration de modèle de lancement.

   Pour supprimer un modèle de lancement, choisissez Supprimer.
7. Continuez à spécifier les paramètres supplémentaires dont vous avez besoin, puis choisissez Créer des paramètres pour créer votre nouvelle ressource de paramètres de distribution.

Ajoutez un modèle de lancement EC2 aux paramètres de distribution de l'AMI à partir du AWS CLI

Cette section décrit comment configurer un fichier de paramètres de distribution avec un modèle de lancement et utiliser la create-image commande contenue dans le AWS CLI pour créer et distribuer une AMI Image Builder et une nouvelle version du modèle de lancement qui l'utilise.

1. Configuration d'un fichier de paramètres de distribution

   Avant de créer une AMI Image Builder avec un modèle de lancement AWS CLI, vous devez créer une structure JSON de configuration de distribution qui spécifie les `launchTemplateConfigurations` paramètres. Vous devez spécifier au moins une `launchTemplateConfigurations` entrée dans la région source.

   Le fichier d'exemple suivant, nommé`create-distribution-config-launch-template.json`, montre quelques scénarios possibles pour la configuration du modèle de lancement dans la région source.

```
{
    "name": "NewDistributionConfiguration",
    "description": "This is just a test",
    "distributions": [
        {
```

```
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "test-{{imagebuilder:buildDate}}-
{{imagebuilder:buildVersion}}",
                "description": "description"
            },
            "launchTemplateConfigurations": [
                {
                    "launchTemplateId": "lt-0a1bcde2fgh34567",
                    "accountId": "935302948087",
                    "setDefaultVersion": true
                },
                {
                    "launchTemplateId": "lt-0aaa1bcde2ff3456"
                },
                {
                    "launchTemplateId": "lt-12345678901234567",
                    "accountId": "123456789012"
                }
            ]
        }
    ],
    "clientToken": "clientToken1"
}
```

2. Création des paramètres de distribution

   Pour créer une ressource de paramètres de distribution Image Builder à l'aide de la create-distribution-configurationcommande AWS CLI, entrez les paramètres suivants dans la commande :

   - Entrez le nom de la distribution dans le `--name` paramètre.
   - Joignez le fichier JSON de configuration de distribution que vous avez créé dans le `--cli-input-json` paramètre.

```
aws imagebuilder create-distribution-configuration --name my distribution name--
cli-input-json file://create-distribution-config-launch-template.json
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

Vous pouvez également fournir du JSON directement dans la commande, à l'aide du `--distributions` paramètre.

Utilisez des fonctionnalités de distribution d'AMI améliorées

Image Builder propose des fonctionnalités de distribution avancées qui vous offrent flexibilité et contrôle sur la façon dont vos AMI sont distribuées entre les régions et les comptes. Ces fonctionnalités séparent la distribution du processus de création, ce qui vous permet de distribuer les images existantes à la demande, de vous remettre efficacement en cas d'échec de distribution et de mettre en œuvre des stratégies de distribution contrôlées en plusieurs étapes via des flux de travail personnalisables.

Vous pouvez utiliser les fonctionnalités de distribution AMI améliorées d'Image Builder pour effectuer directement des activités de distribution sans avoir à réexécuter une génération d'image complète.

Distribution découplée

L' DistributeImage API accepte trois types de références d'image source :

- ID AMI : identifiant AMI standard (par exemple,`ami-0abcdef1234567890`)
- Paramètre SSM : paramètre SSM qui stocke un ID AMI (par exemple,) `ssm:/my/ami/parameter`
- ARN de la version d'Image Builder — Un ARN de version d'image d'Image Builder

Nouvelle tentative de distribution

Si la distribution d'une image échoue, utilisez l'`RetryImage`API pour réessayer de la distribuer. Cela réduit le temps nécessaire pour résoudre la cause de la panne en évitant une reconstruction

complète de l'image. À utiliser `RetryImage` après avoir résolu la cause sous-jacente de l'échec de distribution.

L' RetryImage API accepte un ARN de version de génération d'image (par exemple,`arn:aws:imagebuilder:us-west-2:123456789012:image/my-image/1.0.0/1`). Lorsque vous appelez l'API, Image Builder reprend automatiquement le processus de distribution à partir du point d'échec en utilisant la configuration et les paramètres de distribution d'origine. L'`RetryImage`API peut réessayer des distributions qui ont échoué pendant la phase de distribution, la phase de test ou la phase d'intégration. Il fonctionne avec les AMI dans les états suivants : en attente, en échec, supprimée ou disponible.

Conditions préalables

Avant de réessayer une distribution, assurez-vous de ce qui suit :

- Vous avez identifié et résolu la cause première de l'échec. Consultez les journaux de distribution dans CloudWatch Logs pour plus de détails sur les erreurs.
- Vous disposez des autorisations IAM nécessaires pour réessayer de générer l'image.
- En cas d'échec de distribution entre comptes, vérifiez que `EC2ImageBuilderDistributionCrossAccountRole` la `Ec2ImageBuilderCrossAccountDistributionAccess` politique est associée au compte de destination.

Important : Réessayer sans résoudre le problème sous-jacent entraînera des échecs répétés.

Workflows de distribution

Les flux de travail de distribution sont un nouveau type de flux de travail qui complète les flux de création et de test, vous permettant de définir et de contrôler le processus de distribution par étapes séquentielles. Avec les flux de distribution, vous pouvez créer des processus de distribution personnalisés qui incluent des opérations de copie AMI, des points de contrôle en attente d'action, des modifications des attributs d'image et d'autres étapes liées à la distribution. Cela fournit un contrôle structuré sur la façon dont vos AMI sont distribuées, avec une visibilité progressive, des fonctionnalités de distribution parallèle et des rapports d'erreurs granulaires.

Pour en savoir plus sur la création et la personnalisation de flux de travail, voir Gérer les flux de travail liés aux images.

Partagez les ressources d'Image Builder avec AWS RAM

EC2 Image Builder s' AWS Resource Access Manager intègre àAWS RAM() afin que vous puissiez partager les types de ressources d'Image Builder suivants avec Compte AWS n'importe qui ou AWS Organizations via.

- Éléments
- Images
- Recettes

Pour partager des ressources AWS RAM, vous devez créer un partage de ressources. Un partage de ressources indique les ressources à partager et les consommateurs avec qui les partager. Les consommateurs peuvent être des individus Comptes AWS, des unités organisationnelles ou une organisation entière AWS Organizations. La liste suivante inclut les types de comptes et d'organisations avec lesquels vous pouvez partager.

- Spécifique Comptes AWS à l'intérieur ou à l'extérieur de son organisation dans AWS Organizations.
- Une unité organisationnelle (UO) au sein de son organisation dans AWS Organizations.
- L'ensemble de son organisation dans AWS Organizations.
- AWS Organizations ou OUs en dehors de son organisation.

Dans ce modèle, le Compte AWS propriétaire de la ressource (propriétaire) la partage avec d'autres personnes Comptes AWS ou par l'intermédiaire AWS Organizations (consommateurs) au sein de la même région. Lorsqu'une ressource partagée est mise à jour, les consommateurs reçoivent ces mises à jour automatiquement.

> **Note**
>
> Les composants, images et recettes d'images partagés ne sont pris en compte que dans les limites de ressources correspondantes du propriétaire. Les limites de ressources des consommateurs ne sont pas affectées par les ressources partagées avec eux.

Rubriques

- Propriétaires des ressources
- Consommateurs de ressources
- Créez un partage de AWS RAM ressources pour vos ressources Image Builder
- Annuler le partage d'une ressource Image Builder depuis AWS RAM

Propriétaires des ressources

Les ressources Image Builder ne peuvent être partagées que Région AWS là où elles ont été créées. Lorsque vous partagez ces ressources, elles ne seront pas répliquées d'une région à l'autre.

Pour obtenir la liste des ressources Image Builder que vous possédez et que vous pouvez partager, spécifiez le filtre de propriété dans la console ou lorsque vous exécutez la commande dans le AWS CLI.

- Liste des composants d'Image Builder
- Liste des images
- Répertorier et afficher les détails des recettes avec des images
- Répertorier et afficher les détails des recettes des contenants

Pour plus d'informations AWS RAM, consultez le guide de AWS RAM l'utilisateur.

Conditions préalables au partage des ressources Image Builder

Pour partager une ressource Image Builder, telle qu'un composant, une image ou une recette :

- Vous Compte AWS devez être propriétaire de la ressource Image Builder que vous souhaitez partager. Vous ne pouvez pas partager les ressources qui ont été partagées avec vous.
- La clé AWS Key Management Service (AWS KMS) associée aux ressources chiffrées doit être explicitement partagée avec les comptes cibles, les organisations ou OUs.
- Pour partager vos ressources Image Builder AWS Organizations et les OUs utiliser AWS RAM, vous devez activer le partage. Pour plus d'informations, consultez Activation du partage avec AWS Organizations dans le Guide de l'utilisateur AWS RAM .
- Si vous distribuez une image chiffrée AWS KMS sur plusieurs comptes dans différentes régions, vous devez créer une clé KMS et un alias dans chaque région cible. En outre, les personnes qui lanceront des instances dans ces régions devront avoir accès à la clé KMS spécifiée dans la politique clé.

Les ressources suivantes créées par Image Builder à partir de votre pipeline ne sont pas considérées comme des ressources Image Builder. Il s'agit plutôt de ressources externes qu'Image Builder distribue dans votre compte Régions AWS, ainsi qu'aux comptes et aux organisations ou unités organisationnelles (OUs) que vous spécifiez dans votre configuration de distribution.

- Images de machines Amazon (AMIs)
- Images de conteneurs résidant dans Amazon ECR

Pour plus d'informations sur les paramètres de distribution de votre AMI, consultezCréation et mise à jour des configurations de distribution AMI. Pour plus d'informations sur les paramètres de distribution de votre image de conteneur dans Amazon ECR, consultezCréation et mise à jour des paramètres de distribution pour les images de conteneurs.

Pour plus d'informations sur le partage de votre AMI avec AWS Organizations et OUs, voir Partager une AMI avec des organisations ou OUs.

Consommateurs de ressources

Les consommateurs peuvent utiliser une ressource partagée, mais ne peuvent en aucun cas la modifier. Lorsqu'ils créent des recettes Image Builder, ils peuvent spécifier une image partagée comme image de base et ajouter des composants partagés. Ils peuvent également spécifier une recette partagée lorsqu'ils créent un pipeline d'images Image Builder ou lorsqu'ils utilisent la create-image commande du AWS CLI.

Si vous appartenez à une organisation et que le partage au sein de votre organisation est activé, les consommateurs de votre organisation ont automatiquement accès à la ressource partagée. AWS Organizations Dans le cas contraire, les consommateurs reçoivent une invitation à rejoindre le partage de ressources et ont accès à la ressource partagée après avoir accepté l'invitation.

Créez un partage de AWS RAM ressources pour vos ressources Image Builder

Pour partager un composant, une image ou une recette Image Builder, vous devez l'ajouter à un partage de AWS Resource Access Manager ressources. Le partage des ressources indique les ressources à partager et les consommateurs avec lesquels elles sont partagées.

Les options suivantes sont disponibles pour partager vos ressources.

Option 1 : créer un partage de ressources RAM

Lorsque vous créez un partage de ressources RAM, vous pouvez partager un composant, une image ou une recette que vous possédez en une seule étape. Utilisez l'une des méthodes suivantes pour créer votre partage de ressources :

- Console

  Pour créer votre partage de ressources à l'aide de la AWS RAM console, voir Partager AWS les ressources qui vous appartiennent dans le guide de AWS RAM l'utilisateur.
- AWS CLI

  Pour créer votre partage de ressources à l'aide de l'interface de ligne de AWS RAM commande, exécutez la create-resource-share commande dans le AWS CLI.

Option 2 : appliquer une politique de ressources et promouvoir vers un partage de ressources existant

La deuxième option pour partager vos ressources implique deux étapes, en exécutant des commandes dans AWS CLI les deux cas. La première étape utilise les commandes Image Builder AWS CLI pour appliquer des politiques basées sur les ressources à la ressource partagée. La deuxième étape fait passer la ressource à un partage de ressources RAM à l'aide de la promote-resource-share-created-from-policy AWS RAM commande contenue dans le AWS CLI afin de garantir que la ressource est visible par tous les principaux avec lesquels vous l'avez partagée.

1. Appliquer la politique en matière de ressources

   Pour appliquer correctement la politique de ressources, vous devez vous assurer que le compte avec lequel vous partagez est autorisé à accéder à toutes les ressources sous-jacentes.

   Choisissez l'onglet correspondant à votre type de ressource pour la commande applicable.

   Image

   Vous pouvez appliquer une politique de ressources à une image afin de permettre à d'autres utilisateurs de l'utiliser comme image de base dans leurs recettes.

   Exécutez la commande put-image-policy Image Builder dans le AWS CLI, pour identifier les AWS principaux acteurs avec lesquels partager l'image.

```
aws imagebuilder put-image-policy --image-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.03/1 --policy '{ "Version":
 "2012-10-17",        "Statement": [ { "Effect": "Allow", "Principal":
 { "AWS": [ "123456789012" ] }, "Action": ["imagebuilder:GetImage",
 "imagebuilder:ListImages"], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.03/1" ] } ] }'
```

Component

Vous pouvez appliquer une politique de ressources à un composant de génération ou de test pour permettre le partage entre comptes. Cette commande autorise les autres comptes à utiliser votre composant dans leurs recettes. Pour appliquer correctement la politique de ressources, vous devez vous assurer que le compte avec lequel vous partagez est autorisé à accéder à toutes les ressources référencées par le composant partagé, telles que les fichiers hébergés sur des référentiels privés.

Exécutez la commande put-component-policy Image Builder dans le AWS CLI, pour identifier les AWS principaux acteurs avec lesquels partager le composant.

```
aws imagebuilder put-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/my-example-component/2019.12.03/1 --
policy '{ "Version": "2012-10-17",        "Statement": [ { "Effect":
 "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetComponent", "imagebuilder:ListComponents" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-
component/2019.12.03/1" ] } ] }'
```

Image recipe

Vous pouvez appliquer une politique de ressources à une recette d'image pour permettre le partage entre comptes. Cette commande autorise les autres comptes à utiliser votre recette pour créer des images dans leurs comptes. Pour appliquer correctement la politique de ressources, vous devez vous assurer que le compte avec lequel vous partagez est autorisé à accéder à toutes les ressources auxquelles la recette fait référence, telles que l'image de base ou les composants sélectionnés.

Exécutez la commande put-image-recipe-policy Image Builder dans le AWS CLI, pour identifier les AWS principaux acteurs avec lesquels partager l'image.

```
aws imagebuilder put-image-recipe-policy --image-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-
recipe/2019.12.03 --policy '{ "Version": "2012-10-17",        "Statement":
 [ { "Effect": "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetImageRecipe", "imagebuilder:ListImageRecipes" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-
recipe/2019.12.03" ] } ] }'
```

Container recipe

Vous pouvez appliquer une politique de ressources à une recette de conteneur pour permettre le partage entre comptes. Cette commande autorise les autres comptes à utiliser votre recette pour créer des images dans leurs comptes. Pour appliquer correctement la politique de ressources, vous devez vous assurer que le compte avec lequel vous partagez est autorisé à accéder à toutes les ressources auxquelles la recette fait référence, telles que l'image de base ou les composants sélectionnés.

Exécutez la commande put-container-recipe-policy Image Builder dans le AWS CLI, pour identifier les AWS principaux acteurs avec lesquels partager l'image.

```
aws imagebuilder put-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-
example-container-recipe/2021.12.03 --policy '{ "Version": "2012-10-17",
        "Statement": [ { "Effect": "Allow", "Principal": { "AWS":
 [ "123456789012" ] }, "Action": [ "imagebuilder:GetContainerRecipe",
 "imagebuilder:ListContainerRecipes" ], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:container-recipe/my-example-container-
recipe/2021.12.03" ] } ] }'
```

> **Note**
>
> Pour définir les bonnes politiques de partage et de non-partage d'une ressource, le propriétaire de la ressource doit disposer des `imagebuilder:put*` autorisations nécessaires.

2. Promouvez en tant que partage de ressources RAM

   Pour que la ressource soit visible par tous les principaux avec lesquels vous l'avez partagée, exécutez la promote-resource-share-created-from-policy AWS RAM commande dans le AWS CLI.

Annuler le partage d'une ressource Image Builder depuis AWS RAM

Pour annuler le partage d'une ressource Image Builder dont vous êtes propriétaire, telle qu'un composant, une image ou une recette partagés, vous devez la supprimer du partage de AWS Resource Access Manager ressources. Pour ce faire, vous pouvez utiliser soit la console AWS RAM , soit l' AWS CLI.

> **Note**
>
> Les propriétaires ne peuvent pas supprimer une ressource partagée tant qu'elle n'est plus partagée. Un propriétaire ne peut pas annuler le partage de ces ressources tant qu'aucun consommateur n'en dépend.

Pour annuler le partage d'un composant, d'une image ou d'une recette partagés dont vous êtes propriétaire à l'aide de la console AWS Resource Access Manager

Consultez Mise à jour d'un partage de ressources dans le Guide de l'utilisateur AWS RAM .

Pour annuler le partage d'un composant, d'une image ou d'une recette partagés dont vous êtes propriétaire à l'aide du AWS CLI

Utilisez la disassociate-resource-share commande pour arrêter de partager la ressource.

Ressources de sortie de Tag Image Builder

Le balisage de vos ressources peut être utile pour filtrer et suivre les coûts des ressources, ou pour d'autres catégories. Vous pouvez également contrôler l'accès en fonction des balises. Pour plus d'informations sur l'autorisation basée sur des balises, voir Autorisation basée sur les balises Image Builder

Image Builder prend en charge les balises dynamiques suivantes :

- `- {{imagebuilder:buildDate}}`

  Résout à la construction date/time au moment de la construction.
- `- {{imagebuilder:buildVersion}}`

  Résout à une version de build, qui est un numéro situé à la fin du nom de ressource Amazon (ARN) d'Image Builder. Par exemple, `"arn:aws:imagebuilder:us-west-2:123456789012:component/myexample-component/2019.12.02/1"` affiche la version de compilation sous la forme1.

Pour vous aider à suivre les Amazon Machine Images (AMIs) que vous avez distribuées, Image Builder ajoute automatiquement les balises suivantes à votre sortie AMIs.

- `"CreatedBy":"EC2 Image Builder"`
- `"Ec2ImageBuilderArn":"arn:aws:imagebuilder:`*`us-west-2`*`:`*`123456789012`*`:image/`*`simple-recipe-linux`*`/`*`1.0.0/10`*`"`. Cette balise contient l'ARN de la ressource d'image Image Builder qui a été utilisée pour créer l'AMI.

Table des matières

- Marquez une ressource depuis AWS CLI
- Supprimer le tag d'une ressource du AWS CLI
- Répertoriez toutes les balises d'une ressource spécifique à partir du AWS CLI

Marquez une ressource depuis AWS CLI

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour ajouter et étiqueter une ressource dans EC2 Image Builder. Vous devez fournir les balises `resourceArn` et à appliquer.

Le `tag-resource.json` contenu de l'exemple est le suivant :

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-example-pipeline",
    "tags": {
        "KeyName": "KeyValue"
    }
}
```

Exécutez la commande suivante, qui fait référence au `tag-resource.json` fichier précédent.

```
aws imagebuilder tag-resource --cli-input-json file://tag-resource.json
```

Supprimer le tag d'une ressource du AWS CLI

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour supprimer une balise d'une ressource. Vous devez fournir les clés `resourceArn` et pour retirer le tag.

Le `untag-resource.json` contenu de l'exemple est le suivant :

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-example-pipeline",
    "tagKeys": [
        "KeyName"
    ]
}
```

Exécutez la commande suivante, qui fait référence au `untag-resource.json` fichier précédent.

```
aws imagebuilder untag-resource --cli-input-json file://untag-resource.json
```

Répertoriez toutes les balises d'une ressource spécifique à partir du AWS CLI

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour répertorier toutes les balises d'une ressource spécifique.

```
aws imagebuilder list-tags-for-resource --resource-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

Supprimer les ressources d'Image Builder obsolètes ou inutilisées

Votre environnement Image Builder, tout comme votre maison, a besoin d'un entretien régulier pour vous aider à trouver ce dont vous avez besoin et à accomplir vos tâches sans vous encombrer. Assurez-vous de nettoyer régulièrement les ressources temporaires que vous avez créées pour les tests. Sinon, vous pourriez oublier ces ressources et, plus tard, ne plus vous souvenir de leur utilisation. D'ici là, il se peut que vous ne sachiez pas si vous pouvez vous en débarrasser en toute sécurité.

La suppression de ressources n'entraîne pas la suppression des AMI Amazon EC2 ou des images de conteneur Amazon ECR créées pendant le processus de création de l'image. Vous devez les nettoyer séparément, à l'aide des actions de console Amazon EC2 ou Amazon ECR appropriées, de l'API ou des commandes. AWS CLI

> **Tip**
>
> Pour éviter les erreurs de dépendance lorsque vous supprimez des ressources, veillez à supprimer vos ressources dans l'ordre suivant :
>
> 1. Pipeline d'images
> 2. Recette d'images
> 3. Toutes les ressources restantes

Supprimer des ressources du Console de gestion AWS

Pour supprimer un pipeline d'images et ses ressources, procédez comme suit :

Supprimer le pipeline

1. Pour voir la liste des pipelines de génération créés sous votre compte, choisissez Image pipelines dans le volet de navigation.
2. Cochez la case à côté du nom du pipeline pour sélectionner le pipeline que vous souhaitez supprimer.
3. En haut du panneau Pipelines d'images, dans le menu Actions, choisissez Supprimer.

4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer la recette

1. Pour voir la liste des recettes créées sous votre compte, choisissez Image recipes dans le volet de navigation.
2. Cochez la case à côté du nom de la recette pour sélectionner la recette que vous souhaitez supprimer.
3. En haut du panneau des recettes d'images, dans le menu Actions, choisissez Supprimer la recette.
4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer la configuration de l'infrastructure

1. Pour consulter la liste des configurations d'infrastructure créées sous votre compte, choisissez Configuration de l'infrastructure dans le volet de navigation.
2. Cochez la case à côté du nom de la configuration pour sélectionner la configuration d'infrastructure que vous souhaitez supprimer.
3. En haut du panneau de configuration de l'infrastructure, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer les paramètres de distribution

1. Pour voir la liste des paramètres de distribution créés sous votre compte, choisissez Paramètres de distribution dans le volet de navigation.
2. Cochez la case à côté du nom de la configuration pour sélectionner les paramètres de distribution que vous avez créés pour ce didacticiel.
3. En haut du panneau des paramètres de distribution, choisissez Supprimer.
4. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprime une image

1. Pour voir la liste des images créées sous votre compte, choisissez Images dans le volet de navigation.

2. Choisissez la version de l'image que vous souhaitez supprimer. Cela ouvre la page des versions de génération d'images.
3. Cochez la case à côté de la version pour toute image que vous souhaitez supprimer. Vous pouvez sélectionner plusieurs versions d'image à la fois.
4. En haut du panneau des versions de génération d'images, choisissez Supprimer la version.
5. Pour confirmer la suppression, entrez `Delete` dans le champ et choisissez Supprimer.

Supprimer un pipeline d'images du AWS CLI

Les exemples suivants montrent comment supprimer des ressources Image Builder à l'aide du AWS CLI. Comme indiqué précédemment, les ressources doivent être supprimées dans l'ordre suivant pour éviter les erreurs de dépendance :

1. Pipeline d'images
2. Recette d'images
3. Toutes les ressources restantes

Supprimer le pipeline d'images du AWS CLI

L'exemple suivant montre comment supprimer un pipeline d'images en spécifiant son ARN.

```
aws imagebuilder delete-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

Supprimer la recette d'image du AWS CLI

L'exemple suivant montre comment supprimer une recette d'image en spécifiant son ARN.

```
aws imagebuilder delete-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2019.12.03
```

Supprime une configuration d'infrastructure

L'exemple suivant montre comment supprimer une ressource de configuration d'infrastructure en spécifiant son ARN.

```
aws imagebuilder delete-infrastructure-configuration --infrastructure-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-
infrastructure-configuration
```

Supprimer les paramètres de distribution

L'exemple suivant montre comment supprimer une ressource de paramètres de distribution en spécifiant son ARN.

```
aws imagebuilder delete-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

Supprime une image

L'exemple suivant montre comment supprimer une version de génération d'image en spécifiant son ARN.

```
aws imagebuilder delete-image --image-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.02/1
```

Supprime un composant

L'exemple suivant montre comment utiliser une commande imagebuilder CLI pour supprimer une version de construction d'un composant en spécifiant son ARN.

```
aws imagebuilder delete-component --component-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-
component/2019.12.02/1
```

> **Important**
>
> Assurez-vous qu'aucune recette ne fait référence à la version de construction du composant de quelque manière que ce soit avant de la supprimer. Si vous ne le faites pas, cela pourrait entraîner des défaillances du pipeline.

Gérez les flux de travail de création, de test et de distribution pour les images Image Builder

Un flux de travail d'image définit la séquence d'étapes qu'EC2 Image Builder exécute au cours d'une étape du processus de création d'image. Vous créez un flux de travail sous forme de document YAML et vous le créez en tant que ressource versionnée et réutilisable. Image Builder exécute ensuite vos flux de travail lorsqu'il crée, teste et distribue vos images.

Les flux de travail vous permettent de contrôler directement le processus de création d'images. Vous décidez quelles étapes s'exécutent, dans quel ordre, ce que fait chaque étape en cas d'échec et si Image Builder annule une étape échouée. Vous pouvez partir d'un flux de travail géré par Amazon, le cloner et le personnaliser, ou créer le vôtre en partant de zéro.

Avantages du flux de travail d'imagerie

- Grâce aux flux de travail d'images, vous bénéficiez d'une flexibilité, d'une visibilité et d'un contrôle accrus sur le processus de création d'images.
- Vous pouvez ajouter des étapes de flux de travail personnalisées lorsque vous définissez votre document de flux de travail, ou vous pouvez choisir d'utiliser le flux de travail par défaut d'Image Builder.
- Vous pouvez exclure les étapes du flux de travail qu'Image Builder inclut dans les flux de travail d'imagerie par défaut.
- Vous pouvez créer des flux de travail de test uniquement qui ignorent complètement le processus de génération. Vous pouvez faire de même pour créer des flux de travail uniquement liés à la création ou à la distribution.

> **Note**
>
> Vous ne pouvez pas modifier un flux de travail existant, mais vous pouvez le cloner ou créer une nouvelle version.

> **Tip**
>
> Pour plus d'informations sur les exemples de flux de travail déployables, y compris les portes d'approbation et AWS Step Functions l'intégration, consultez les exemples de flux de travail

sur GitHub. Pour de plus amples informations, veuillez consulter Découvrez des exemples de projets Image Builder sur GitHub.

Sujets relatifs aux flux

- Structure du flux de travail : étapes
- Fonctionnement d'un flux de travail
- Surveillez l'exécution du flux de travail avec Amazon EventBridge
- Contraintes liées aux ressources du flux
- Accès au service
- Utilisez des flux de travail gérés pour vos images
- Répertorier les flux de
- Création d'un flux de travail d'images
- Création d'un document de flux de travail YAML

Structure du flux de travail : étapes

Image Builder organise le processus de création d'images en trois étapes. Chaque étape exécute un flux de travail du type correspondant. Les étapes se déroulent dans un ordre fixe et une étape ne commence que lorsque l'étape précédente s'est terminée avec succès.

Le tableau suivant répertorie le comportement par défaut de chaque étape. Comme vous définissez les étapes de chaque flux de travail, vous pouvez personnaliser le fonctionnement d'une étape.

Étapes du flux de travail Image Builder

Ordre	Étape	Type de flux de travail	Ce qu'il fait (par défaut)
1	Build	`BUILD`	Lance une instance de génération, exécute vos composants de génération et crée une image (AMI ou image de conteneur) à partir de l'instance.
2	Test	`TEST`	Lance une instance de test à partir de la nouvelle image, exécute vos composants de

Ordre	Étape	Type de flux de travail	Ce qu'il fait (par défaut)
			test et collecte éventuellement les résultats de numérisation de l'image.
3	Distribution	`DISTRIBUTION`	Copie l'image dans les régions et les comptes cibles, modifie les attributs de l'image et applique des configurations post-distribution telles que des modèles de lancement et des configurations de licence.

Vous ne pouvez exécuter un flux de travail que pour certaines étapes. Par exemple, vous pouvez ignorer l'étape de génération pour tester ou distribuer une image qui existe déjà, ou ignorer l'étape de test pour créer et distribuer sans tester. Pour ignorer la phase de création ou de test, omettez un flux de travail de ce type dans votre pipeline ou votre demande d'image.

> **Note**
>
> Le flux de travail de distribution est facultatif. Si vous l'omettez, Image Builder n'ignore pas la distribution. Il distribue tout de même votre AMI en exécutant la configuration de distribution que vous attachez au pipeline ou à l'image. Ajoutez un flux de distribution lorsque vous souhaitez remplacer cette configuration de distribution par d'autres paramètres de distribution, ou pour obtenir plus de visibilité sur le processus de distribution. Pour ignorer complètement la distribution, fournissez une configuration de distribution nulle ou vide.

> **Note**
>
> Un pipeline d'images ou une demande d'image peut inclure au plus un flux de travail de création et un flux de travail de distribution. Il peut également inclure un ou plusieurs flux de travail de test. Le nombre total de flux de travail ne peut pas dépasser dix. Par exemple, vous pouvez associer un flux de création et un flux de distribution avec jusqu'à huit flux de travail de test.

Fonctionnement d'un flux de travail

Un flux de travail exécute ses étapes dans l'ordre dans lequel vous les avez définies dans le document YAML. Chaque étape exécute une action, telle que le lancement d'une instance ou l'exécution de composants.

Résultats de l'étape

Chaque étape se termine dans l'un des états suivants.

Résultats des étapes du flux de travail

State	Signification
`COMPLETED`	L'étape s'est terminée avec succès.
`SKIPPED`	Une `if` déclaration conditionnelle sur l'étape évaluée à`false`, Image Builder a donc ignoré l'étape.
`FAILED`	L'étape ne s'est pas terminée correctement.
`WAITING`	Une `WaitForAction` étape est suspendue, en attente d'une action externe.
`TIMED_OUT`	Le pas a duré plus longtemps que sa `timeoutSeconds` valeur.

Gestion des défaillances

L'`onFailure`attribut de chaque étape contrôle ce qui se passe en cas d'échec d'une étape.

- `Abort`(par défaut) — Image Builder échoue à l'étape, échoue au flux de travail et n'exécute aucune autre étape. Si la restauration est activée, elle commence à l'étape qui a échoué et revient aux étapes précédentes.
- `Continue`— Image Builder échoue à l'étape mais exécute les étapes restantes. Aucun retour en arrière ne se produit.

Restauration

Lorsqu'une étape échoue avec `onFailure: Abort` et `rollbackEnabled: true` (valeur par défaut), Image Builder annule les étapes terminées dans l'ordre inverse, en commençant par l'étape

qui a échoué. Toutes les actions ne peuvent pas être annulées. Une action qui n'a pas de retour en arrière enregistre un statut de`NO_ROLLBACK`.

Surveillez l'exécution du flux de travail avec Amazon EventBridge

Image Builder émet des EventBridge événements Amazon au fur et à mesure de l'exécution de vos flux de travail, afin que vous puissiez réagir à la progression et aux étapes nécessitant une attention particulière. Par exemple, une `WaitForAction` étape publie un événement avec le type de détail dans votre bus `EC2 Image Builder Workflow Step Waiting` d'événements par défaut. Vous pouvez acheminer cet événement vers une cible telle qu'une fonction Lambda ou une rubrique Amazon SNS afin de piloter un processus d'approbation. Pour plus d'informations sur le fonctionnement d'Image Builder EventBridge, consultezEventBridge Intégration d'Amazon dans Image Builder.

Pour suivre la progression au niveau des étapes par programmation, utilisez les opérations`ListWorkflowExecutions`, `GetWorkflowExecutionListWorkflowStepExecutions`, et `GetWorkflowStepExecution` API.

Contraintes liées aux ressources du flux

Vos documents et configurations de flux de travail doivent respecter les limites par défaut suivantes.

Contraintes liées aux ressources du flux

Contrainte	Limite par défaut	Remarques
Étapes par document de flux de travail	15	Les étapes s'exécutent dans l'ordre des documents.
Sorties par document de flux de travail	25	Les sorties transmettent des valeurs aux flux de travail ultérieurs.
Paramètres par document de flux de travail	25	Entrées que l'appelant peut définir.
Longueur de la valeur du paramètre	1,024 caractères	Par valeur de paramètre.

Contrainte	Limite par défaut	Remarques
Taille du document en ligne () `data`	16 000 octets	Utilisez l'option `uri` (Amazon S3) pour les documents plus volumineux.
Flux de travail par image ou pipeline	10	Au plus 1 build et 1 distribution, plus des workflows de test, pour un total combiné de 10.

Les ressources du flux de travail sont immuables. Pour modifier un flux de travail, créez une nouvelle version ou clonez-la. Image Builder conserve toutes les versions, afin que vous puissiez suivre le flux de travail qui a produit chaque image.

Accès au service

Pour exécuter des flux de travail liés à l'imagerie, Image Builder doit être autorisé à effectuer des actions de flux de travail. Vous accordez cette autorisation avec un rôle d'exécution qu'Image Builder assume en votre nom. Vous attribuez le rôle d'exécution comme suit.

> **Important**
>
> Nous vous recommandons de ne pas transmettre le rôle AWSServiceRoleForImageBuilder lié à un service en tant que rôle d'exécution. Créez plutôt un rôle IAM personnalisé et associez la politique EC2ImageBuilderExecutionPolicy AWS gérée. Cette politique accorde les mêmes autorisations dont Image Builder a besoin pour appeler Services AWS en votre nom. L'utilisation d'un rôle personnalisé vous permet de contrôler totalement les autorisations utilisées par Image Builder. Il permet également de maintenir vos politiques de contrôle des services (SCP) et vos politiques de contrôle des ressources (RCP) en vigueur pour les opérations qu'Image Builder effectue en votre nom.

- Console : dans l'assistant de pipeline Étape 3 Définissez le processus de création d'images, sélectionnez votre rôle personnalisé dans la liste des rôles IAM du panneau d'accès au service.
- API Image Builder : dans la demande CreateImage d'action, spécifiez votre rôle personnalisé comme valeur du `executionRole` paramètre.

Pour créer un rôle d'exécution personnalisé, consultez la section Création d'un rôle pour déléguer des autorisations à un AWS service dans le Guide de Gestion des identités et des accès AWS l'utilisateur.

Utilisez des flux de travail gérés pour vos images

AWS crée et gère des flux de travail gérés. Lorsque vous utilisez des flux de travail gérés dans vos pipelines d'images ou pour la création d'images ponctuelles, vous pouvez sélectionner le nom de ressource Amazon (ARN) du flux de travail géré que vous souhaitez utiliser. Amazon fournit les dernières versions auxquelles des correctifs et autres mises à jour ont été appliqués. Pour obtenir la liste des flux de travail gérésRépertorier les flux de, consultez et filtrez sur Owner = Amazon (console).

Vous pouvez choisir un flux de travail géré en fonction de votre vitesse de création et de vos exigences de validation. Voici des exemples de types de flux de travail gérés :

Workflows gérés standard (par défaut)

Les flux de travail gérés standard incluent des étapes complètes pour créer, tester et valider vos images avec des contrôles d'état EC2 complets.

Les flux de travail gérés standard sont les suivants :

- build-image — Workflow de création d' Amazon-managed AMI par défaut.
- test-image — Workflow de test Amazon-managed AMI par défaut.
- build-container — Workflow de création de Amazon-managed conteneurs par défaut.
- test-container — Workflow de test de Amazon-managed conteneur par défaut.

Flux de travail gérés Express

Les flux de travail gérés par Express incluent uniquement les étapes essentielles et réduisent le temps de création d'images.

Les flux de travail gérés en mode express sont les suivants :

- express-build-image — Workflow de création d'Express Amazon-managed AMI qui réduit le temps de génération. Attend uniquement l'état d'exécution de l'instance au lieu des vérifications complètes de l'état EC2, et ignore la collecte de métadonnées d'image.
- express-build-container — Flux de travail de création de Amazon-managed conteneurs Express qui réduit le temps de génération. Attend uniquement l'état d'exécution de l'instance au lieu de vérifications complètes de l'état EC2.

- express-test-image — Workflow de test Express Amazon-managed AMI qui réduit la durée des tests. Attend uniquement l'état d'exécution de l'instance au lieu de procéder à des vérifications complètes de l'état EC2, et ignore la collecte des résultats des scans de sécurité.

Répertorier les flux de

Sur la page de liste des flux de travail d'imagerie de la console Image Builder, vous pouvez obtenir la liste des ressources de flux de travail d'image que vous possédez ou auxquelles vous avez accès, ainsi que certains détails clés sur ces ressources. Vous pouvez également utiliser des commandes ou des actions avec l'API Image Builder, les kits SDK ou AWS CLI pour répertorier les flux de travail liés à l'image dans votre compte.

Vous pouvez utiliser l'une des méthodes suivantes pour répertorier les ressources de flux d'images que vous possédez ou auxquelles vous avez accès. Pour l'action de l'API, voir le ListWorkflows document de référence de l'API EC2 Image Builder. Pour la demande de SDK associée, consultez le lien Voir aussi sur la même page.

Vous pouvez filtrer les informations suivantes pour rationaliser la liste des résultats. Par exemple, si vous filtrez sur Propriétaire = Amazon dans la console, la liste affiche uniquement les flux de travail AWS gérés.

- Flux de travail
- Version
- Type
- Propriétaire

Console

Détails du flux de travail

Les détails figurant sur la page de liste des flux de travail liés à l'image de la console Image Builder sont notamment les suivants :

- Workflow : nom de la version la plus récente de la ressource de flux de travail d'images. Dans la console Image Builder, la colonne Workflow renvoie à la page détaillée du flux de travail.
- Version : version la plus récente de la ressource de flux de travail d'images.
- Type : type de flux de travail : `BUILDTEST`, ou`DISTRIBUTION`.

- Propriétaire : propriétaire de la ressource de flux de travail.
- Heure de création : date et heure auxquelles Image Builder a créé la version la plus récente de la ressource de flux de travail d'imagerie.
- ARN : nom de ressource Amazon (ARN) de la version actuelle de la ressource Image Workflow.

Répertorier les flux de

Pour répertorier les ressources du flux de travail d'imagerie dans la console Image Builder, effectuez les opérations suivantes :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez Flux de travail liés à l'image dans le volet de navigation.

Filtrer les résultats

Sur la page de liste des flux de travail liés à l'image, vous pouvez rechercher des flux de travail liés à l'image spécifiques pour filtrer vos résultats. Les filtres suivants sont disponibles pour les flux de travail liés aux images :

`Workflow`

Vous pouvez saisir tout ou partie du nom d'un flux de travail pour rationaliser les résultats. Par défaut, tous les flux de travail sont affichés dans la liste.

`Version`

Vous pouvez saisir tout ou partie d'un numéro de version pour rationaliser les résultats. Par défaut, toutes les versions sont affichées dans la liste.

`Type`

Vous pouvez filtrer par type de flux de travail ou afficher tous les types. Par défaut, tous les types de flux de travail sont affichés dans la liste.

- BUILD
- TEST
- DISTRIBUTION

`Owner`

Lorsque vous sélectionnez le filtre propriétaire dans la barre de recherche, Image Builder affiche la liste des propriétaires des flux de travail liés aux images de votre compte. Vous pouvez sélectionner un propriétaire dans la liste pour rationaliser les résultats. Par défaut, tous les propriétaires sont affichés dans la liste.

- Compte AWS— Le compte propriétaire de la ressource de flux de travail.
- Amazon : ressources de flux de production, de test et de distribution AWS gérées qu'Amazon possède et gère.

AWS CLI

Lorsque vous exécutez la list-workflows commande dans le AWS CLI, vous pouvez obtenir une liste des flux de travail d'images que vous possédez ou auxquels vous avez accès.

L'exemple de commande suivant montre comment utiliser la list-workflows commande sans filtre pour répertorier toutes les ressources de flux de travail d'image Image Builder que vous possédez ou auxquelles vous avez accès.

Exemple : répertorier tous les flux de travail liés à l'image

```
aws imagebuilder list-workflows
```

Sortie :

```
{
    "workflowVersionList": [
        {
            "name": "example-test-workflow",
            "dateCreated": "2023-11-21T22:53:14.347Z",
            "version": "1.0.0",
            "owner": "111122223333",
            "type": "TEST",
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
test/example-test-workflow/1.0.0"
        },
        {
            "name": "example-build-workflow",
            "dateCreated": "2023-11-20T12:26:10.425Z",
```

```
            "version": "1.0.0",
            "owner": "111122223333",
            "type": "BUILD",
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
build/example-build-workflow/1.0.0"
        },
        {
            "name": "example-distribution-workflow",
            "dateCreated": "2025-11-19T10:25:10.425Z",
            "version": "1.0.0",
            "owner": "111122223333",
            "type": "DISTRIBUTION",
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
distribution/example-distribution-workflow/1.0.0"
        }
    ]
}
```

Lorsque vous exécutez la list-workflows commande, vous pouvez appliquer des filtres pour rationaliser les résultats, comme le montre l'exemple suivant. Pour plus d'informations sur la façon de filtrer vos résultats, consultez la commande list-workflows dans la référence des AWS CLI commandes.

Exemple : filtre pour créer des flux de travail

```
aws imagebuilder list-workflows --filters name="type",values="BUILD"
```

Sortie :

```
{
    "workflowVersionList": [
        {
            "name": "example-build-workflow",
            "dateCreated": "2023-11-20T12:26:10.425Z",
            "version": "1.0.0",
            "owner": "111122223333",
            "type": "BUILD",
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
build/example-build-workflow/1.0.0"
        }
    ]
}
```

Obtenir les détails du flux de travail

Pour obtenir les détails d'une version spécifique du flux de travail d'imagerie, exécutez la commande get-workflow dans le. AWS CLI Pour l'action de l'API, voir le GetWorkflow document de référence de l'API EC2 Image Builder.

Exemple : obtenir les détails du flux de travail

```
aws imagebuilder get-workflow \
    --workflow-build-version-arn arn:aws:imagebuilder:us-
west-2:111122223333:workflow/build/example-build-workflow/1.0.0/1
```

La réponse inclut le flux de travail `typestate`, `data` (le document de flux de travail YAML), `parametersowner`, et`kmsKeyId`.

Création d'un flux de travail d'images

Lorsque vous créez un flux de travail d'images, vous avez davantage de contrôle sur votre processus de création d'images. Vous pouvez spécifier le flux de travail qui s'exécute lorsqu'Image Builder crée votre image, les flux de travail qui s'exécutent lorsqu'il teste l'image et le flux de travail qui s'exécute lorsqu'il distribue l'image. Vous pouvez également spécifier une clé gérée par le client pour chiffrer les ressources de votre flux de travail. Pour en savoir plus sur le chiffrement des ressources de votre flux de travail, consultezChiffrement et gestion des clés dans Image Builder.

Pour la création d'images, vous pouvez spécifier un flux de travail pour la phase de création, un ou plusieurs flux de travail pour la phase de test et un flux de travail pour l'étape de distribution. Vous pouvez même sauter complètement une étape, selon vos besoins. Vous configurez les actions que votre flux de travail entreprend dans le document de définition YAML qu'il utilise. Pour plus d'informations sur la syntaxe de votre document YAML, consultezCréation d'un document de flux de travail YAML.

Pour créer un nouveau flux de production, de test ou de distribution, sélectionnez l'onglet correspondant à votre environnement.

Console de gestion AWS

Vous pouvez utiliser le processus suivant pour créer un nouveau flux de travail dans la console Image Builder.

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Choisissez Image workflows dans le volet de navigation. Cela affiche la liste des flux de travail liés aux images que votre compte possède ou auxquels vous avez accès.

   > **Note**
   >
   > La liste inclut toujours les ressources de flux de travail gérés par Amazon qu'Image Builder utilise pour ses flux de travail par défaut. Pour afficher les détails de ces flux de travail, sélectionnez le lien Workflow.

3. Pour créer un nouveau flux de travail, choisissez Créer un flux de travail avec des images. La page Créer un flux de travail d'image s'affiche.
4. Configurez les détails de votre nouveau flux de travail. Sélectionnez le type de flux de travail en haut du formulaire : choisissez Créer pour créer un flux de travail de génération, Tester pour créer un flux de travail de test ou Distribution pour créer un flux de travail de distribution. Le type de flux de travail correspond à la `Type` valeur de l'API (`BUILDTEST`, ou`DISTRIBUTION`). Image Builder remplit la liste des modèles en fonction de cette option. Toutes les autres étapes sont identiques pour les workflows de création, de test et de distribution.

   Général

   La section générale inclut les paramètres qui s'appliquent à votre ressource de flux de travail, tels que le nom et la description. Les paramètres généraux sont les suivants :

   - Nom du flux de travail d'image (obligatoire) : nom de votre flux de travail d'image. Le nom doit être unique dans votre compte. Le nom peut comporter jusqu'à 128 caractères. Les caractères valides incluent les lettres, les chiffres`-`, les espaces et`_`.
   - Version (obligatoire) : version sémantique de la ressource de flux de travail à créer (major.minor.patch).
   - Description (facultatif) — Ajoutez éventuellement une description de votre flux de travail.
   - Clé KMS (optionnelle) : vous pouvez chiffrer les ressources de votre flux de travail à l'aide d'une clé gérée par le client. Pour de plus amples informations, veuillez consulter Chiffrez les flux d'images à l'aide d'une clé gérée par le client.

   Document de définition

Le document de flux de travail YAML contient toutes les configurations de votre flux de travail.

Prise en main

- Pour commencer avec un modèle par défaut d'Image Builder comme référence pour votre flux de travail, sélectionnez l'option Commencer à partir de modèles. Cette option est sélectionnée par défaut. Une fois que vous avez choisi le modèle à utiliser dans la liste des modèles, la configuration par défaut du modèle que vous avez sélectionné est copiée dans le contenu de votre nouveau document de flux de travail, où vous pouvez apporter des modifications.
- Pour définir votre document de flux de travail à partir de zéro, sélectionnez l'option Partir de zéro. Cela permet de remplir le contenu avec un bref aperçu de certaines parties importantes du format du document pour vous aider à démarrer.

Le panneau Contenu comprend une barre d'état en bas qui affiche des avertissements ou des erreurs pour votre document YAML. Pour plus d'informations sur la création d'un document de flux de travail YAML, consultezCréation d'un document de flux de travail YAML.

5. Lorsque vous avez terminé votre flux de travail, ou si vous souhaitez enregistrer la progression et y revenir plus tard, choisissez Créer un flux de travail.

AWS CLI

Avant d'exécuter la create-workflow commande dans le AWS CLI, vous devez créer le document YAML contenant l'ensemble de la configuration de votre flux de travail. Pour de plus amples informations, veuillez consulter Création d'un document de flux de travail YAML.

L'exemple suivant montre comment créer un flux de travail de génération à l'aide de la commande create-workflow AWS CLI . Le `--data` paramètre fait référence à un document YAML qui contient la configuration de génération pour le flux de travail que vous créez.

Exemple : création d'un flux de travail

```
aws imagebuilder create-workflow --name example-build-workflow --semantic-
version 1.0.0 --type BUILD --data file://example-build-workflow.yml
```

Sortie :

```
{
 "workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
build/example-build-workflow/1.0.0/1",
 "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

L'exemple suivant montre comment créer un flux de travail de test à l'aide de la commande create-workflow AWS CLI . Le `--data` paramètre fait référence à un document YAML qui contient la configuration de test pour le flux de travail que vous créez.

Exemple : création d'un flux de travail de test

```
aws imagebuilder create-workflow --name example-test-workflow --semantic-
version 1.0.0 --type TEST --data file://example-test-workflow.yml
```

Sortie :

```
{
 "workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
test/example-test-workflow/1.0.0/1",
 "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

L'exemple suivant montre comment créer un flux de travail de distribution à l'aide de la commande create-workflow AWS CLI . Le `--data` paramètre fait référence à un document YAML qui contient la configuration de distribution pour le flux de travail que vous créez.

Exemple : création d'un flux de travail de distribution

```
aws imagebuilder create-workflow --name example-distribution-workflow --semantic-
version 1.0.0 --type DISTRIBUTION --data file://example-distribution-workflow.yml
```

Sortie :

```
{
 "workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
distribution/example-distribution-workflow/1.0.0/1",
 "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

Création d'un document de flux de travail YAML

Le document de définition du format YAML configure les étapes d'entrée, de sortie et de flux de travail pour les étapes de création, de test et de distribution du processus de création d'images. Le même format de document s'applique aux trois types de flux de travail. Vous pouvez partir de modèles qui incluent des étapes standardisées, ou vous pouvez repartir de zéro pour définir votre propre flux de travail. Que vous utilisiez un modèle ou que vous partiez de zéro, vous pouvez personnaliser le flux de travail en fonction de vos besoins.

Structure d'un document de flux de travail YAML

Les sections suivantes décrivent la structure du document de flux de travail YAML qu'Image Builder utilise pour effectuer des actions de création, de test et de distribution d'images.

- Identification des documents relatifs au flux
- Paramètres de saisie des documents du flux de travail
- Étapes du document de travail
- Sorties de documents liés au flux

Identification des documents relatifs au flux

Identifie le flux de travail de manière unique. Cette section peut inclure les attributs suivants.

Champ	Description	Type	Obligatoire
name	Nom du document de flux de travail.	Chaîne	Non
description	Description du document.	Chaîne	Non
schemaVersion	La version du schéma du document, actuellement 1.0.	Chaîne	Oui

Exemple

```
---
name: sample-test-image
description: Workflow for a sample image, with extra configuration options exposed
 through workflow parameters.
schemaVersion: 1.0
```

Paramètres de saisie des documents du flux de travail

Cette partie du document de flux de travail définit les paramètres d'entrée que l'appelant peut spécifier. Si vous n'avez aucun paramètre, vous pouvez omettre cette section. Si vous spécifiez des paramètres, chaque paramètre peut inclure les attributs suivants.

Vous pouvez définir jusqu'à 25 paramètres dans un document de flux de travail. Chaque valeur de paramètre peut contenir jusqu'à 1 024 caractères. Un paramètre doit utiliser l'un des quatre types de données pris en charge : `string`,`integer`,`boolean`, ou`stringList`. Si vous ne spécifiez pas de `default` valeur pour un paramètre, vous devez fournir la valeur du paramètre au moment de l'exécution.

Champ	Description	Type	Obligatoire	Contraintes
name	Nom du paramètre.	Chaîne	Oui	
description	Description du paramètre.	Chaîne	Non	
default	La valeur par défaut du paramètre, si aucune valeur n'est fournie. Si vous n'incluez pas de valeur par défaut dans	Correspond au type de données du paramètre.	Non	

Champ	Description	Type	Obligatoire	Contraintes
	la définition du paramètre, celle-ci est requise lors de l'exécution.			
type	Type de données du paramètre. Si vous n'incluez pas le type de données dans la définition du paramètre, celui-ci est défini par défaut sur une valeur de chaîne requise lors de l'exécution.	Chaîne	Oui	Le type de données du paramètre doit être l'un des suivants : • `string` • `integer` • `boolean` • `stringList`

Exemple

Spécifiez le paramètre dans le document de flux de travail.

```
parameters:
  - name: waitForActionAtEnd
    type: boolean
    default: true
    description: "Wait for an external action at the end of the workflow"
```

Utilisez la valeur du paramètre dans le document de flux de travail.

```
$.parameters.waitForActionAtEnd
```

Étapes du document de travail

Spécifie jusqu'à 15 actions par étapes pour le flux de travail. Les étapes sont exécutées dans l'ordre dans lequel elles sont définies dans le document de flux de travail. En cas d'échec, une restauration s'exécute dans l'ordre inverse, en commençant par l'étape qui a échoué et en revenant aux étapes précédentes.

Chaque étape peut faire référence au résultat de toutes les actions de l'étape précédente. C'est ce que l'on appelle le chaînage ou le référencement. Pour faire référence à la sortie d'une action d'étape précédente, vous pouvez utiliser un sélecteur JSONPath. Par exemple :

```
$.stepOutputs.step-name.output-name
```

Pour de plus amples informations, veuillez consulter Utilisez des variables dynamiques dans votre document de flux de travail.

> **Note**
>
> Même si l'étape elle-même ne possède pas d'attribut de sortie, toute sortie d'une action d'étape est incluse `stepOutput` pour l'étape.

> **Note**
>
> Si vous définissez `waitSeconds` une étape, la valeur doit être inférieure à`timeoutSeconds`. Chaque action d'étape définit également une `timeoutSeconds` valeur maximale. Si vous définissez `timeoutSeconds` une valeur supérieure à la valeur maximale pour l'action, la validation échoue. Pour les maximums par action, consultez. Actions par étapes prises en charge pour votre document de flux de travail

Chaque étape peut inclure les attributs suivants.

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
action		Chaîne	Oui		

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
	Action de flux de travail exécutée par cette étape.				Il doit s'agir d'une action par étape prise en charge pour les documents de flux de travail Image Builder.

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
`if`, suivi d'un ensemble d'instructions condition nelles qui modifient l'`if`opérateur.	Les instructi ons condition nelles ajoutent un flux de points de décision de contrôle au corps des étapes de votre flux de travail.	Décret	Non		Image Builder prend en charge les instructions condition nelles suivantes en tant que modificat eurs de l'`if`opérateur : • Condition s de brancheme nt et modificat eurs :`if`,,`and`,`or`. `not` Les condition s de brancheme nt sont spécifiée s sur une ligne par elles-mêm es. • Opérateur s de comparais

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
					on : `booleanEq uals numberEqu als ,numberGre aterThan ,numbe aterThanE quals ,numberLes sThan ,numberLes sThanEqua ls ,stringEqu als .`
description	Description de l'étape.	Chaîne	Non		Les chaînes vides ne sont pas autorisées. Si elle est incluse, la longueur doit être comprise entre 1 et 1024 caractères.

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
inputs	Contient les paramètres que l'action de l'étape doit exécuter. Vous pouvez spécifier les valeurs clés sous forme de valeurs statiques ou à l'aide d'une variable JSONPath dont la résolution correspond au type de données correct.	Décret	Oui		

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
name	Le nom de l'étape. Ce nom doit être unique dans le document de flux de travail.	Chaîne	Oui		La longueur doit être comprise entre 3 et 128 caractères. Peut inclure des caractères alphanumériques et. _ Pas d'espaces.

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
onFailure	Configure l'action à effectuer en cas d'échec de l'étape, comme suit. Comportement • `Abort`: échoue à l'étape, échoue au flux de travail et n'exécute aucune autre étape après l'étape qui a échoué. Si la restauration est activée, elle commence par l'étape qui a échoué et se poursuit	Chaîne	Non	`Abort`	`Abort` | `Continue`

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
	jusqu'à ce que toutes les étapes qui l'autorisent soient annulées. • `Continue`— Échoue l'étape, mais continue d'exécuter les étapes restantes après l'étape qui a échoué. Dans ce cas, il n'y a pas de retour en arrière.				

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
Annulation activée	Détermine si l'étape sera annulée en cas de panne. Vous pouvez utiliser une valeur booléenne statique ou une variable JSONPath dynamique qui se résout en une valeur booléenne.	Booléen	Non	`true`	`true` | `false` | ou une variable JSONPath dont la résolution prend la valeur true ou false.
timeoutSeconds	Durée maximale, en secondes, pendant laquelle l'étape s'exécute avant d'échouer et de réessayer, si de nouvelles tentatives s'appliquent.	Entier	Non	Dépend de la valeur par défaut définie pour l'action de l'étape, le cas échéant.	Ne peut pas dépasser le délai maximum de l'action de l'étape

Champ	Description	Type	Obligatoire	Valeur par défaut	Contraintes
Quelques secondes d'attente	Durée, en secondes, pendant laquelle l'exécution de l'étape sera suspendue.	Entier	Non	0	Ne peut pas dépasser le délai TimeoutSeconds de l'action de l'étape

Exemple

```
steps:
  - name: LaunchTestInstance
    action: LaunchInstance
    onFailure: Abort
    inputs:
      waitFor: "ssmAgent"

  - name: ApplyTestComponents
    action: ExecuteComponents
    onFailure: Abort
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"

  - name: TerminateTestInstance
    action: TerminateInstance
    onFailure: Continue
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"

  - name: WaitForActionAtEnd
    action: WaitForAction
    if:
      booleanEquals: true
      value: "$.parameters.waitForActionAtEnd"
```

Sorties de documents liés au flux

Définit les sorties pour le flux de travail. Chaque sortie est une paire clé-valeur qui spécifie le nom de la sortie et la valeur. Vous pouvez utiliser les sorties pour exporter des données lors de l'exécution que les flux de travail suivants pourront utiliser. Cette section est facultative.

Vous pouvez définir jusqu'à 25 sorties dans un document de flux de travail. Chaque nom de sortie doit être unique pour tous les flux de travail que vous incluez dans votre pipeline. Un flux de travail ultérieur peut ensuite référencer une sortie par son nom avec`$.workflowOutputs.name`.

> **Note**
>
> Vous faites référence à la sortie d'une étape du même flux de travail avec`$.stepOutputs.stepName.field`, mais vous faites référence à la sortie d'un flux de travail antérieur avec uniquement le nom de sortie (`$.workflowOutputs.name`) et aucun nom d'étape.

Chaque sortie que vous définissez inclut les attributs suivants.

Champ	Description	Type	Obligatoire
name	Nom de la sortie. Le nom doit être unique pour tous les flux de travail que vous incluez dans votre pipeline.	Chaîne	Oui
value	La valeur de la sortie. La valeur de la chaîne peut être une variable dynamique, telle qu'un fichier de sortie issu d'une action d'étape. Pour de plus	Chaîne	Oui

Champ	Description	Type	Obligatoire
	amples informations, veuillez consulter Utilisez des variables dynamiques dans votre document de flux de travail.		

Exemple

Créez un ID d'image de sortie pour le document de flux de travail avec la sortie d'étape à partir de l'`createProdImage`étape.

```
outputs:
  - name: 'outputImageId'
    value: '$.stepOutputs.createProdImage.imageId'
```

Reportez-vous à la sortie du flux de travail dans le flux de travail suivant.

```
$.workflowOutputs.outputImageId
```

Actions par étapes prises en charge pour votre document de flux de travail

Cette section détaille les actions par étapes prises en charge par Image Builder.

Termes utilisés dans cette section

AMI

Amazon Machine Image

ARN

Amazon Resource Name

Le tableau suivant indique quelles étapes du flux de travail prennent en charge chaque action d'étape. Une coche (X) indique que vous pouvez utiliser l'action de l'étape à cette étape. Image Builder limite certaines actions à des étapes spécifiques : `ExecuteComponents` aux étapes

de création et de test, `CollectImageScanFindings` à la phase de test `DistributeImage` `ModifyImageAttributesApplyImageConfigurations`, et `DistributeContainerImage` à la phase de distribution. Vous pouvez utiliser toutes les autres actions à n'importe quel stade.

Action par étapes	Génération	test	Distribution	Remarques
LaunchInstance	X	X	X	Lance une instance Amazon EC2.
BootstrapInstanceForContainer	X	X	X	Installe Docker et les versions AWS CLI de conteneurs pour.
ExecuteComponents	X	X		Exécute les composants sur l'instance. Limité aux étapes de construction et de test.
CollectImageMetadata	X	X	X	AMI construit uniquement.
RunSysPrep	X	X	X	Windows se construit.
SanitizeInstance	X	X	X	Les versions de l'AMI Linux ne sont pas destinées aux conteneurs.
CreateImage	X	X	X	Crée une AMI à partir de l'instance de génération.
CreateImageFromIso	X	X	X	Crée une AMI à partir d'une image disque ISO.
RegisterImage	X	X	X	Enregistre une AMI à partir de captures instantanées.
DistributeContainerImage			X	Distribue une image de conteneur. Limité à la phase de distribution.
CollectImageScanFindings		X		Collecte les résultats de la numérisation d'images. Limité à la phase de test. Nécessite

Action par étapes	Génération	test	Distribution	Remarques
				Amazon Inspector et la numérisation d'images.
RunCommand	X	X	X	Exécute un document de commande AWS Systems Manager (Systems Manager).
WaitForSSMAgent	X	X	X	Attend la connectivité de Systems Manager après un redémarrage.
WaitForAction	X	X	X	Des pauses pour une action extérieure.
ExecuteStateMachine	X	X	X	Exécute une machine d'état Step Functions.
TerminateInstance	X	X	X	Met fin à une instance. Par exemple, un flux de travail de distribution de conteneurs met fin à l'instance de génération après avoir envoyé l'image.
DistributeImage			X	Copie une AMI dans les régions et les comptes. Limité à la phase de distribution.
ModifyImageAttributes			X	Modifie les autorisations de lancement de l'AMI. Limité à la phase de distribution.

Action par étapes	Génération	test	Distribution	Remarques
ApplyImageConfigurations			X	Applique les modèles de lancement, les licences, les exportations, le lancement rapide et les paramètres de Systems Manager. Limité à la phase de distribution.

> **Important**
>
> Un seul flux de travail distribue des AMI ou des images de conteneurs, et non les deux. Si un flux de travail inclut`DistributeContainerImage`, il ne peut pas également inclure `DistributeImageModifyImageAttributes`, ou`ApplyImageConfigurations`. Ces actions de distribution d'AMI fonctionnent sur les AMI Amazon EC2, tout en `DistributeContainerImage` agissant sur les images de conteneurs dans Amazon Elastic Container Registry (Amazon ECR).

Actions prises en charge

- ApplyImageConfigurations
- BootstrapInstanceForContainer
- CollectImageMetadata
- CollectImageScanFindings
- CreateImage
- CreateImageFromIso
- DistributeContainerImage
- DistributeImage
- ExecuteComponents
- ExecuteStateMachine
- LaunchInstance
- ModifyImageAttributes

- RegisterImage
- RunCommand
- RunSysPrep
- SanitizeInstance
- TerminateInstance
- WaitForAction
- WaitForSSMAgent

ApplyImageConfigurations

Utilisez cette étape pour appliquer diverses configurations et intégrations aux AMI distribuées, telles que les configurations de licence, les configurations de modèles de lancement, les configurations d'exportation S3, les configurations de lancement rapide EC2 et les configurations de paramètres de Systems Manager. Les configurations s'appliquent aux images distribuées uniquement dans le compte source, à l'exception des configurations de paramètres SSM qui peuvent être appliquées entre comptes.

> **Note**
>
> La `configurations` saisie est facultative. Si vous le fournissez, Image Builder utilise les entrées de votre flux de travail et ignore la configuration de distribution jointe. Si vous l'omettez, Image Builder revient à la configuration de distribution associée au pipeline ou à l'image.

Délai d'attente par défaut : 360 minutes

Délai maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure à la valeur maximale échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape. La `distributedImages.$` saisie est obligatoire. Les lignes restantes décrivent les champs de chaque élément du `configurations` tableau.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
Images distribuées. $	La liste des images distribuées à configurer. Référencez le `distributedImages` résultat d'une `DistributeImage` étape du même flux de travail.	Chaîne	Oui		Une référence JSONPath qui commence par `$.stepOutputs.`
configurations	Les configurations à appliquer aux AMI distribuées, sous forme de liste. Chaque objet cible une AWS région.	Tableau	Non		
configurations : région	AWS Région de l'AMI distribuée à configurer.	Chaîne	Oui		Obligatoire pour chaque article de `configurations` .
configurations : licence	L'ARN de configuration	Tableau	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
ConfigurationArns	de licence pour l'image.				
configurations : launchTemplateConfigurations	Les configurations du modèle de lancement à appliquer aux AMI distribuées.	Tableau	Non		
configurations : launchTemplateConfigurations:launchTemplateId	L'ID du modèle de lancement à appliquer à l'image.	Chaîne	Oui si cela `launchTemplateConfigurations` est spécifié		
configurations : launchTemplateConfigurations:accountId	Les identifiants de compte du modèle de lancement à appliquer à l'image.	Chaîne	Non		
configurations : launchTemplateConfigurations:setDefaultVersion	Paramètre de version par défaut du modèle de lancement pour l'image.	Booléen	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
configurations : s3ExportConfiguration	Configuration d'exportation Amazon S3 pour les AMI distribuées.	Tableau	Non		
configurations : s3ExportConfiguration:roleName	Le nom du rôle de configuration d'exportation Amazon S3 pour l'image.	Chaîne	Oui si cela `s3ExportConfiguration` est spécifié		
configurations : s3ExportConfiguration:diskImageFormat	Format d'image disque de configuration d'exportation Amazon S3 pour l'image.	Chaîne	Oui si cela `s3ExportConfiguration` est spécifié		Valeurs autorisées - VMDK|RAW|VHD
configurations : s3ExportConfiguration:s3Bucket	Le nom du compartiment de configuration d'exportation Amazon S3 pour l'image.	Chaîne	Oui si cela `s3ExportConfiguration` est spécifié		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
configurations : s3ExportConfiguration:s3Prefix	Le préfixe du compartiment de configuration d'exportation Amazon S3 pour l'image.	Chaîne	Non		
configurations : rapidesLaunchConfigurations	Configuration Amazon EC2 Fast Launch pour l'image.	Tableau	Non		
configurations : rapidesLaunchConfigurations:enabled	Indique si Amazon EC2 Fast Launch est activé pour l'image.	Booléen	Oui si cela `fastLaunchConfigurations` est spécifié		
configurations : rapidesLaunchConfigurations:snapshotConfiguration	Configuration des instantanés pour le pré-provisionnement des instantanés Amazon EC2 Fast Launch.	Map	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
configurations : rapidesLaunchConfigurations:snapshotConfiguration:targetResourceCount	Le nombre de ressources cibles d'Amazon EC2 Fast Launch pour l'image.	Entier	Non		
configurations : rapidesLaunchConfigurations:maxParallelLaunches	Le nombre maximal de lancements parallèles d'Amazon EC2 Fast Launch pour l'image.	Entier	Non		
configurations : rapidesLaunchConfigurations:launchTemplate	Le modèle de lancement qu'Amazon EC2 Fast Launch utilise pour préparer des instantanés préprovisionnés.	Map	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
configurations : rapidesLaunchConfigurations:launchTemplate: launchTemplateId	L'ID du modèle de lancement Amazon EC2 Fast Launch pour l'image.	Chaîne	Non		
configurations : rapidesLaunchConfigurations:launchTemplate: launchTemplateName	Le nom du modèle de lancement Amazon EC2 Fast Launch pour l'image.	Chaîne	Non		
configurations : rapidesLaunchConfigurations:launchTemplate: launchTemplateVersion	La version du modèle de lancement Amazon EC2 Fast Launch pour l'image.	Chaîne	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
configurations : SSM Parameter Configurations	Configuration des paramètres de Systems Manager pour l'image.	Map	Non		
configurations : SSM Parameter Configurations:amiAccountId	ID du compte AMI du paramètre Systems Manager pour l'image.	Chaîne	Non		
configurations : SSM Parameter Configurations:parameterName	Nom du paramètre Systems Manager pour l'image.	Chaîne	Oui si cela `ssmParameterConfigurations` est spécifié		
configurations : SSM Parameter Configurations:dataType	Type de données de paramètre Systems Manager pour l'image.	Chaîne	Non		Valeurs autorisées - text|aws:ec2:image)

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
Images configurées	Liste des images configurées.	Tableau

Nom de sortie	Description	Type
configuré Images:accountId	L'ID du compte de destination de l'image distribuée.	Chaîne
configuré Images:name	Le nom de l'AMI.	Chaîne
configuré Images:amiId	ID AMI de l'image distribuée.	Chaîne
configuré Images:dateStarted	Heure UTC à laquelle la distribution a commencé.	Chaîne
configuré Images:dateStopped	Heure UTC à laquelle la distribution est terminée.	Chaîne
configuré Images:step	Étape à laquelle la distribution s'est arrêtée.	Terminé | | Associate LicensesRunning | UpdateLaunchTemplateRunning | PutSsmParametersRunning | UpdateFastLaunchConfiguration ExportAmiQueued ExportAmiRunning
configuré Images:region	AWS Région de l'image distribuée.	Chaîne
configuré Images:status	État de distribution.	Terminé | Échoué | Annulé | TimedOut
configuré Images:errorMessage	Message d'erreur, le cas échéant.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: ApplyImageConfigurations
  action: ApplyImageConfigurations
  onFailure: Abort
```

```
  inputs:
    distributedImages.$: $.stepOutputs.DistributeImageStep.distributedImages
    configurations:
      - region: us-west-2
        licenseConfigurationArns:
          - arn:aws:license-manager:us-west-2:111122223333:license-configuration:lic-
example
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.ApplyImageConfigurationsStep.configuredImages
```

BootstrapInstanceForContainer

Cette action d'étape exécute un script de service pour démarrer l'instance avec la configuration minimale requise pour exécuter des flux de travail de conteneur. Image Builder utilise sendCommand l'API Systems Manager pour exécuter ce script. Pour plus d'informations, voir AWS Systems Manager Exécuter la commande.

> **Note**
>
> Le script bootstrap installe les packages Docker AWS CLI et les packages Docker qui sont nécessaires pour qu'Image Builder puisse créer correctement des conteneurs Docker. Si vous n'incluez pas cette étape, la création de l'image risque d'échouer.

Délai d'attente par défaut : 60 minutes

Délai d'attente maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance à amorcer.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
courir CommandId	L'ID du gestionnaire de systèmes sendCommand qui a exécuté le script d'amorçage sur l'instance.	Chaîne
status	L'état a été renvoyé par le gestionnaire de systèmessendCommand.	Chaîne
output	Sortie renvoyée par le gestionnaire de systèmessendCommand.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: ContainerBootstrapStep
```

```
  action: BootstrapInstanceForContainer
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.ContainerBootstrapStep.status
```

CollectImageMetadata

Cette action d'étape n'est valide que pour les flux de travail de création.

EC2 Image Builder exécute l'agent AWS Systems Manager (Systems Manager) sur les instances EC2 qu'il lance pour créer et tester votre image. Image Builder collecte des informations supplémentaires sur l'instance utilisée pendant la phase de génération avec Systems Manager Inventory. Ces informations incluent le nom et la version du système d'exploitation (OS), ainsi que la liste des packages et leurs versions respectives, telles que signalées par votre système d'exploitation.

> **Note**
>
> Cette action d'étape ne fonctionne que pour les images qui créent des AMI.

Délai d'attente par défaut : 30 minutes

Délai d'attente maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : Image Builder annule toutes les ressources Systems Manager créées au cours de cette étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'instance de génération à laquelle appliquer les paramètres de métadonnées.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance de génération pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
osVersion	Le nom et la version du système d'exploitation collectés à partir de l'instance de compilation.	Chaîne
Identifiant de l'association	L'ID d'association Systems Manager utilisé pour la collecte de l'inventaire.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: CollectMetadataStep
  action: CollectImageMetadata
  onFailure: Abort
  inputs:
```

```
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Utilisez le résultat de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.CollectMetadataStep.osVersion
```

CollectImageScanFindings

Si Amazon Inspector est activé pour votre compte et que la numérisation d'images est activée pour votre pipeline, cette étape collecte les résultats de numérisation d'images signalés par Amazon Inspector pour votre instance de test. Cette action d'étape n'est pas disponible pour les flux de travail de création.

Délai d'attente par défaut : 120 minutes

Délai d'attente maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance sur laquelle le scan a été exécuté.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
courir CommandId	ID du responsable des systèmes sendCommand qui a exécuté le script pour collecter les résultats.	Chaîne
status	L'état a été renvoyé par le gestionnaire de systèmessendCommand.	Chaîne
output	Sortie renvoyée par le gestionnaire de systèmessendCommand.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: CollectFindingsStep
  action: CollectImageScanFindings
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.CollectFindingsStep.status
```

CreateImage

Cette étape crée une image à partir d'une instance en cours d'exécution avec l'`CreateImage`API Amazon EC2. Pendant le processus de création, l'étape d'action attend si nécessaire pour vérifier que les ressources ont atteint l'état correct avant de continuer.

Délai d'attente par défaut : 720 minutes

Délai maximum : 3 jours (259 200 secondes)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure à la valeur maximale échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'instance à partir de laquelle créer la nouvelle image.	Chaîne	Oui		L'instance correspondant à l'ID d'instance fourni doit être dans un `running` état au démarrage de cette étape.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
imageId	ID AMI de l'image créée.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: CreateImageFromInstance
  action: CreateImage
  onFailure: Abort
  inputs:
      instanceId.$: "i-1234567890abcdef0"
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.CreateImageFromInstance.imageId
```

CreateImageFromIso

Utilisez cette action pour créer une Amazon Machine Image (AMI) à partir d'une image disque ISO sur l'instance de génération. Utilisez-le dans un flux de création lorsque votre source est un fichier ISO plutôt qu'une AMI de base. Image Builder convertit l'ISO en AMI et attend que les ressources atteignent l'état requis avant de continuer.

> **Note**
>
> Cette action présente les mêmes exigences que l'importation d'une image disque ISO. Pour de plus amples informations, veuillez consulter Importez des images de disque ISO Windows vérifiées avec Image Builder.

Délai d'attente par défaut : 720 minutes

Délai maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure à la valeur maximale échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance de compilation qui contient l'image disque ISO.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance de

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
					génération pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
Nom de l'ami	Nom de l'AMI créée par l'étape.	Chaîne
SnapshotID	L'ID du snapshot qui sauvegarde l'AMI.	Chaîne
application	L'architecture de l'AMI (`x86_64`ou`arm64`).	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: CreateImageFromIsoStep
  action: CreateImageFromIso
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.CreateImageFromIsoStep.amiName
```

DistributeContainerImage

Utilisez cette action pour distribuer une image de conteneur créée par l'étape de création dans les référentiels Amazon ECR dans les régions cibles définies pour l'image. Utilisez cette action dans

un flux de distribution pour les images de conteneur (DOCKER). Il fonctionne comme l'équivalent de l'image de conteneur de. DistributeImage

Cette étape exécute une commande Systems Manager sur l'instance de génération pour envoyer l'image du conteneur. Docker doit être AWS CLI installé sur l'instance. Pour de plus amples informations, veuillez consulter BootstrapInstanceForContainer.

> **Important**
>
> Un flux de travail qui inclut DistributeContainerImage ne peut pas également inclure DistributeImageModifyImageAttributes, ouApplyImageConfigurations. Un seul flux de travail distribue des images de conteneurs ou des AMI, et non les deux.

Délai d'attente par défaut : 120 minutes

Délai maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut partimeoutSeconds, jusqu'à la valeur maximale affichée. Une valeur supérieure à la valeur maximale échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance de génération qui contient l'image du conteneur.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance de génération pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
status	L'état a été renvoyé par le gestionnaire de systèmessendCommand.	Chaîne
Régions succédées	Les régions où la distribution a réussi.	Tableau
Régions défaillantes	Les régions où la distribution a échoué.	Tableau

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: DistributeContainer
  action: DistributeContainerImage
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.DistributeContainer.status
```

DistributeImage

Cette étape distribue une AMI aux régions et comptes cibles que vous spécifiez. Il utilise la configuration de distribution que vous avez `CreateImage` `CreateImagePipeline` demandée ou des paramètres de distribution personnalisés dans le flux de travail qui remplacent la configuration par défaut.

> **Note**
>
> La `distributions` saisie est facultative. Si vous le fournissez, Image Builder utilise les entrées de votre flux de travail et ignore la configuration de distribution jointe. Si vous

l'omettez, Image Builder revient à la configuration de distribution associée au pipeline ou à l'image.

Délai d'attente par défaut : 360 minutes

Délai maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure à la valeur maximale échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
distributions	Les paramètres de distribution de l'AMI, sous forme de liste. Chaque objet cible une AWS région. Si vous omettez cette entrée, Image Builder utilise la configuration de distribution associée au pipeline ou à l'image.	Tableau	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
distributions:région	La AWS région à laquelle distribuer l'AMI.	Chaîne	Oui		Longueur minimale de 1. Longueur maximale de 1024.
distributions:nom	Nom à appliquer à l'AMI distribuée.	Chaîne	Non		
distributions:description	Description à appliquer à l'AMI distribuée.	Chaîne	Non		
distributions:cibleAccountIds	Les identifiants de compte auxquels distribuer l'AMI.	Tableau	Non		
Distributions : Ami Tags	Les balises à appliquer à l'AMI distribuée.	Map	Non		Maximum de 50 étiquettes.
distributions:kmKeyId	La clé KMS à utiliser pour chiffrer l'AMI distribuée.	Chaîne	Non		

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
Images distribuées	Liste des images distribuées	Tableau
distribué Images:region	AWS Région de l'image distribuée.	Chaîne
distribué Images:name	Le nom de l'AMI.	Chaîne
distribué Images:amiId	ID AMI de l'image distribuée.	Chaîne
distribué Images:accountId	L'ID du compte de destination de l'image distribuée.	Chaîne
distribué Images:dateStarted	Heure UTC à laquelle la distribution a commencé.	Chaîne
distribué Images:dateStopped	Heure UTC à laquelle la distribution est terminée.	Chaîne
distribué Images:status	État de distribution.	Terminé | Échoué | Annulé | TimedOut
distribué Images:step	Étape à laquelle la distribution s'est arrêtée.	Terminé| CopyAmiRunning
distribué Images:errorMessage	Message d'erreur, le cas échéant.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: DistributeImage
  action: DistributeImage
  onFailure: Abort
  inputs:
```

```
    distributions:
      - region.$: "$.parameters.SourceRegion"
        description: "AMI distribution to source region"
        amiTags:
          DistributionTest: "SourceRegion"
          WorkflowStep: "DistributeToSourceRegion"
          BuildDate: "{{imagebuilder:buildDate:yyyyMMHHss}}"
          BuildVersion: "{{imagebuilder:buildVersion}}"
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.DistributeImageStep.distributedImages
```

ExecuteComponents

Cette action d'étape exécute les composants spécifiés dans la recette pour l'image en cours de création. Les flux de travail de génération exécutent les composants de génération sur l'instance de génération. Les flux de travail de test exécutent uniquement des composants de test sur l'instance de test.

Image Builder utilise sendCommand l'API Systems Manager pour exécuter les composants. Pour plus d'informations, voir AWS Systems Manager Exécuter la commande.

Délai d'attente par défaut : 720 minutes

Délai d'attente maximum : 1 jour (24 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance sur laquelle les	Chaîne	Oui		Il doit s'agir de l'ID d'instance

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
	composants doivent s'exécuter.				en sortie de l'étape du flux de travail qui a lancé l'instance pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
courir CommandId	L'ID du gestionnaire de systèmes sendCommand qui a exécuté les composants sur l'instance.	Chaîne
status	L'état a été renvoyé par le gestionnaire de systèmessendCommand.	Chaîne
output	Sortie renvoyée par le gestionnaire de systèmessendCommand.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: ExecComponentsStep
  action: ExecuteComponents
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Utilisez le résultat de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.ExecComponentsStep.status
```

ExecuteStateMachine

Cette action d'étape lance l'exécution d'une machine à AWS Step Functions états à partir d'un flux de travail Image Builder. Image Builder utilise l'`StartExecution`API Step Functions pour lancer la machine à états et attend qu'elle soit terminée. Cela est utile pour intégrer des flux de travail complexes, des processus de validation de conformité ou de certification dans votre pipeline de création d'images.

Pour plus d'informations, voir En savoir plus sur les machines à états dans Step Functions du Guide du AWS Step Functions développeur.

Délai d'attente par défaut : 6 heures

Délai d'attente maximum : 24 heures

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
état MachineArn	L'ARN de la machine d'état Step Functions à exécuter.	Chaîne	Oui		Il doit s'agir d'un ARN de machine d'état valide.
input	Données d'entrée JSON à fournir à la	Chaîne	Non	{}	Il doit s'agir d'une chaîne JSON valide, longueur

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
	machine d'état.				maximale : 16 Ko.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
executionArn	L'ARN de l'exécution de la machine d'état.	Chaîne
output	Le résultat de l'exécution de la machine d'état.	Chaîne

Autorisations IAM requises

Votre rôle d'exécution personnalisé doit disposer des autorisations suivantes pour utiliser cette action d'étape :

Autoriser les actions

- `states:StartExecution`
- `states:DescribeExecution`

Spécifiez les ressources

- `arn:aws:states:`*`us-west-2`*`:`*`111122223333`*`:stateMachine:`*`state-machine-name`*
- `arn:aws:states:`*`us-west-2`*`:`*`111122223333`*`:execution:`*`state-machine-name`*`:*`

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: ValidateImageCompliance
  action: ExecuteStateMachine
```

```
  timeoutSeconds: 3600
  onFailure: Abort
  inputs:
    stateMachineArn: arn:aws:states:us-
west-2:111122223333:stateMachine:ImageComplianceValidation
    input: |
      {
        "imageId": "{{ $.stepOutputs.CreateImageFromInstance.imageId }}",
        "region": "us-west-2",
        "complianceLevel": "high",
        "requiredScans": ["cve", "benchmark", "configuration"]
      }
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.ValidateImageCompliance.executionArn
```

LaunchInstance

Utilisez cette action pour lancer une instance dans votre Compte AWS. L'action attend que l'instance réponde à la condition de contrôle de santé spécifiée et que l'agent Systems Manager soit en cours d'exécution avant de passer à l'étape suivante. L'action de lancement utilise les paramètres de votre recette et des ressources de configuration de l'infrastructure. Par exemple, le type d'instance à lancer provient de la configuration de l'infrastructure. La sortie est l'ID de l'instance qui a été lancée.

L'`waitFor`entrée configure la condition qui satisfait à l'exigence d'achèvement de l'étape. L'`healthCheck`entrée spécifie le niveau de validation de l'état de santé de l'instance EC2 qui doit réussir avant que l'étape ne se poursuive.

Délai d'attente par défaut : 75 minutes

Délai maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : pour les instances de build, la restauration exécute l'action que vous avez configurée dans la ressource de configuration de votre infrastructure. Par défaut, les instances de génération sont interrompues en cas d'échec de la création de l'image. Cependant, il existe un paramètre dans la configuration de l'infrastructure pour conserver l'instance de compilation à des fins de dépannage.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
image IdOverride	L'image à utiliser pour lancer l'instance	Chaîne	Non	Étape de construction : image, image de base de la recette Étape de test : sortie de l'AMI depuis la phase de construction	Doit être un identifiant AMI valide
instance TypesOver ride	Image Builder essaie chaque type d'instance de la liste jusqu'à ce qu'il en trouve un qui se lance correctement	Liste d'éléments String	Non	Types d'instanc es spécifiés dans votre configura tion d'infrast ructure	Il doit s'agir de types d'instance valides
Attendre	La condition à attendre avant de terminer l'étape du flux de travail et de passer à l'étape suivante	Chaîne	Oui	Aucune	Compatible avec Image Builders`smAgent`.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
Bilan de santé	Le niveau de validation de l'état EC2 requis avant de passer à l'étape du flux de travail. Spécifiez `RUNNING` pour des compilations plus rapides avec un minimum de validation, ou `STATUS_CHECK_PASSED` pour attendre que les contrôles d'état EC2 soient réussis.	Chaîne	Non	`STATUS_CHECK_PASSED`	Valeurs valides : `RUNNING` | `STATUS_CHECK_PASSED` .

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
instanceId	L'ID d'instance de l'instance qui a été lancée.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: LaunchStep
  action: LaunchInstance
  onFailure: Abort
  inputs:
    healthCheck: "RUNNING"
    waitFor: ssmAgent
```

Utilisez le résultat de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.LaunchStep.instanceId
```

ModifyImageAttributes

Utilisez cette action pour modifier les attributs des AMI distribuées, tels que les autorisations de lancement et d'autres attributs d'AMI. Il fonctionne sur des AMI qui ont été distribuées aux régions et aux comptes cibles.

> **Note**
>
> La `attributes` saisie est facultative. Si vous le fournissez, Image Builder utilise les entrées de votre flux de travail et ignore la configuration de distribution jointe. Si vous l'omettez, Image Builder revient à la configuration de distribution associée au pipeline ou à l'image.

Délai d'attente par défaut : 120 minutes

Délai maximum : 180 minutes (3 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
Images distribuées. $	La liste des images distribuées à modifier. Référencez le `distributedImages` résultat d'une `DistributeImage` étape du même flux de travail.	Chaîne	Oui		Une référence JSONPath qui commence par `$.stepOutputs.`
attributs	Les attributs de l'image à modifier, sous forme de liste. Chaque objet cible une AWS région.	Tableau	Non		
attributs : région	La AWS région de l'AMI distribuée à modifier.	Chaîne	Oui		Obligatoire pour chaque article `deattributes` .
Attributs : LaunchPermission	Les paramètres d'autorisation de lancement à appliquer	Map	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
	à l'AMI distribuée.				
attributs :lancement Permission:userIds	Les identifiants de compte à ajouter ou à supprimer des autorisations de lancement de l'AMI.	Tableau	Non		
attributs :lancement Permission:userGroups	Les groupes d'utilisateurs à ajouter ou à supprimer des autorisations de lancement pour l'AMI.	Tableau	Non		
attributs :lancement Permission:organizationArns	Les AWS organisations ARN doivent ajouter ou supprimer des autorisations de lancement pour l'AMI.	Tableau	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
attributs :lancement Permission:organizationalUnitArns	L'unité organisationnelle (OU) doit ajouter ou supprimer des autorisations de lancement pour l'AMI.	Tableau	Non		

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
Images modifiées	Liste des images modifiées	Tableau
modifié Images:accountId	L'ID du compte de destination de l'image distribuée.	Chaîne
modifié Images:name	Le nom de l'AMI.	Chaîne
modifié Images:amiId	ID AMI de l'image distribuée.	Chaîne
modifié Images:dateStarted	Heure UTC à laquelle la distribution a commencé.	Chaîne
modifié Images:dateStopped	Heure UTC à laquelle la distribution est terminée.	Chaîne
modifié Images:step	Étape à laquelle la distribution s'est arrêtée.	Terminé| ModifyAmiRunning
modifié Images:region	La AWS région de l'image.	Chaîne

Nom de sortie	Description	Type
modifié Images:status	État de la distribution.	Terminé | Échoué | Annulé | TimedOut
modifié Images:errorMessage	Message d'erreur, le cas échéant.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: ModifyImageAttributes
  action: ModifyImageAttributes
  onFailure: Abort
  inputs:
    distributedImages.$: $.stepOutputs.DistributeImageStep.distributedImages
    attributes:
      - region: us-west-2
        launchPermission:
          userIds:
            - "111122223333"
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.ModifyImageAttributesStep.modifiedImages
```

RegisterImage

Cette étape enregistre une nouvelle Amazon Machine Image (AMI) à l'aide de l' RegisterImage API Amazon EC2. Il vous permet de créer une AMI à partir d'un instantané ou d'un ensemble d'instantanés existant, en spécifiant divers attributs d'image.

Délai d'attente par défaut : 540 minutes

Délai maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
application	L'architecture de l'AMI.	Chaîne	Non		Valeurs valides : i386, x86_64, arm64, x86_64_mac, arm64_mac
bloquerDeviceMapping	Les entrées de mappage des périphériques de bloc pour l'AMI.	Tableau	Non		
Mode de démarrage	Mode de démarrage de l'AMI.	Chaîne	Non		Valeurs valides : legacy-bios, uefi, uefi-preferred
description	Description de l'AMI.	Chaîne	Non		
Assistance ENA	Si la mise en réseau améliorée avec ENA est activée.	Booléen	Non		
Emplacement de l'image	Emplacement du manifeste AMI.	Chaîne	Non		Obligatoire pour les S3-backed AMI
Assistance IMDS	Le niveau de support IMDSv2.	Chaîne	Non		Valeurs valides : v2.0
inclureSnapshotTags	S'il faut inclure les balises du premier instantané défini dans le mappage des périphériques en mode bloc.	Booléen	Non	FALSE	Lorsque ce paramètre est défini sur true, les balises sont incluses comme suit : •

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
					Les balises `SnapshotId` du premier volume EBS de la `blockDeviceMapping` liste contenant a `SnapshotId` sont fusionnées avec les balises AMI de sortie. • Les balises AMI de sortie ont priorité sur les balises de capture instantanée dotées de la même clé. • AWS les balises réservées (celles dont les clés commencent par `aws:`) sont automatiquement exclues. • Si plusieurs volumes EBS `SnapshotId` sont définis, seules les balises du premier volume EBS de la

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
					liste contenant un `SnapshotId` sont incluses.
Identifiant du noyau	L'ID du noyau à utiliser.	Chaîne	Non		
ID de disque RAM	L'ID du disque RAM à utiliser.	Chaîne	Non		
racine DeviceName	Le nom de l'appareil racine.	Chaîne	Non		Exemple :/dev/sda1
sriov NetSupport	Mise en réseau améliorée grâce à l'interface Intel 82599 VF.	Chaîne	Non		
Assistance TPM	Prise en charge de la version TPM.	Chaîne	Non		Valeurs valides : v2.0
Données UEFI	Base64-encoded Données UEFI.	Chaîne	Non		
Type de virtualisation	Type de virtualisation.	Chaîne	Non		Valeurs valides : hvm, paravirtual

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
imageId	ID AMI de l'image enregistrée.	Chaîne

Autorisations IAM requises

Votre rôle d'exécution personnalisé doit disposer des autorisations suivantes pour utiliser cette action d'étape :

Autoriser les actions

- `ec2:DescribeSnapshots`
- `ec2:CreateTags`

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: RegisterNewImage
  action: RegisterImage
  onFailure: Abort
  inputs:
    architecture: "x86_64"
    bootMode: "uefi"
    blockDeviceMapping:
      - DeviceName: "/dev/sda1"
        Ebs:
          SnapshotId: "snap-1234567890abcdef0"
          VolumeSize: 100
          VolumeType: "gp3"
    rootDeviceName: "/dev/sda1"
    virtualizationType: "hvm"
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.RegisterNewImage.imageId
```

Exemple avec une étape SnapshotId issue d'une autre étape et des balises de capture instantanée incluses dans l'AMI générée

```
- name: CreateSnapshot
  action: RunCommand
  onFailure: Abort
  inputs:
    instanceId: "i-1234567890abcdef0"
    documentName: "AWS-RunShellScript"
    parameters:
      commands:
        - "aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description
 'Snapshot for AMI' --query 'SnapshotId' --output text"

- name: RegisterImageFromSnapshot
  action: RegisterImage
  onFailure: Abort
  inputs:
    architecture: "x86_64"
    bootMode: "uefi"
    blockDeviceMapping:
      - DeviceName: "/dev/sda1"
        Ebs:
          SnapshotId.$: "$.stepOutputs.CreateSnapshot.output[0]"
          VolumeSize: 100
          VolumeType: "gp3"
    includeSnapshotTags: true
    rootDeviceName: "/dev/sda1"
    virtualizationType: "hvm"
```

RunCommand

Cette étape exécute un document de commande pour votre flux de travail. Image Builder utilise sendCommand l'API de Systems Manager pour l'exécuter pour vous. Pour plus d'informations, voir AWS Systems Manager Exécuter la commande.

Délai d'attente par défaut : 720 minutes

Délai maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance sur laquelle exécuter le document de commande.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance pour ce flux de travail.
documentName	Nom du document de commande Systems Manager à exécuter.	Chaîne	Oui		
parameters	Liste de paires clé-valeur pour tous les paramètres requis par le document de commande.	<string>dictionnaire<chaîne, liste >	Conditionnel		
Version du document	Version du document de commande à exécuter.	Chaîne	Non	$DEFAULT	

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
courir CommandId	L'ID du gestionnaire de systèmes sendCommand qui a exécuté le document de commande sur l'instance.	Chaîne
status	L'état a été renvoyé par le gestionnaire de systèmessendCommand.	Chaîne
output	Sortie renvoyée par le gestionnaire de systèmessendCommand.	Liste de chaînes

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: RunCommandDoc
  action: RunCommand
  onFailure: Abort
  inputs:
    documentName: SampleDocument
    parameters:
        osPlatform:
          - "linux"
    instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.RunCommandDoc.status
```

RunSysPrep

Cette étape utilise l'sendCommandAPI Systems Manager pour exécuter le `AWSEC2-RunSysprep` document pour les instances Windows avant que l'instance de génération ne s'arrête pour la capture d'écran. Ces actions suivent les AWS meilleures pratiques en matière de durcissement et de nettoyage de l'image .

Délai d'attente par défaut : 60 minutes

Délai d'attente maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance sur laquelle exécuter le `AWSEC2-RunSysprep` document.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
courir CommandId	L'ID du gestionnaire de systèmes sendCommand qui a exécuté le `AWSEC2-RunSysprep` document sur l'instance.	Chaîne
status	L'état a été renvoyé par le gestionnaire de systèmessendCommand.	Chaîne

Nom de sortie	Description	Type
output	Sortie renvoyée par le gestionnaire de systèmessendCommand.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: RunSysprep
  action: RunSysPrep
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.RunSysprep.status
```

SanitizeInstance

Cette étape exécute le script de nettoyage recommandé pour les instances Linux avant l'arrêt de l'instance de génération pour le snapshot. Le script de nettoyage permet de s'assurer que l'image finale respecte les meilleures pratiques de sécurité et que les artefacts de création ou les paramètres qui ne devraient pas être transférés sur votre instantané sont supprimés. Pour plus d'informations sur le script, consultezNettoyage requis après la construction. Cette action d'étape ne s'applique pas aux images de conteneurs.

Image Builder utilise sendCommand l'API Systems Manager pour exécuter ce script. Pour plus d'informations, voir AWS Systems Manager Exécuter la commande.

Délai d'attente par défaut : 60 minutes

Délai d'attente maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance à assainir.	Chaîne	Oui		Il doit s'agir de l'ID d'instance en sortie de l'étape du flux de travail qui a lancé l'instance pour ce flux de travail.

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
courir CommandId	L'ID du gestionnaire de systèmes sendCommand qui a exécuté le script de nettoyage sur l'instance.	Chaîne
status	L'état a été renvoyé par le gestionnaire de systèmessendCommand.	Chaîne
output	Sortie renvoyée par le gestionnaire de systèmessendCommand.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: SanitizeStep
  action: SanitizeInstance
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.SanitizeStep.status
```

TerminateInstance

Cette action met fin à l'instance que vous spécifiez avec l'`instanceId`entrée. Par défaut, il déclenche la résiliation et passe à l'étape suivante sans attendre. Un processus d'arrière-plan gère le nettoyage si la terminaison rencontre des erreurs de service EC2. Pour attendre la fin complète avant de continuer, réglez l'`waitForTermination`entrée sur`true`.

Délai d'attente par défaut : 30 minutes

Délai d'attente maximum : 720 minutes (12 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance à terminer.	Chaîne	Oui	Aucune	Aucune
attendre ForTermin ation	Spécifie si l'étape attend que l'instance	Booléen	Non	`false`	`true` | `false`

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
	atteigne l'état terminé avant de poursuivre. Lorsqu'elle est réglée sur`true`, l'étape attend la fin complète (10 secondes à plusieurs minutes).				

Sorties : aucune sortie n'est disponible pour cette action d'étape.

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: TerminateInstance
  action: TerminateInstance
  onFailure: Continue
  inputs:
      instanceId.$: i-1234567890abcdef0
      waitForTermination: false
```

WaitForAction

Cette étape interrompt le flux de travail en cours et attend de recevoir une action externe de l'action de l'SendWorkflowStepActionAPI Image Builder. Cette étape publie un EventBridge événement dans votre bus d' EventBridge événements par défaut avec un type de détail`EC2 Image Builder Workflow Step Waiting`. L'étape peut également envoyer une notification SNS si vous fournissez un ARN de rubrique SNS, ou invoquer une fonction Lambda de manière asynchrone si vous fournissez un nom de fonction Lambda.

Fonctionnement de la pause et de la reprise

Lorsqu'une `WaitForAction` étape est exécutée, Image Builder définit le statut de l'étape sur `WAITING` et publie un EventBridge événement (type de détail`EC2 Image Builder Workflow Step Waiting`). Si vous le fournissez`snsTopicArn`, il envoie également une notification Amazon SNS. Si vous fournissez`lambdaFunctionName`, il invoque cette fonction de manière asynchrone. Le flux de travail reste suspendu jusqu'à ce que l'une des situations suivantes se produise :

- Vous appelez SendWorkflowStepAction with `RESUME` pour continuer ou `STOP` pour mettre fin au flux de travail.
- L'étape atteint son délai d'expiration (3 jours par défaut, maximum 7 jours) et le flux de travail échoue.

Le `reason` message que vous passez SendWorkflowStepAction est enregistré avec l'étape et est disponible pour les étapes suivantes en tant que `reason` résultat de l'étape.

Délai d'attente par défaut : 3 jours

Délai d'attente maximum : 7 jours (604 800 secondes)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
sns TopicArn	Un ARN de rubrique SNS facultatif pour envoyer une notification lorsque l'étape du flux de travail est en attente.	Chaîne	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
lambdaFunctionName	Nom ou ARN facultatif de la fonction Lambda à invoquer de manière asynchrone lorsque l'étape du flux de travail est en attente.	Chaîne	Non		

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
payload	Chaîne JSON utilisée comme message pour SNS et charge utile pour Lambda. Si elle est fournie, une charge utile personnalisée est encapsulée par défaut message/payload, utilisée respectivement pour SNS et Lambda. S'il n'est pas fourni, génère la valeur par défaut message/payload.	Chaîne	Non		Doit être une chaîne JSON valide, 16 Ko maximum

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
action	Action renvoyée par l'action d'SendWorkflowStepActionAPI.	Chaîne (RESUMEouSTOP)
raison	La raison de l'action renvoyée.	Chaîne

Autorisations IAM requises

Votre rôle d'exécution personnalisé doit disposer des autorisations suivantes pour utiliser cette action d'étape :

Autoriser les actions

- `lambda:InvokeFunction`

Spécifiez les ressources

- `arn:aws:lambda:us-west-2:111122223333:function:function-name`
- `arn:aws:lambda:us-west-2:111122223333:function:*`

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail avec notification SNS.

```
- name: SendEventAndWait
  action: WaitForAction
  onFailure: Abort
  inputs:
    snsTopicArn: arn:aws:sns:us-west-2:111122223333:ExampleTopic
```

Spécifiez l'action de l'étape dans le document de flux de travail en invoquant la fonction Lambda.

```
- name: SendEventAndWaitWithLambda
  action: WaitForAction
  onFailure: Abort
  inputs:
```

```
    lambdaFunctionName: ExampleFunction
    payload: |
      {
        "imageId": "{{ $.stepOutputs.CreateImageFromInstance.imageId }}",
        "region": "us-west-2"
      }
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.SendEventAndWait.reason
```

WaitForSSMAgent

Cette étape attend qu'une instance EC2 devienne gérable AWS Systems Manager après les périodes d'absence de réponse prévues. Il est particulièrement utile pour les flux de travail présentant des interruptions d'instance connues, telles que les redémarrages du système, les mises à niveau du système d'exploitation ou les opérations spécifiques à la plate-forme qui déconnectent temporairement l'instance de SSM. Image Builder surveille l'instance jusqu'à ce qu'elle retrouve la connectivité SSM ou qu'elle expire.

Délai d'attente par défaut : 60 minutes

Délai d'attente maximum : 180 minutes (3 heures)

Vous pouvez remplacer la valeur par défaut par`timeoutSeconds`, jusqu'à la valeur maximale affichée. Une valeur supérieure au maximum échoue à la validation.

Annulation : aucune annulation n'est possible pour cette action d'étape.

Entrées : le tableau suivant répertorie les entrées prises en charge pour cette action d'étape.

Nom de l'entrée	Description	Type	Obligatoire	Par défaut	Contraintes
instanceId	L'ID de l'instance à surveille r pour la connectivité SSM.	Chaîne	Oui		Doit être un ID d'instance EC2 valide

Sorties : le tableau suivant inclut les sorties pour cette action d'étape.

Nom de sortie	Description	Type
status	État de connexion de l'agent SSM.	Chaîne

Exemple

Spécifiez l'action de l'étape dans le document de flux de travail.

```
- name: WaitForInstanceAfterReboot
  action: WaitForSSMAgent
  onFailure: Abort
  timeoutSeconds: 900
  inputs:
    instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Utilisez la sortie de la valeur de l'action de l'étape dans le document de flux de travail.

```
$.stepOutputs.WaitForInstanceAfterReboot.status
```

Utilisez des variables dynamiques dans votre document de flux de travail

Vous pouvez utiliser des variables dynamiques dans vos documents de flux de travail pour représenter des valeurs qui varient au moment de l'exécution de votre processus de création d'images. L'interpolation de chaînes pour les variables dynamiques vous permet d'intégrer des expressions JSONPath dans du contenu structuré tel que des chaînes JSON. Cela est particulièrement utile lorsque vous devez transmettre des valeurs d'exécution dans des charges utiles complexes à des actions d'étape telles que ou. `ExecuteStateMachine` `WaitForAction`

Pour utiliser l'interpolation de chaînes pour les variables dynamiques, placez les expressions JSONPath entre accolades `"{{...}}"` doubles dans le contenu de votre chaîne. Seules les expressions JSONPath entourées de doubles accolades sont traitées comme des variables. Toutes les expressions JSONPath qui ne sont pas entourées de doubles accolades sont traitées comme du contenu de chaîne littéral.

Syntaxe des variables de flux de travail dynamiques JSONPath

```
$.<document structure>.[<step name>.]<variable name>
```

Les valeurs des variables dynamiques sont représentées sous forme de sélecteurs JSONPath avec des nœuds structurels qui identifient de manière unique la variable cible. Le premier nœud après la racine ($) fait référence à la structure du document du flux de travail, telle que, dans le cas des variables système Image Builder`stepOutputs`, ou dans le cas des variables système Image Builder,`imageBuilder`. La liste suivante contient les nœuds de structure de document de flux de travail JSONPath pris en charge.

Nœuds de structure de document

- parameters - Les paramètres du flux de travail
- StepOutputs - Résultats d'une étape du même document de flux de travail
- WorkflowOutputs - Sorties d'un document de flux de travail déjà exécuté
- imagebuilder - Variables système Image Builder

Les nœuds `parameters` et la structure du `stepOutputs` document incluent un nœud facultatif pour le nom de l'étape. Cela permet de garantir des noms de variables uniques à toutes les étapes.

Le nœud final du JSONPath est le nom de la variable cible, par exemple. `instanceId`

Chaque étape peut faire référence à la sortie de toute action d'étape précédente avec ces variables dynamiques JSONPath. Ceci est également connu sous le nom de chaînage ou de référencement. Pour faire référence à la sortie d'une action d'étape précédente, vous pouvez utiliser la variable dynamique suivante.

```
$.stepOutputs.step-name.output-name
```

> **Important**
>
> Lorsqu'un paramètre d'entrée fait référence à une variable dynamique, l'indicateur de chaînage (`.$`) doit être attaché à la fin du nom du paramètre.

Exemple 1 : indicateur de chaînage des paramètres d'entrée

L'exemple suivant montre un paramètre d'entrée qui utilise l'interpolation de chaîne pour résoudre une variable dynamique dans la valeur du paramètre au moment de l'exécution.

```
  - name: ApplyTestComponents
    action: ExecuteComponents
    onFailure: Abort
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"
```

Exemple 2 : interpolation de chaînes dans des variables dynamiques

L'exemple suivant montre comment les variables dynamiques utilisent l'interpolation de chaînes pour déterminer les valeurs au moment de l'exécution.

```
- name: ValidateImageConfiguration
  action: ExecuteStateMachine
  inputs:
    stateMachineArn: arn:aws:states:us-east-1:111122223333:stateMachine:ImageValidation
    input: |
      {
        "imageId": "{{ $.stepOutputs.CreateImageFromInstance.imageId }}",
        "region": "us-east-1",
        "buildDate": "{{ $.imagebuilder.dateTime }}",
        "instanceType": "{{ $.stepOutputs.LaunchStep.instanceType }}"
      }
```

Dans cet exemple, les expressions JSONPath entourées de doubles accolades sont résolues au moment de l'exécution :

- `{{ $.stepOutputs.CreateImageFromInstance.imageId }}`- Corrige à l'identifiant d'image réel de l' CreateImageFromInstance étape
- `{{ $.imagebuilder.dateTime }}`- Corrige à l'horodatage de compilation actuel. Consultez Utiliser les variables système Image Builder la liste des variables système Image Builder que vous pouvez utiliser.
- `{{ $.stepOutputs.LaunchStep.instanceType }}`- Corrige au type d'instance utilisé dans LaunchStep

Les chaînes littérales like `"region": "us-east-1"` restent inchangées.

> **Note**
>
> L'interpolation de chaînes fonctionne avec n'importe quel contenu de chaîne de votre document de flux de travail, y compris les chaînes multilignes à l'aide de l'opérateur YAML pipe (). | L'exigence d'accolades agit comme un mécanisme d'échappement pour faire clairement la distinction entre les variables JSONPath et le contenu textuel littéral.

Utiliser les variables système Image Builder

Image Builder fournit les variables système suivantes que vous pouvez utiliser dans votre document de flux de travail. Il s'agit de valeurs en lecture seule qu'Image Builder renseigne lors de l'exécution. Vous faites référence à une variable système avec le nœud de structure du `imagebuilder` document, par exemple`$.imagebuilder.platform`.

Nom de la variable	Description	Type	Exemple de valeur
nuage WatchLogGroup	Nom du groupe de CloudWatch journaux pour les journaux de sortie. Format : `/aws/imagebuilder/`*`<recipe-name>`*	Chaîne	`/aws/imagebuilder/`*`sampleImageRecipe`*
nuage WatchLogStream	Le nom du flux CloudWatch Logs pour les logs de sortie.	Chaîne	*`1.0.0/1`*
collecter ImageMetadata	Paramètre qui indique à Image Builder s'il souhaite collecter	Booléen	`true` | `false`

Nom de la variable	Description	Type	Exemple de valeur
	les métadonnées de l'instance.		
collecter ImageScan Findings	La valeur actuelle du paramètre qui permet à Image Builder de collecter les résultats de numérisation d'images.	Booléen	`true` | `false`
image BuildNumber	Le numéro de version de build de l'image.	Entier	*1*
imageId	ID AMI de l'image de base.	Chaîne	*ami-12345 67890abcdef1*
Nom de l'image	Le nom de l'image.	Chaîne	*sampleImage*
imageType	Type de sortie d'image.	Chaîne	`AMI` | `Docker`
image VersionNumber	Numéro de version de l'image.	Chaîne	*1.0.0*
instance ProfileName	Nom du rôle de profil d'instance qu'Image Builder utilise pour lancer des instances de création et de test.	Chaîne	*SampleIma geBuilder InstanceP rofileRole*

Nom de la variable	Description	Type	Exemple de valeur
platform	Plateforme du système d'exploitation de l'image créée.	Chaîne	`Linux` | `Windows` | `MacOS`
Journaux S3	Objet JSON contenant la configuration des journaux S3 écrits par Image Builder.	Objet JSON	{'S3Logs' : {'s3 BucketName ':' ', 's3*sample-bucket* ' : KeyPrefix ''}} *ib-logs*
securityGroups	Les ID des groupes de sécurité qui s'appliquent à la création et au test des instances.	Liste [Chaîne]	*[sg-1234567890abcdef1, sg-11112222333344445]*
Source : Imagearn	Le nom de ressource Amazon (ARN) de la ressource d'image Image Builder que le flux de travail utilise pour les étapes de création et de test.	Chaîne	arn:aws:imagebuilder : ::image//*us-east-1 111122223333 sampleImage 1.0.0/1*
subnetId	L'ID du sous-réseau dans lequel lancer les instances de génération et de test.	Chaîne	*subnet-1234567890abcdef1*

Nom de la variable	Description	Type	Exemple de valeur
mettre fin InstanceO nFailure	La valeur actuelle du paramètre qui demande à Image Builder de mettre fin à l'instance en cas de panne ou de la conserver à des fins de dépannage.	Booléen	`true` | `false`
Phase de flux de travail	Étape en cours d'exécution du flux de travail.	Chaîne	`Build` | `Test`
workingDirectory	Le chemin d'accès au répertoire de travail.	Chaîne	`/tmp`

Utilisez des instructions conditionnelles dans les étapes de votre flux de travail

Les instructions conditionnelles commencent par l'attribut de document de `if` déclaration. L'objectif ultime de l'`if`instruction est de déterminer s'il faut exécuter l'action de l'étape ou l'ignorer. Si l'`if`instruction se résout à`true`, l'action de l'étape est exécutée. Si tel est le cas`false`, Image Builder ignore l'action de l'étape et enregistre l'état de l'étape `SKIPPED` dans le journal.

L'`if`instruction prend en charge les instructions de branchement (`and`,`or`) et les modificateurs conditionnels (`not`). Il prend également en charge les opérateurs de comparaison suivants qui effectuent des comparaisons de valeurs (égal, inférieur à, supérieur à) en fonction des types de données qu'il compare (chaîne ou nombre).

Opérateurs de comparaison pris en charge

- `booleanEquals`

- `numberEquals`
- `numberGreaterThan`
- `numberGreaterThanEquals`
- `numberLessThan`
- `numberLessThanEquals`
- `stringEquals`

Règles pour les instructions de branchement et les modificateurs conditionnels

Les règles suivantes s'appliquent aux instructions de branchement (`and`,`or`) et aux modificateurs conditionnels (`not`).

- Les instructions de branchement et les modificateurs conditionnels doivent apparaître seuls sur une ligne.
- Les instructions de branchement et les modificateurs conditionnels doivent respecter les règles de niveau.
  - Il ne peut y avoir qu'une seule déclaration au niveau des parents.
  - Chaque branche ou modificateur enfant commence un nouveau niveau.

  Pour plus d'informations sur les niveaux, consultezNiveaux imbriqués dans les déclarations conditionnelles.
- Chaque instruction de branchement doit avoir au moins une instruction conditionnelle enfant, mais pas plus de dix.
- Les modificateurs conditionnels ne fonctionnent que sur une seule instruction conditionnelle enfant.

Niveaux imbriqués dans les déclarations conditionnelles

Les déclarations conditionnelles fonctionnent à plusieurs niveaux dans une section qui leur est propre. Par exemple, l'attribut de `if` déclaration apparaît au même niveau dans votre document de flux de travail que le nom de l'étape et l'action. C'est la base de la déclaration conditionnelle.

Vous pouvez spécifier jusqu'à quatre niveaux d'instructions conditionnelles, mais une seule instruction peut apparaître au niveau parent. Toutes les autres instructions de branchement, modificateurs conditionnels ou opérateurs conditionnels sont indentés à partir de là, un retrait par niveau.

Le schéma suivant indique le nombre maximum de niveaux imbriqués pour une instruction conditionnelle.

```
base:
  parent:
    - child (level 2)
        - child (level 3)
            child (level 4)
```

Attribut `if`

L'`if`attribut spécifie la déclaration conditionnelle en tant qu'attribut de document. Il s'agit du niveau zéro.

Au niveau des parents

Il s'agit du premier niveau d'imbrication pour les déclarations conditionnelles. Il ne peut y avoir qu'une seule déclaration à ce niveau. Si vous n'avez pas besoin de branchement ou de modificateurs, il peut s'agir d'un opérateur conditionnel sans instructions enfants. Ce niveau n'utilise pas la notation en tiret, à l'exception des opérateurs conditionnels.

Niveaux pour enfants

Les niveaux 2 à 4 sont considérés comme des niveaux pour enfants. Les instructions enfants peuvent inclure des instructions de branchement, des modificateurs conditionnels ou des opérateurs conditionnels.

Exemple : niveaux imbriqués

L'exemple suivant montre le nombre maximum de niveaux dans une instruction conditionnelle.

```
if:
  and:                            #first level
    - stringEquals: 'my_string'    #second level
      value: 'my_string'
    - and:                        #also second level
        - numberEquals: '1'        #third level
          value: 1
        - not:                    #also third level
            stringEquals: 'second_string'      #fourth level
            value: "diff_string"
```

Règles d'imbrication

- Chaque branche ou modificateur au niveau de l'enfant ouvre un nouveau niveau.
- Chaque niveau est en retrait.
- Il peut y avoir un maximum de quatre niveaux, dont une instruction, un modificateur ou un opérateur au niveau parent, et jusqu'à trois niveaux supplémentaires.

Exemples de déclarations conditionnelles

Ce groupe d'exemples montre différents aspects des déclarations conditionnelles.

Branchement : et

L'instruction de `and` branchement fonctionne sur une liste d'expressions qui sont des enfants de la branche, qui doivent toutes être évaluées à`true`. Image Builder évalue les expressions dans l'ordre dans lequel elles apparaissent dans la liste. Si une expression est évaluée à`false`, le traitement s'arrête et la branche est prise en compte`false`.

L'exemple suivant évalue à`true`, car les deux expressions sont évaluées à`true`.

```
if:
  and:
    - stringEquals: 'test_string'
      value: 'test_string'
    - numberEquals: 1
      value: 1
```

Branchement : ou

L'instruction de `or` branchement fonctionne sur une liste d'expressions qui sont des enfants de la branche, dont au moins une doit être évaluée à`true`. Image Builder évalue les expressions dans l'ordre dans lequel elles apparaissent dans la liste. Si une expression est évaluée à`true`, le traitement s'arrête et la branche est prise en compte`true`.

L'exemple suivant évalue à`true`, même si la première expression l'est`false`.

```
if:
  or:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
```

```
    - numberEquals: 1
      value: 1
```

Modificateur conditionnel : non

Le modificateur `not` conditionnel annule les instructions conditionnelles qui sont des enfants de la branche.

L'exemple suivant évalue le `true` moment où le `not` modificateur annule l'instruction `stringEquals` conditionnelle.

```
if:
  not:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
```

Déclaration conditionnelle : BooleanEquals

L'opérateur de `booleanEquals` comparaison compare les valeurs booléennes et renvoie true si les valeurs booléennes correspondent exactement.

L'exemple suivant détermine si cette option `collectImageScanFindings` est activée.

```
if:
  - booleanEquals: true
    value: '$.imagebuilder.collectImageScanFindings'
```

Déclaration conditionnelle : StringEquals

L'opérateur de `stringEquals` comparaison compare deux chaînes et renvoie true si les chaînes correspondent exactement. Si l'une des valeurs n'est pas une chaîne, Image Builder la convertit en chaîne avant de procéder à la comparaison.

L'exemple suivant compare la variable système de plate-forme pour déterminer si le flux de travail s'exécute sur une plate-forme Linux.

```
if:
  - stringEquals: 'Linux'
    value: '$.imagebuilder.platform'
```

Déclaration conditionnelle : NumberEquals

L'opérateur de `numberEquals` comparaison compare deux nombres et renvoie true s'ils sont égaux. Les nombres à comparer doivent respecter l'un des formats suivants.

- Entier
- Float
- Une chaîne qui correspond au modèle d'expression régulière suivant :`^-?[0-9]+(\.)?[0-9]+$`.

Les exemples de comparaison suivants évaluent tous à`true`.

```
if:
  # Value provider as a number
  numberEquals: 1
  value: '1'

  # Comparison value provided as a string
  numberEquals: '1'
  value: 1

  # Value provided as a string
  numberEquals: 1
  value: '1'

  # Floats are supported
  numberEquals: 5.0
  value: 5.0

  # Negative values are supported
  numberEquals: -1
  value: -1
```

Gérez la création d'images personnalisées dans Image Builder via un processus de pipeline reproductible

Les pipelines d'images Image Builder fournissent un cadre d'automatisation permettant de créer et de gérer des images personnalisées AMIs et des images de conteneur. Les pipelines fournissent les fonctionnalités suivantes :

- Assemblez l'image de base, créez et testez les composants pour la personnalisation de l'image, la configuration de l'infrastructure et les paramètres de distribution.
- Facilitez la planification des processus de maintenance automatisés `Schedule builder` à l'aide de l'assistant intégré à la console ou en saisissant des expressions cron pour les mises à jour récurrentes de vos images.
- Activez la détection des modifications pour l'image de base et les composants, afin d'ignorer automatiquement les builds planifiés en l'absence de modifications.
- Activez l'automatisation basée sur des règles via Amazon. EventBridge

> **Note**
>
> Pour plus d'informations sur l'utilisation de l' EventBridge API pour consulter ou modifier les règles, consultez le Amazon EventBridge API Reference. Pour plus d'informations sur l'utilisation EventBridge events des commandes dans les règles AWS CLI d'affichage ou de modification, voir les événements dans le manuel de référence des AWS CLI commandes.

Rubriques

- Configurer les paramètres d'exécution du pipeline pour les pipelines d'images
- Répertorier et afficher les détails du pipeline
- Création et mise à jour de pipelines d'images AMI
- Création et mise à jour de pipelines d'images de conteneurs
- Configuration des flux de travail du pipeline d'images dans Image Builder
- Utilisation EventBridge avec les pipelines Image Builder

Configurer les paramètres d'exécution du pipeline pour les pipelines d'images

Vous pouvez choisir l'une des options suivantes pour planifier l'exécution du pipeline :

Générateur de calendrier

Utilisez le Schedule Builder pour configurer l'exécution automatique et récurrente du pipeline. Vous définissez quand et à quelle fréquence votre pipeline doit fonctionner (jour, heure et fréquence). Le calendrier par défaut est hebdomadaire en fonction du jour et de l'heure de création du calendrier (UTC).

Expression cron

Exécutez automatiquement le pipeline avec une expression cron qui spécifie le calendrier. Pour plus d'informations sur la syntaxe cron utilisée par Image Builder, consultezUtiliser des expressions cron dans Image Builder.

Manuelle

Le pipeline ne fonctionne pas selon un calendrier. Dans la console, choisissez Exécuter le pipeline dans le menu Actions pour exécuter le pipeline. À partir de AWS CLI là, tu peux courir`start-image-pipeline-execution`.

Paramètres de dépendance

Pour les builds planifiés, vous pouvez choisir de toujours s'exécuter selon le calendrier ou d'ignorer l'exécution du pipeline, sauf en cas de mise à jour des dépendances, telle qu'une modification de l'image de base ou d'un composant utilisé dans la recette.

Désactiver automatiquement un pipeline défaillant

Pour les pipelines d'images qui s'exécutent selon un calendrier, vous pouvez configurer le nombre maximum de défaillances consécutives à autoriser (jusqu'à`10`) avant qu'Image Builder ne désactive automatiquement le pipeline. Si vous ne définissez pas de valeur`AutoDisablePolicy`, Image Builder utilise une valeur par défaut pour les échecs `5` consécutifs.

Auto-disable paramètres

Image Builder suit le nombre d'échecs consécutifs lors des exécutions de pipeline planifiées et effectue l'une des actions suivantes chaque fois qu'il s'exécute selon un calendrier :

- Si l'exécution du pipeline est réussie, le nombre de défaillances consécutives est remis à zéro.
- Si l'exécution du pipeline échoue, Image Builder augmente le nombre d'échecs consécutifs. Si le nombre de défaillances dépasse la limite définie dans le`AutoDisablePolicy`, Image Builder désactive le pipeline.

Le nombre de défaillances consécutives est également remis à zéro dans les conditions suivantes :

- Le pipeline s'exécute manuellement et réussit.
- La configuration du pipeline est mise à jour.

Si le pipeline s'exécute manuellement et échoue, le nombre reste le même. La prochaine exécution planifiée continue à s'incrémenter là où elle s'était arrêtée auparavant.

Configurer la journalisation du pipeline

Lorsque vous créez ou mettez à jour un pipeline d'images, vous pouvez configurer des groupes de CloudWatch journaux personnalisés pour la création d'images et les journaux de pipeline. Assurez-vous que votre rôle d'exécution de pipeline personnalisé dispose des autorisations suivantes pour créer et accéder aux ressources du groupe de journaux.

- journaux : CreateLogGroup
- journaux : CreateLogStream
- journaux : PutLogEvents

Groupes de journaux personnalisés

Pour utiliser des groupes de journaux personnalisés pour la création d'images ou l'exécution de pipelines, créez d'abord le groupe de CloudWatch journaux dans Logs. Pour plus d'informations, consultez la section Créer un groupe de CloudWatch journaux dans le guide de l'utilisateur Amazon Logs. Pour obtenir des conseils supplémentaires sur les exigences CreateLogGroupen matière de dénomination des groupes de journaux, consultez le manuel Amazon CloudWatch Logs API Reference.

Console

Spécifiez le groupe de journaux d'images ou le groupe de journaux de pipeline pour votre pipeline dans la section Configuration de la journalisation sous Paramètres avancés.

CLI

Spécifiez les champs suivants dans l'`logging-configuration`objet si vous utilisez un objet JSON pour la configuration :

- `imageLogGroupName`
- `pipelineLogGroupName`

Pour spécifier tous les paramètres directement dans la ligne de commande, voir create-image-pipeline dans la référence des commandes.AWS CLI

Image Builder utilise les groupes de journaux par défaut suivants si vous ne spécifiez aucun groupe de journaux personnalisé :

Journaux de création d'images

Image Builder écrit les journaux de compilation dans le groupe Image Builder CloudWatch Logs et le flux suivants :

LogGroup: `/aws/imagebuilder/`*`ImageName`*

LogStream (x.x.x/x) : *`ImageVersion/ImageBuildVersion`*

Journaux d'exécution du pipeline

Image Builder écrit les journaux d'exécution du pipeline dans le groupe et le flux Image Builder CloudWatch Logs suivants :

LogGroup: `/aws/imagebuilder/pipeline/`*`pipeline-name`*

LogStream: *`2025/09/01`* (la date d'exécution du pipeline au YYYY/MM/DD format)

Chaque journal de pipeline est ajouté au flux pour ce jour.

Exécuter un pipeline d'images manuellement

Si vous avez choisi l'option de planification manuelle pour votre pipeline, celui-ci ne s'exécutera que lorsque vous lancerez manuellement la construction. Si vous avez choisi l'une des options de planification automatique, vous pouvez également l'exécuter manuellement, entre les exécutions

planifiées régulièrement. Par exemple, si vous avez un pipeline qui s'exécute normalement une fois par mois, mais que vous devez intégrer une mise à jour de l'un de vos composants deux semaines après l'exécution précédente, vous pouvez choisir d'exécuter votre pipeline manuellement.

Console

Pour exécuter votre pipeline depuis la page des détails du pipeline de la console Image Builder, choisissez Run pipeline dans le menu Actions en haut de la page. Un message d'état apparaît en haut de la page pour vous informer que votre pipeline a démarré ou s'il y a une erreur.

1. Dans le coin supérieur gauche de la page des détails du pipeline, choisissez Exécuter le pipeline dans le menu Actions.
2. Vous pouvez voir l'état actuel de votre pipeline dans l'onglet Images de sortie, dans la colonne État.

AWS CLI

L'exemple suivant montre comment utiliser la start-image-pipeline-execution commande du AWS CLI pour démarrer un pipeline d'images manuellement. Lorsque vous exécutez cette commande, le pipeline crée et distribue une nouvelle image.

```
aws imagebuilder start-image-pipeline-execution --image-pipeline-arn
 arn:aws:imagebuilder:us-west-2:111122223333:image-pipeline/my-example-pipeline
```

Pour voir quelles ressources sont créées lors de l'exécution du pipeline de génération, voirRessources créées.

Utiliser des expressions cron dans Image Builder

Utilisez des expressions cron pour EC2 Image Builder afin de configurer une fenêtre temporelle afin d'actualiser votre image avec des mises à jour qui s'appliquent à l'image de base et aux composants de votre pipeline. La fenêtre temporelle pour l'actualisation de votre pipeline commence à l'heure que vous avez définie dans l'expression cron. Vous pouvez définir l'heure dans votre expression cron à la minute près. La construction de votre pipeline peut s'exécuter à l'heure de début ou après.

L'exécution de votre build peut parfois prendre quelques secondes, voire une minute.

> **Note**
>
> Les expressions Cron utilisent le fuseau horaire UTC (Universal Coordinated Time) par défaut, ou vous pouvez spécifier le fuseau horaire. Pour plus d'informations sur l'heure UTC et pour connaître le décalage correspondant à votre fuseau horaire, consultez la section Abréviations des fuseaux horaires — Liste mondiale.

Valeurs prises en charge pour les expressions cron dans Image Builder

EC2 Image Builder utilise un format cron composé de six champs obligatoires. Chacune est séparée des autres par un espace entre les deux, sans espaces de début ou de fin :

```
<Minute> <Hour> <Day> <Month> <Day of the week> <Year>
```

Le tableau suivant montre les valeurs prises en charge pour les entrées cron obligatoires.

Valeurs prises en charge pour les expressions cron

Champ	Valeurs	Caractères génériques
Minute	`0-59`	`, - * /`
Heure	`0-23`	`, - * /`
jour	`1-31`	`, - * ? / L W`
Mois	`1-12` ou `jan-dec`	`, - * /`
Jour de la semaine	`1-7` ou `sun-sat`	`, - * ? L #`
Année	`1970-2199`	`, - * /`

Caractères génériques

Le tableau suivant décrit comment Image Builder utilise des caractères génériques pour les expressions cron. N'oubliez pas que le démarrage de la compilation peut prendre jusqu'à une minute après l'heure spécifiée.

Caractères génériques pris en charge pour les expressions cron

Caractère générique	Description
,	Le caractère générique , (virgule) inclut des valeurs supplémentaires. Dans le champ Mois, `jan,feb,mar` inclut janvier, février et mars.
-	Le caractère générique - (tiret) spécifie des plages. Dans le champ Jour du mois, `1-15` inclut les jours 1 à 15 du mois spécifié.
*	Le caractère générique * (astérisque) inclut toutes les valeurs valides du champ.
?	Le caractère générique ? (point d'interrogation) indique que la valeur du champ dépend d'un autre paramètre. Dans le cas du jour et des Day-of-week champs, lorsque l'un est spécifié ou inclut toutes les valeurs possibles (`*`), l'autre doit être ? a. Vous ne pouvez pas spécifier les deux. Par exemple, si vous entrez un 7 dans le champ Jour (lancez le build le septième jour du mois), le Day-of-week poste doit contenir un?.
/	Le caractère générique / (barre oblique) spécifie les incréments. Par exemple, si vous souhaitez que votre build soit exécuté tous les deux jours, entrez `*/2` dans le champ jour.
L	Le caractère générique L dans l'un ou l'autre des champs du jour indique le dernier jour : du 28 au 31 pour le jour du mois, selon le mois, ou le dimanche, pour le jour de la semaine.
W	Le caractère générique W dans le champ spécifie un jour de la semaine. Day-of-month Dans le Day-of-month champ, si vous entrez un nombre avant le`W`, cela signifie que vous

Caractère générique	Description
	souhaitez cibler le jour de la semaine le plus proche de ce jour. Par exemple, si vous le spécifiez`3W`, vous souhaitez que votre build soit exécuté le jour de la semaine le plus proche du troisième jour du mois.
#	Le # (hachage) n'est autorisé que pour le champ du jour de la semaine et doit être suivi d'un nombre compris entre 1 et 5. Le numéro indique les semaines d'un mois donné qui s'appliquent à l'exécution de la version. Par exemple, si vous souhaitez que votre build soit exécuté le deuxième vendredi de chaque mois, utilisez `fri#2` le champ du jour de la semaine.

Restrictions

- Vous ne pouvez pas spécifier les champs Day-of-month et Day-of-week de la même expression cron. Si vous spécifiez une valeur ou * dans l'un de ces champs, vous devez utiliser un ? dans l'autre.
- Les expressions cron qui entraînent des fréquences d'une rapidité supérieure à une minute ne sont pas prises en charge.

Exemples d'expressions cron dans Image Builder

Les expressions Cron sont saisies différemment pour la console Image Builder et pour l'API ou la CLI. Pour voir des exemples, choisissez l'onglet qui s'applique à vous.

Image Builder console

Les exemples suivants montrent les expressions cron que vous pouvez saisir dans la console pour votre calendrier de génération. L'heure UTC est spécifiée à l'aide d'une horloge de 24 heures.

Fonctionne tous les jours à 10 h 00 (UTC)

```
0 10 * * ? *
```

Fonctionne tous les jours à 12 h 15 (UTC)

```
15 12 * * ? *
```

Fonctionne tous les jours à minuit (UTC)

```
0 0 * * ? *
```

Ouvert à 10 h 00 (UTC) tous les matins de la semaine

```
0 10 ? * 2-6 *
```

Ouvert à 18 h (UTC) tous les soirs de semaine

```
0 18 ? * mon-fri *
```

Ouvert à 8 h 00 (UTC) le premier jour de chaque mois

```
0 8 1 * ? *
```

Organisé le deuxième mardi de chaque mois à 22 h 30 (UTC)

```
30 22 ? * tue#2 *
```

> **Tip**
>
> Si vous ne souhaitez pas que votre tâche de pipeline s'étende jusqu'au lendemain pendant son exécution, assurez-vous de prendre en compte le temps de construction lorsque vous spécifiez l'heure de début.

API/CLI

Les exemples suivants montrent les expressions cron que vous pouvez saisir pour votre calendrier de construction à l'aide de commandes CLI ou de requêtes d'API. Seule l'expression cron est affichée.

Fonctionne tous les jours à 10 h 00 (UTC)

```
cron(0 10 * * ? *)
```

Fonctionne tous les jours à 12 h 15 (UTC)

```
cron(15 12 * * ? *)
```

Fonctionne tous les jours à minuit (UTC)

```
cron(0 0 * * ? *)
```

Ouvert à 10 h 00 (UTC) tous les matins de la semaine

```
cron(0 10 ? * 2-6 *)
```

Ouvert à 18 h 00 (UTC) tous les soirs de semaine

```
cron(0 18 ? * mon-fri *)
```

Ouvert à 8 h 00 (UTC) le premier jour de chaque mois

```
cron(0 8 1 * ? *)
```

Organisé le deuxième mardi de chaque mois à 22 h 30 (UTC)

```
cron(30 22 ? * tue#2 *)
```

> **Tip**
>
> Si vous ne souhaitez pas que votre tâche de pipeline s'étende jusqu'au lendemain pendant son exécution, assurez-vous de prendre en compte le temps de construction lorsque vous spécifiez l'heure de début.

Expressions de taux dans Image Builder

Une expression de fréquence démarre au moment où vous créez la règle d'événement planifié, puis s'exécute selon le calendrier défini.

Les expressions de fréquence comportent deux champs obligatoires. Ces champs sont séparés par un espace.

Syntaxe

```
rate(value unit)
```

value

Nombre positif.

unité

Unité de temps. Des unités différentes sont nécessaires pour les valeurs de 1 (par exemple `minute`) et les valeurs supérieures à 1, (par exemple, `minutes`).

Valeurs valides : minute | minutes | heure | heures | jour | jours

Restrictions

Si la valeur est égale à 1, l'unité doit être au singulier. De même, pour les valeurs supérieures à 1, l'unité doit être au pluriel. Par exemple, `rate(1 hours)` et `rate(5 hour)` ne sont pas valides, mais `rate(1 hour)` et `rate(5 hours)` sont valides.

Répertorier et afficher les détails du pipeline

Cette section décrit les différentes manières de trouver des informations et d'afficher les détails de vos pipelines d'images EC2 Image Builder.

Détails du pipeline

- Répertorier les pipelines d'images à partir du AWS CLI
- Obtenez des informations détaillées sur le pipeline d'images à partir du AWS CLI

Répertorier les pipelines d'images à partir du AWS CLI

L'exemple suivant montre comment utiliser la list-image-pipelines commande figurant dans la liste AWS CLI de tous vos pipelines d'images.

```
aws imagebuilder list-image-pipelines
```

Obtenez des informations détaillées sur le pipeline d'images à partir du AWS CLI

L'exemple suivant montre comment utiliser la get-image-pipeline commande du AWS CLI pour obtenir des informations sur un pipeline d'images via son ARN.

```
aws imagebuilder get-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

Création et mise à jour de pipelines d'images AMI

Vous pouvez configurer, configurer et gérer des pipelines d'images AMI depuis la console Image Builder, via l'API Image Builder, les kits SDK ou le AWS CLI. Dans la console, vous pouvez utiliser l'assistant de console Créer un pipeline d'images pour vous guider tout au long des étapes suivantes.

- Spécifiez les détails du pipeline tels que le nom, la description et les balises de ressource.
- Configurez la planification du pipeline et les paramètres de journalisation par défaut. Pour l'exécution planifiée du pipeline, vous pouvez définir le nombre d'échecs consécutifs autorisés avant qu'Image Builder ne désactive le pipeline.
- Sélectionnez une recette d'image AMI qui inclut une image de base issue d'images gérées par Amazon à démarrage rapide, d'images que vous avez créées ou partagées avec vous, ou d'images auxquelles vous êtes abonné via le. AWS Marketplace La recette inclut également des composants qui exécutent les tâches suivantes sur les instances EC2 utilisées par Image Builder pour créer votre image :
  - Ajouter et supprimer des logiciels
  - Personnalisez les paramètres et les scripts
  - Exécuter les tests sélectionnés
- Spécifiez des flux de travail pour configurer les étapes de création et de test des images exécutées par votre pipeline.
- Définissez la configuration de l'infrastructure de votre pipeline à l'aide de paramètres par défaut ou de paramètres que vous configurez vous-même. La configuration inclut le type d'instance et la paire de clés à utiliser pour votre image, les paramètres de sécurité et de réseau, les paramètres de stockage des journaux et de dépannage, ainsi que les notifications SNS.

  Il s'agit d'une étape facultative. Image Builder utilise les paramètres par défaut pour la configuration de votre infrastructure si vous ne la définissez pas vous-même.
- Définissez les paramètres de distribution pour diffuser vos images vers les AWS régions et les comptes de destination. Vous pouvez spécifier une clé KMS pour le chiffrement, configurer le partage des AMI ou la configuration des licences, ou configurer un modèle de lancement pour les AMI que vous distribuez.

Il s'agit d'une étape facultative. Si vous ne définissez pas vous-même la configuration, Image Builder utilise le nom par défaut pour votre AMI de sortie et distribue l'AMI à la région source. La région source est la région dans laquelle vous gérez le pipeline.

Pour plus d'informations et un didacticiel étape par étape sur l'utilisation de l'assistant de console Create image pipeline avec les valeurs par défaut lorsqu'elles sont fournies, consultezTutoriel : Création d'un pipeline d'images avec une AMI de sortie à partir de l'assistant de console Image Builder.

Table des matières

- Créez un pipeline d'images AMI à partir du AWS CLI
- Mettre à jour les pipelines d'images AMI depuis la console
- Mettez à jour les pipelines d'images AMI à partir du AWS CLI

Créez un pipeline d'images AMI à partir du AWS CLI

Pour créer un pipeline d'images à partir de AWS CLI, exécutez la create-image-pipeline commande avec les options de configuration qui s'appliquent à votre pipeline. Vous avez la possibilité de créer un fichier JSON contenant l'ensemble de la configuration de votre pipeline ou de spécifier la configuration lors de l'exécution. Cette section utilise la méthode du fichier de configuration JSON pour simplifier la commande.

La fréquence à laquelle votre pipeline crée une nouvelle image pour intégrer les mises à jour en attente à partir de votre image de base et de vos composants dépend de `schedule` ce que vous avez configuré. A `schedule` possède les attributs suivants :

- `scheduleExpression`— Définit le calendrier d'exécution de votre pipeline afin d'évaluer `pipelineExecutionStartCondition` et de déterminer s'il doit démarrer une construction. Le calendrier est configuré avec des expressions cron. Pour plus d'informations sur la mise en forme d'une expression cron dans Image Builder, consultezUtiliser des expressions cron dans Image Builder.
- `pipelineExecutionStartCondition`— Détermine si votre pipeline doit démarrer la construction. Les valeurs valides sont les suivantes :
  - `EXPRESSION_MATCH_ONLY`— votre pipeline créera une nouvelle image chaque fois que l'expression cron correspond à l'heure actuelle.

- `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`— votre pipeline crée une nouvelle image uniquement lorsque l'expression du calendrier correspond et que des mises à jour sont en attente de votre image de base ou de vos composants. Si aucune mise à jour n'est en attente, la compilation est ignorée. Pour créer immédiatement sans attendre les mises à jour, exécutez le pipeline manuellement.

Lorsque vous exécutez la create-image-pipeline commande dans le AWS CLI, de nombreuses ressources de configuration sont facultatives. Certaines ressources sont toutefois soumises à des exigences conditionnelles, en fonction du type d'image créé par le pipeline. Les identificateurs de ressources suivants sont requis pour les pipelines d'images AMI :

- Recette d'image ARN
- ARN de configuration de l'infrastructure

Exemple : création d'une image Windows 2019

Cet exemple configure un pipeline dont l'exécution est programmée une fois par semaine le dimanche. Le fichier de configuration présenté à la première étape utilise les ressources existantes pour la configuration de la recette, de l'infrastructure et de la distribution de l'image, ainsi que d'autres paramètres pour créer une image Windows 2019.

1. Création d'un fichier de configuration (facultatif)

   Cet exemple utilise un fichier de configuration nommé `create-image-pipeline.json` pour configurer les paramètres en un seul endroit. Vous pouvez également utiliser les options de ligne de commande lorsque vous exécutez la commande pour spécifier tous les détails affichés ici dans le fichier de configuration.

```
{
 "name": "ExampleWindows2019Pipeline",
 "description": "Builds Windows 2019 Images",
 "enhancedImageMetadataEnabled": true,
 "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-infrastructure-configuration",
```

```
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

2. Exécutez la commande pour créer une image

   Cet exemple utilise le fichier de configuration créé lors de la première étape comme entrée de la `create-image-pipeline` commande. Vous pouvez également spécifier les paramètres et les ressources de votre pipeline directement lorsque vous exécutez la commande. Pour plus d'informations, consultez create-image-pipeline dans la Référence. AWS CLI

   ```
   aws imagebuilder create-image-pipeline --cli-input-json file://create-image-
   pipeline.json
   ```

Exemple : création d'un pipeline avec numérisation d'images et flux de travail personnalisés

Cet exemple configure un pipeline qui est vérifié tous les sept jours et qui s'exécute uniquement lorsque des mises à jour des dépendances sont disponibles. Le fichier de configuration utilise les ressources existantes pour la recette et l'infrastructure de l'image. Il permet l'analyse des

vulnérabilités à l'aide d'Amazon Inspector. La configuration spécifie également des flux de travail personnalisés dotés d'un rôle d'exécution. Des groupes parallèles exécutent simultanément des tests de sécurité et des tests fonctionnels. Les journaux de pipeline sont envoyés à des groupes de CloudWatch journaux personnalisés.

1. Création d'un fichier de configuration

   Créez un fichier JSON nommé`create-pipeline-with-workflows.json`. Ce fichier définit la configuration du pipeline avec la numérisation d'images activée et les flux de travail de test personnalisés. Remplacez les valeurs des espaces réservés par les ARN de vos propres ressources.

```
{
 "name": "MyPipelineWithScanning",
 "description": "Pipeline with vulnerability scanning and custom workflows",
 "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-recipe/1.0.0",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-infra-config",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-dist-config",
 "imageScanningConfiguration": {
  "imageScanningEnabled": true
 },
 "workflows": [
  {
   "workflowArn": "arn:aws:imagebuilder:us-west-2:123456789012:workflow/build/my-build-workflow/1.0.0"
  },
  {
   "workflowArn": "arn:aws:imagebuilder:us-west-2:123456789012:workflow/test/my-security-scan/1.0.0",
   "onFailure": "ABORT",
   "parallelGroup": "security"
  },
  {
   "workflowArn": "arn:aws:imagebuilder:us-west-2:123456789012:workflow/test/my-compliance-check/1.0.0",
   "onFailure": "ABORT",
   "parallelGroup": "security"
  },
  {
```

```
    "workflowArn": "arn:aws:imagebuilder:us-west-2:123456789012:workflow/test/my-
functional-test/1.0.0",
    "onFailure": "CONTINUE",
    "parallelGroup": "functional"
   },
   {
    "workflowArn": "arn:aws:imagebuilder:us-west-2:123456789012:workflow/test/my-
performance-test/1.0.0",
    "onFailure": "CONTINUE",
    "parallelGroup": "functional"
   }
  ],
  "executionRole": "arn:aws:iam::123456789012:role/ImageBuilderExecutionRole",
  "loggingConfiguration": {
   "imageLogGroupName": "/aws/imagebuilder/my-pipeline-image-logs",
   "pipelineLogGroupName": "/aws/imagebuilder/my-pipeline-execution-logs"
  },
  "imageTestsConfiguration": {
   "imageTestsEnabled": true,
   "timeoutMinutes": 60
  },
  "schedule": {
   "scheduleExpression": "rate(7 days)",
   "pipelineExecutionStartCondition":
  "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
  },
  "status": "ENABLED"
}
```

Note

- Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
- Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

2. Exécutez la commande

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-pipeline-
with-workflows.json
```

Mettre à jour les pipelines d'images AMI depuis la console

Après avoir créé un pipeline d'images Image Builder pour votre image AMI, vous pouvez modifier la configuration de l'infrastructure et les paramètres de distribution depuis la console Image Builder.

Pour mettre à jour un pipeline d'images avec une nouvelle recette d'image, vous devez utiliser le AWS CLI. Pour plus d'informations, consultez Mettez à jour les pipelines d'images AMI à partir du AWS CLI dans ce guide.

Choisissez un pipeline Image Builder existant

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Pour consulter la liste des pipelines d'images créés sous votre compte, choisissez Pipelines d'images dans le volet de navigation.

> **Note**
>
> La liste des pipelines d'images inclut un indicateur pour le type d'image de sortie créée par le pipeline : AMI ouDocker.

3. Pour afficher les détails ou modifier un pipeline, cliquez sur le lien Nom du pipeline. Cela ouvre la vue détaillée du pipeline.

> **Note**
>
> Vous pouvez également cocher la case à côté du nom du pipeline, puis choisir Afficher les détails.

Détails du pipeline

La page des détails du pipeline comprend les sections suivantes :

Résumé

La section en haut de la page récapitule les principaux détails du pipeline qui sont visibles lorsque l'un des onglets de détail est ouvert. Les détails affichés dans cette section sont modifiables uniquement dans leurs onglets de détails respectifs.

Onglets détaillés

- Images de sortie : affiche les images de sortie produites par le pipeline.
- Recette illustrée — Affiche les détails de la recette. Une fois que vous avez créé une recette, vous ne pouvez pas la modifier. Vous devez créer une nouvelle version de la recette à partir de la page des recettes Image de la console Image Builder ou à l'aide des commandes Image Builder du AWS CLI. Pour de plus amples informations, veuillez consulter Gérer les recettes dans Image Builder.
- Configuration de l'infrastructure : affiche des informations modifiables pour configurer l'infrastructure de votre pipeline de build.
- Paramètres de distribution : affiche des informations modifiables pour la distribution des AMI.
- EventBridge règles : pour le bus d'événements sélectionné, affiche EventBridge les règles qui ciblent le pipeline actuel. Inclut la création d'un bus d'événements et la création d'actions de règles liées à la EventBridge console. Pour plus d'informations sur cet onglet, consultezUtiliser EventBridge.

Modifier la configuration de l'infrastructure de votre pipeline

La configuration de l'infrastructure inclut les informations suivantes que vous pouvez modifier après avoir créé le pipeline :

- Description de la configuration de votre infrastructure.
- Le rôle IAM à associer au profil d'instance.
- AWS infrastructure, y compris le type d'instance et une rubrique SNS pour les notifications.
- VPC, sous-réseau et groupes de sécurité.
- Paramètres de résolution des problèmes, notamment l'arrêt de l'instance en cas de panne, la paire de clés pour la connexion et l'emplacement facultatif du compartiment S3 pour les journaux d'instance.

Pour modifier la configuration de l'infrastructure depuis la page des détails du pipeline, procédez comme suit :

1. Choisissez l'onglet Configuration de l'infrastructure.
2. Choisissez Modifier dans le coin supérieur droit du panneau des détails de configuration.
3. Lorsque vous êtes prêt à enregistrer les mises à jour que vous avez apportées à la configuration de votre infrastructure, choisissez Enregistrer les modifications.

Modifier les paramètres de distribution de votre pipeline

Les paramètres de distribution incluent les informations suivantes que vous pouvez modifier après avoir créé le pipeline :

- Description de votre configuration de distribution.
- Paramètres régionaux pour les régions dans lesquelles vous distribuez votre image. La région 1 correspond par défaut à la région dans laquelle vous avez créé le pipeline. Vous pouvez ajouter des régions à distribuer à l'aide du bouton Ajouter une région, et vous pouvez supprimer toutes les régions à l'exception de la région 1.

  Les paramètres régionaux incluent :

  - Région cible
  - Le nom de l'AMI de sortie
  - Autorisations de lancement et comptes avec lesquels les partager
  - Licences associées (configurations de licences associées)

  > **Note**
  >
  > Les paramètres du gestionnaire de licences ne seront pas répliqués entre les AWS régions qui doivent être activées sur votre compte, par exemple entre les régions `ap-east-1` (Hong Kong) et `me-south-1` (Bahreïn).

Pour modifier vos paramètres de distribution depuis la page des détails du pipeline, procédez comme suit :

1. Choisissez l'onglet Paramètres de distribution.

2. Choisissez Modifier dans le coin supérieur droit du panneau Détails de la distribution.
3. Lorsque vous êtes prêt à enregistrer vos mises à jour, choisissez Enregistrer les modifications.

Modifiez le calendrier de construction de votre pipeline

La page Modifier le pipeline inclut les informations suivantes que vous pouvez modifier après avoir créé le pipeline :

- Description de votre pipeline.
- Collecte de métadonnées améliorée. Cette option est activée par défaut. Pour la désactiver, désactivez la case à cocher Activer la collecte améliorée de métadonnées.
- Le calendrier de construction de votre pipeline. Vous pouvez modifier vos options de planification et tous les paramètres ici.

Pour modifier votre pipeline depuis la page des détails du pipeline, procédez comme suit :

1. Dans le coin supérieur droit de la page des détails du pipeline, choisissez Actions, puis Modifier le pipeline.
2. Lorsque vous êtes prêt à enregistrer vos mises à jour, choisissez Enregistrer les modifications.

> **Note**
>
> Pour plus d'informations sur la planification de votre build à l'aide d'expressions cron, consultezUtiliser des expressions cron dans Image Builder.

Mettez à jour les pipelines d'images AMI à partir du AWS CLI

Vous pouvez mettre à jour un pipeline d'images AMI à l'aide d'un fichier JSON comme entrée de la update-image-pipeline commande du AWS CLI. Pour configurer le fichier JSON, vous devez disposer d'Amazon Resource Names (ARN) pour référencer les ressources existantes suivantes :

- Pipeline d'images à mettre à jour
- Recette imagée
- Configuration de l'infrastructure
- Paramètres de distribution

Vous pouvez mettre à jour un pipeline d'images AMI à l'aide de la update-image-pipeline commande AWS CLI suivante :

> **Note**
>
> La update-image-pipeline commande remplace l'intégralité de la configuration du pipeline. Vous devez spécifier toutes les propriétés requises dans la demande de mise à jour. Incluez toutes les propriétés, qu'il s'agisse des propriétés que vous souhaitez modifier ou des propriétés qui doivent rester les mêmes. Les propriétés que vous omettez sont rétablies aux valeurs par défaut ou supprimées.
>
> Pour éviter la suppression involontaire de paramètres existants, utilisez la get-image-pipeline commande pour récupérer la configuration actuelle. Modifiez ensuite uniquement les champs que vous souhaitez modifier :
>
> ```
> aws imagebuilder get-image-pipeline --image-pipeline-arn
>  arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-pipeline
> ```
>
> Utilisez la sortie comme base pour votre fichier JSON de demande de mise à jour.

1. Créer un fichier JSON d'entrée CLI

   Utilisez votre outil d'édition de fichiers préféré pour créer un fichier JSON avec les clés suivantes, ainsi que des valeurs valides pour votre environnement. Cet exemple utilise un fichier nommé`create-component.json` :

   ```
    {
    "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
   pipeline/my-example-pipeline",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
   example-recipe/2019.12.08",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
   west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
   configuration",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
   west-2:123456789012:distribution-configuration/my-example-distribution-
   configuration",
    "imageTestsConfiguration": {
     "imageTestsEnabled": true,
     "timeoutMinutes": 120
   ```

```
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "DISABLED"
}
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-
pipeline.json
```

Création et mise à jour de pipelines d'images de conteneurs

Vous pouvez configurer, configurer et gérer des pipelines d'images de conteneurs à l'aide de la console Image Builder, via l'API Image Builder ou à l'aide imagebuilder des commandes du AWS CLI. L'assistant de console Create Image Pipeline fournit des artefacts de départ et vous guide pas à pas pour :

- Sélectionnez une image de base parmi les images gérées à démarrage rapide, les référentiels Amazon ECR ou Docker Hub
- Ajouter et supprimer des logiciels
- Personnalisez les paramètres et les scripts
- Exécuter les tests sélectionnés
- Créez un Dockerfile à l'aide de variables de construction préconfigurées.
- Diffuser des images dans les AWS régions

Pour plus d'informations et un didacticiel étape par étape sur l'utilisation de l'assistant de console Create Image Pipeline, consultezTutoriel : Création d'un pipeline d'images avec une image de conteneur Docker en sortie à partir de l'assistant de console Image Builder.

Table des matières

- Créez un pipeline d'images de conteneurs à partir du AWS CLI
- Mettre à jour un pipeline d'images de conteneur depuis la console
- Mettez à jour les pipelines d'images de conteneurs à partir du AWS CLI

Créez un pipeline d'images de conteneurs à partir du AWS CLI

Pour créer un pipeline d'images à partir de AWS CLI, exécutez la create-image-pipeline commande avec les options de configuration qui s'appliquent à votre pipeline. Vous avez la possibilité de créer un fichier JSON contenant l'ensemble de la configuration de votre pipeline ou de spécifier la configuration lors de l'exécution. Cette section utilise la méthode du fichier de configuration JSON pour simplifier la commande.

La fréquence à laquelle votre pipeline crée une nouvelle image pour intégrer les mises à jour en attente à partir de votre image de base et de vos composants dépend de `schedule` ce que vous avez configuré. A `schedule` possède les attributs suivants :

- `scheduleExpression`— Définit le calendrier d'exécution de votre pipeline afin d'évaluer `pipelineExecutionStartCondition` et de déterminer s'il doit démarrer une construction. Le calendrier est configuré avec des expressions cron. Pour plus d'informations sur la mise en forme d'une expression cron dans Image Builder, consultezUtiliser des expressions cron dans Image Builder.
- `pipelineExecutionStartCondition`— Détermine si votre pipeline doit démarrer la construction. Les valeurs valides sont les suivantes :
  - `EXPRESSION_MATCH_ONLY`— votre pipeline créera une nouvelle image chaque fois que l'expression cron correspond à l'heure actuelle.
  - `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`— votre pipeline crée une nouvelle image uniquement lorsque l'expression du calendrier correspond et que des mises à jour sont en attente de votre image de base ou de vos composants. Si aucune mise à jour n'est en attente, la compilation est ignorée. Pour créer immédiatement sans attendre les mises à jour, exécutez le pipeline manuellement.

Lorsque vous exécutez la create-image-pipeline commande dans le AWS CLI, de nombreuses ressources de configuration sont facultatives. Certaines ressources sont toutefois soumises à des exigences conditionnelles, en fonction du type d'image créé par le pipeline. Les ressources suivantes sont requises pour les pipelines d'images de conteneurs :

- Recette de récipient ARN
- ARN de configuration de l'infrastructure

Si vous n'incluez pas de ressource de configuration de distribution lorsque vous exécutez la create-image-pipeline commande, l'image de sortie est stockée dans le référentiel ECR que vous spécifiez comme référentiel cible dans votre recette de conteneur dans la région où vous exécutez la commande. Si vous incluez une ressource de configuration de distribution pour votre pipeline, le référentiel cible que vous avez spécifié pour la première région de la distribution est utilisé.

1. Créer un fichier JSON d'entrée CLI

   Utilisez votre outil d'édition de fichiers préféré pour créer un fichier JSON avec les clés suivantes, ainsi que des valeurs valides pour votre environnement. Cet exemple utilise un fichier nommé`create-image-pipeline.json` :

```
{
 "name": "MyWindows2019Pipeline",
 "description": "Builds Windows 2019 Images",
 "enhancedImageMetadataEnabled": true,
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-example-recipe/2020.12.03",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
```

```
 },
 "status": "ENABLED"
}
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-image-
pipeline.json
```

Exemple : création d'un pipeline de conteneurs avec numérisation d'images et flux de travail personnalisés

Cet exemple configure un pipeline de conteneurs qui est vérifié tous les lundis à 10 h 00 UTC et s'exécute uniquement lorsque des mises à jour des dépendances sont disponibles. Le fichier de configuration utilise les ressources existantes pour la recette et l'infrastructure du conteneur. Il permet l'analyse des vulnérabilités grâce à l'intégration d'Amazon Inspector et d'ECR. La configuration spécifie également des flux de travail personnalisés dotés d'un rôle d'exécution. Des groupes parallèles exécutent simultanément des tests de sécurité et des tests fonctionnels. Les journaux de pipeline sont envoyés à des groupes de CloudWatch journaux personnalisés.

1. Création d'un fichier de configuration

   Créez un fichier JSON nommé`create-container-pipeline-with-workflows.json`. Ce fichier définit la configuration du pipeline de conteneurs avec la numérisation d'images activée et les flux de travail de test personnalisés. Remplacez les valeurs des espaces réservés par les ARN de vos propres ressources.

```
{
 "name": "MyContainerPipelineWithScanning",
```

```
 "description": "Container pipeline with vulnerability scanning and custom
 workflows",
 "containerRecipeArn": "arn:aws:imagebuilder:us-east-1:123456789012:container-
recipe/my-container-recipe/1.0.0",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
east-1:123456789012:infrastructure-configuration/my-infra-config",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
east-1:123456789012:distribution-configuration/my-dist-config",
 "imageScanningConfiguration": {
  "imageScanningEnabled": true,
  "ecrConfiguration": {
   "repositoryName": "my-scanning-repo",
   "containerTags": ["scan-latest"]
  }
 },
 "workflows": [
  {
   "workflowArn": "arn:aws:imagebuilder:us-east-1:123456789012:workflow/build/my-
build-workflow/1.0.0"
  },
  {
   "workflowArn": "arn:aws:imagebuilder:us-east-1:123456789012:workflow/test/my-
security-scan/1.0.0",
   "onFailure": "ABORT",
   "parallelGroup": "security"
  },
  {
   "workflowArn": "arn:aws:imagebuilder:us-east-1:123456789012:workflow/test/my-
compliance-check/1.0.0",
   "onFailure": "ABORT",
   "parallelGroup": "security"
  },
  {
   "workflowArn": "arn:aws:imagebuilder:us-east-1:123456789012:workflow/test/my-
functional-test/1.0.0",
   "onFailure": "CONTINUE",
   "parallelGroup": "functional"
  },
  {
   "workflowArn": "arn:aws:imagebuilder:us-east-1:123456789012:workflow/test/my-
performance-test/1.0.0",
   "onFailure": "CONTINUE",
   "parallelGroup": "functional"
  }
```

```
 ],
 "executionRole": "arn:aws:iam::123456789012:role/ImageBuilderExecutionRole",
 "loggingConfiguration": {
  "imageLogGroupName": "/aws/imagebuilder/my-container-pipeline-image-logs",
  "pipelineLogGroupName": "/aws/imagebuilder/my-container-pipeline-execution-logs"
 },
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 120
 },
 "schedule": {
  "scheduleExpression": "cron(0 10 ? * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

2. Exécutez la commande

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-container-
pipeline-with-workflows.json
```

Mettre à jour un pipeline d'images de conteneur depuis la console

Après avoir créé un pipeline d'images de conteneur Image Builder pour votre image Docker, vous pouvez modifier la configuration de l'infrastructure et les paramètres de distribution depuis la console Image Builder.

Pour mettre à jour un pipeline d'images de conteneur avec une nouvelle recette de conteneur, vous devez utiliser le AWS CLI. Pour plus d'informations, consultez Mettez à jour les pipelines d'images de conteneurs à partir du AWS CLI dans ce guide.

Choisissez un pipeline d'images Docker Image Builder existant

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Pour consulter la liste des pipelines d'images créés sous votre compte, choisissez Pipelines d'images dans le volet de navigation.

   > **Note**
   >
   > La liste des pipelines d'images inclut un indicateur pour le type d'image de sortie créée par le pipeline : AMI ouDocker.

3. Pour afficher les détails ou modifier un pipeline, cliquez sur le lien Nom du pipeline. Cela ouvre la vue détaillée du pipeline.

   > **Note**
   >
   > Vous pouvez également cocher la case à côté du nom du pipeline, puis choisir Afficher les détails.

Détails du pipeline

La page détaillée du pipeline EC2 Image Builder comprend les sections suivantes :

Résumé

La section en haut de la page récapitule les principaux détails du pipeline qui sont visibles lorsque l'un des onglets de détail est ouvert. Les détails affichés dans cette section sont modifiables uniquement dans leurs onglets de détails respectifs.

Onglets détaillés

- Images de sortie : affiche les images de sortie produites par le pipeline.
- Recette du contenant — Affiche les détails de la recette. Une fois que vous avez créé une recette, vous ne pouvez pas la modifier. Vous devez créer une nouvelle version de la recette depuis la

page des recettes du conteneur. Pour de plus amples informations, veuillez consulter Création d'une nouvelle version d'une recette de contenant.

- Configuration de l'infrastructure : affiche des informations modifiables pour configurer l'infrastructure de votre pipeline de build.
- Paramètres de distribution : affiche des informations modifiables pour la distribution d'images Docker.
- EventBridge règles : pour le bus d'événements sélectionné, affiche EventBridge les règles qui ciblent le pipeline actuel. Inclut la création d'un bus d'événements et la création d'actions de règles liées à la EventBridge console. Pour plus d'informations sur cet onglet, consultezUtiliser EventBridge.

Modifier la configuration de l'infrastructure de votre pipeline

La configuration de l'infrastructure inclut les informations suivantes que vous pouvez modifier après avoir créé le pipeline :

- Description de la configuration de votre infrastructure.
- Le rôle IAM à associer au profil d'instance.
- AWS infrastructure, y compris le type d'instance et une rubrique SNS pour les notifications.
- VPC, sous-réseau et groupes de sécurité.
- Paramètres de résolution des problèmes, notamment l'arrêt de l'instance en cas de panne, la paire de clés pour la connexion et l'emplacement facultatif du compartiment S3 pour les journaux d'instance.

Pour modifier la configuration de l'infrastructure depuis la page des détails du pipeline, procédez comme suit :

1. Choisissez l'onglet Configuration de l'infrastructure.
2. Choisissez Modifier dans le coin supérieur droit du panneau des détails de configuration.
3. Lorsque vous êtes prêt à enregistrer les mises à jour que vous avez apportées à la configuration de votre infrastructure, choisissez Enregistrer les modifications.

Modifier les paramètres de distribution de votre pipeline

Les paramètres de distribution incluent les informations suivantes que vous pouvez modifier après avoir créé le pipeline :

- Description de vos paramètres de distribution.
- Paramètres régionaux pour les régions dans lesquelles vous distribuez votre image. La région 1 correspond par défaut à la région dans laquelle vous avez créé le pipeline. Vous pouvez ajouter des régions à distribuer à l'aide du bouton Ajouter une région, et vous pouvez supprimer toutes les régions à l'exception de la région 1.

  Les paramètres régionaux incluent :

  - Région cible
  - Le Service est défini par défaut sur « ECR » et n'est pas modifiable.
  - Nom du référentiel : nom de votre référentiel cible (sans compter l'emplacement Amazon ECR). Par exemple, le nom du référentiel avec l'emplacement ressemblerait au modèle suivant :

    ```
    <account-id>.dkr.ecr.<region>.amazonaws.com/<repository-name>
    ```

    > **Note**
    >
    > Si vous modifiez le nom du référentiel, seules les images créées après le changement de nom seront ajoutées sous le nouveau nom. Toutes les images créées précédemment par votre pipeline restent dans leur référentiel d'origine.

Pour modifier vos paramètres de distribution depuis la page des détails du pipeline, procédez comme suit :

1. Choisissez l'onglet Paramètres de distribution.
2. Choisissez Modifier dans le coin supérieur droit du panneau Détails de la distribution.
3. Lorsque vous êtes prêt à enregistrer les mises à jour apportées à vos paramètres de distribution, choisissez Enregistrer les modifications.

Modifiez le calendrier de construction de votre pipeline

La page Modifier le pipeline inclut les informations suivantes que vous pouvez modifier après avoir créé le pipeline :

- Description de votre pipeline.
- Collecte de métadonnées améliorée. Cette option est activée par défaut. Pour la désactiver, désactivez la case à cocher Activer la collecte améliorée de métadonnées.
- Le calendrier de construction de votre pipeline. Vous pouvez modifier vos options de planification et tous les paramètres dans cette section.

Pour modifier votre pipeline depuis la page des détails du pipeline, procédez comme suit :

1. Dans le coin supérieur droit de la page des détails du pipeline, choisissez Actions, puis Modifier le pipeline.
2. Lorsque vous êtes prêt à enregistrer vos mises à jour, choisissez Enregistrer les modifications.

> **Note**
>
> Pour plus d'informations sur la planification de votre build à l'aide d'expressions cron, consultezUtiliser des expressions cron dans Image Builder.

Mettez à jour les pipelines d'images de conteneurs à partir du AWS CLI

Vous pouvez mettre à jour un pipeline d'images de conteneur à l'aide d'un fichier JSON comme entrée de la update-image-pipeline commande du AWS CLI. Pour configurer le fichier JSON, vous devez disposer d'Amazon Resource Names (ARN) pour référencer les ressources existantes suivantes :

- Pipeline d'images à mettre à jour
- Recette de contenant
- Configuration de l'infrastructure
- Paramètres de distribution (s'ils sont inclus dans le pipeline actuel)

> **Note**
>
> Si la ressource des paramètres de distribution est incluse, le référentiel ECR qui est spécifié comme référentiel cible dans les paramètres de distribution de la région dans laquelle la commande s'exécute (Région 1) a priorité sur le référentiel cible spécifié dans la recette du conteneur.

Procédez comme suit pour mettre à jour un pipeline d'images de conteneur à l'aide de la update-image-pipeline commande suivante AWS CLI :

> **Note**
>
> La update-image-pipeline commande remplace l'intégralité de la configuration du pipeline. Vous devez spécifier toutes les propriétés requises dans la demande de mise à jour. Incluez toutes les propriétés, qu'il s'agisse des propriétés que vous souhaitez modifier ou des propriétés qui doivent rester les mêmes. Les propriétés que vous omettez sont rétablies aux valeurs par défaut ou supprimées.
> Pour éviter la suppression involontaire de paramètres existants, utilisez la get-image-pipeline commande pour récupérer la configuration actuelle. Modifiez ensuite uniquement les champs que vous souhaitez modifier :
>
> ```
> aws imagebuilder get-image-pipeline --image-pipeline-arn
>  arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-container-
> pipeline
> ```
>
> Utilisez la sortie comme base pour votre fichier JSON de demande de mise à jour.

1. Créer un fichier JSON d'entrée CLI

   Utilisez votre outil d'édition de fichiers préféré pour créer un fichier JSON avec les clés suivantes, ainsi que des valeurs valides pour votre environnement. Cet exemple utilise un fichier nommé`create-component.json` :

   ```
   {
    "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
   pipeline/my-example-pipeline",
   ```

```
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-example-recipe/2020.12.08",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 120
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "DISABLED"
}
```

> **Note**
>
> - Vous devez inclure l'option `file://` au début du chemin du fichier JSON.
> - Le chemin d'accès du fichier JSON doit suivre la convention appropriée pour le système d'exploitation de base sur lequel vous exécutez la commande. Par exemple, Windows utilise la barre oblique inverse (\) pour faire référence au chemin du répertoire, tandis que Linux et macOS utilisent la barre oblique (/).

2. Exécutez la commande suivante en utilisant le fichier que vous avez créé en entrée.

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-
pipeline.json
```

Configuration des flux de travail du pipeline d'images dans Image Builder

Grâce aux flux de travail liés aux images, vous pouvez personnaliser les flux de travail exécutés par votre pipeline pour créer, tester et distribuer des images en fonction de vos besoins. Les

flux de travail que vous définissez s'exécutent dans le contexte de la structure de flux de travail Image Builder. Pour plus d'informations sur les étapes qui constituent le cadre du flux de travail, consultezGérez les flux de travail de création, de test et de distribution pour les images Image Builder.

Créez un flux de travail

Créez des flux de travail exécutés au cours de la `Build` phase du framework de flux de travail. Vous ne pouvez spécifier qu'un seul flux de travail de génération pour votre pipeline. Ou vous pouvez ignorer complètement la compilation pour configurer un pipeline de test uniquement.

Workflow de test

Les flux de travail de test sont exécutés au cours de la `Test` phase du framework de flux de travail. Vous pouvez ajouter un ou plusieurs flux de travail de test à votre pipeline, dans la limite de dix flux de travail au total, y compris vos flux de travail de création et de distribution. Les flux de travail de test peuvent être exécutés en parallèle ou un par un, selon la façon dont vous définissez vos groupes de tests. Vous pouvez également ignorer complètement les tests si vous souhaitez uniquement créer votre pipeline.

Workflow de distribution

Les flux de distribution s'exécutent au cours de la `Distribution` phase de la structure des flux de travail. Un flux de travail de distribution est facultatif. Si vous n'en spécifiez aucune, Image Builder distribue toujours votre AMI en exécutant la configuration de distribution que vous attachez au pipeline ou à l'image. Spécifiez un flux de distribution lorsque vous souhaitez remplacer cette configuration de distribution par d'autres paramètres de distribution, ou pour obtenir une meilleure visibilité sur le processus de distribution. Vous pouvez spécifier un flux de travail de distribution pour votre pipeline. Si vous créez des images de conteneurs sans spécifier de flux de travail de distribution, Image Builder ajoute le flux de travail de distribution de conteneurs géré par Amazon pour vous.

Définissez des groupes de test pour les flux de travail de test

Vous définissez des flux de travail de test au sein de groupes de tests. Vous pouvez exécuter plusieurs flux de travail de test pour votre pipeline. La limite combinée est de dix flux de travail au total, y compris vos flux de travail de création et de distribution. Vous décidez d'exécuter les flux de travail de test dans un ordre spécifique ou d'en exécuter le plus grand nombre possible en même

temps. Leur fonctionnement dépend de la façon dont vous définissez vos groupes de test. Les scénarios suivants illustrent plusieurs manières de définir vos flux de travail de test.

> **Note**
>
> Les `onFailure` paramètres `parallelGroup` et s'appliquent uniquement aux flux de travail de test. Image Builder les rejette dans un flux de travail de création ou de distribution avec `unInvalidParameterCombinationException`.

Comment fonctionnent les groupes de test

Image Builder exécute vos flux de travail de test dans des groupes de test. Il gère les groupes un par un. Les flux de travail de test au sein d'un même groupe s'exécutent en même temps (en parallèle). Image Builder exécute jusqu'à cinq flux de travail de test en même temps ; si un groupe en compte plus de cinq, Image Builder lance le flux de travail de test suivant à la fin de chacun d'entre eux. Un flux de travail de test qui ne fait pas partie d'un groupe s'exécute seul.

La façon dont vous définissez les groupes de test varie selon que vous utilisez la console ou l'API AWS CLI et. Sélectionnez l'onglet correspondant à la méthode que vous utilisez.

Console de gestion AWS

Lorsque vous créez ou modifiez un pipeline dans la console, vous définissez vos flux de travail de test dans la section Workflow de test du processus de création d'image. Pour exécuter des tests, choisissez l'option permettant de sélectionner les flux de travail de test. Pour créer des images sans effectuer de test, choisissez l'option permettant d'ignorer les flux de travail de test.

La console organise les flux de travail de test en groupes de tests. Un nouveau pipeline commence avec un groupe de test. Pour créer votre configuration de test, procédez comme suit :

- Ajouter un flux de travail de test à un groupe — Ajoutez un flux de travail de test à un groupe et sélectionnez le flux de travail parmi vos propres flux de travail (dont je suis propriétaire) ou les AWS flux de travail gérés (gérés par Amazon). Si le flux de travail définit des paramètres, la console les affiche afin que vous puissiez saisir des valeurs. Un groupe peut contenir jusqu'à cinq flux de travail de test, qui s'exécutent simultanément.
- Ajouter un groupe de test — Ajoutez un autre groupe de tests pour exécuter un autre ensemble de tests. Vous pouvez ajouter jusqu'à dix groupes de tests. Pour renommer un groupe,

choisissez Modifier le nom. Le nom d'un groupe peut comporter jusqu'à 128 caractères et peut inclure des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (). _

Image Builder exécute les groupes de test un par un, dans l'ordre dans lequel ils apparaissent dans l'assistant. Les flux de travail de test au sein d'un groupe s'exécutent simultanément. Un groupe peut contenir jusqu'à cinq flux de travail de test. Pour exécuter vos tests un par un, placez chaque flux de travail de test dans son propre groupe.

> **Note**
>
> Vous pouvez ajouter ou supprimer des flux de travail et des groupes de test, mais vous ne pouvez pas les réorganiser. Planifiez l'ordre de vos groupes de test avant de les créer.

AWS CLI

Lorsque vous utilisez l'API AWS CLI ou Image Builder, vous ne définissez pas directement de groupes de tests. Vous définissez plutôt l'`parallelGroup`attribut pour chaque flux de travail de test de la `workflows` liste. Les flux de travail de test qui ont la même `parallelGroup` valeur forment un groupe et s'exécutent en même temps. Un flux de travail de test qui n'a aucune `parallelGroup` valeur s'exécute de lui-même. Si vous ne définissez de `parallelGroup` valeur pour aucun de vos flux de travail de test, Image Builder les exécute un par un.

L'API AWS CLI et ne limite pas le nombre de flux de travail de test dans un groupe. La seule limite est le nombre total de flux de travail, qui ne peut pas dépasser dix (y compris vos flux de travail de création et de distribution). La console limite un groupe à cinq flux de travail de test.

Image Builder exécute toujours les flux de création en premier, puis les flux de travail de test, puis le flux de distribution. Cet ordre s'applique quelle que soit la façon dont vous les mettez en vente.

Au cours de la phase de test, Image Builder organise vos flux de travail de test comme suit :

- Il classe les groupes en fonction de la position où chaque groupe apparaît pour la première fois dans votre `workflows` liste. Il ne les classe pas par `parallelGroup` valeur.
- Il conserve chaque flux de travail qui n'a aucun groupe dans sa position répertoriée.
- Il exécute les flux de travail en groupe, à l'endroit où ce groupe apparaît pour la première fois.

Exemple

Supposons que vous listiez cinq flux de travail de test dans cet ordre et que vous `parallelGroup` définissiez la valeur de certains d'entre eux :

- Workflow de test A — aucun groupe
- Workflow de test B — `parallelGroup` : `tests-1`
- Workflow de test C — aucun groupe
- Workflow de test D — `parallelGroup` : `tests-1`
- Flux de travail de test E — `parallelGroup` : `tests-2`

Image Builder les exécute dans l'ordre suivant :

1. Testez le flux de travail A, seul.
2. Testez le flux de travail B et le flux de travail de test D en même temps. Ils partagent le `tests-1` groupe, qui apparaît pour la première fois dans le flux de travail de test B.
3. Testez le flux de travail C seul.
4. Testez le flux de travail E, seul, en `tests-2` groupe.

Le flux de travail de test D s'exécute conjointement avec le flux de travail de test B car ils partagent un groupe, même si vous avez répertorié le flux de travail de test C entre eux. Le groupe nommé `tests-1` s'exécute avant `tests-2` parce qu'il `tests-1` apparaît en premier dans la liste. L'ordre des groupes suit l'ordre de la liste, pas la `parallelGroup` valeur.

Scénario 1 : exécuter un flux de travail de test à la fois

Pour exécuter tous vos flux de travail de test un par un, configurez un groupe de test distinct pour chaque flux de test, dans la limite combinée de dix flux de travail (y compris vos flux de travail de création et de distribution). Image Builder exécute les groupes de test un par un. C'est une façon de s'assurer que vos flux de travail de test s'exécutent un par un dans un ordre spécifique.

Scénario 2 : exécuter plusieurs flux de travail de test en même temps

Si l'ordre n'a pas d'importance et que vous souhaitez exécuter autant de flux de travail de test que possible en même temps, vous pouvez configurer un seul groupe de tests et y placer le nombre maximum de flux de travail de test. Image Builder démarre jusqu'à cinq flux de travail de test en même temps et lance des flux de test supplémentaires au fur et à mesure que d'autres sont terminés.

Si votre objectif est d'exécuter vos flux de travail de test le plus rapidement possible, c'est l'une des manières de le faire.

Scénario 3 : mélanger et assortir

Si vous avez un scénario mixte, avec certains flux de travail de test pouvant être exécutés en même temps et d'autres devant être exécutés un par un, vous pouvez configurer vos groupes de tests pour atteindre cet objectif. La seule limite à la façon dont vous configurez vos groupes de test est le nombre maximum de flux de travail de test pouvant être exécutés pour votre pipeline.

Définissez les paramètres du flux de travail dans un pipeline Image Builder depuis la console

Les paramètres de flux de travail fonctionnent de la même manière pour les flux de travail de création, de test et de distribution. Lorsque vous créez ou mettez à jour un pipeline, vous sélectionnez les flux de travail de création, de test et de distribution que vous souhaitez inclure. Si vous avez défini des paramètres dans le document de flux de travail pour un flux de travail que vous avez sélectionné, Image Builder les affiche dans le panneau Paramètres. Le panneau est masqué pour les flux de travail dont aucun paramètre n'est défini.

Chaque paramètre affiche les attributs suivants définis par votre document de flux de travail :

- Nom (non modifiable) : nom du paramètre.
- Type (non modifiable) — Type de données pour la valeur du paramètre.
- Valeur : valeur du paramètre. Vous pouvez modifier la valeur du paramètre pour la définir pour votre pipeline.

Spécifiez le rôle de service IAM qu'Image Builder utilise pour exécuter des actions de flux de travail

Lorsque vous associez des flux de travail d'images personnalisés à votre pipeline, ou lorsque vous définissez une configuration de journalisation, vous devez également spécifier un rôle d'exécution. Il s'agit du rôle IAM qu'Image Builder assume pour exécuter les actions de votre flux de travail. Le rôle doit faire confiance au responsable du service Image Builder. Pour connaître les autorisations requises, consultez les informations d'accès au service de flux de travail qui suivent.

Pour exécuter des flux de travail liés à l'imagerie, Image Builder doit être autorisé à effectuer des actions de flux de travail. Vous accordez cette autorisation avec un rôle d'exécution qu'Image Builder assume en votre nom. Vous attribuez le rôle d'exécution comme suit.

> **Important**
>
> Nous vous recommandons de ne pas transmettre le rôle AWSServiceRoleForImageBuilder lié à un service en tant que rôle d'exécution. Créez plutôt un rôle IAM personnalisé et associez la politique EC2ImageBuilderExecutionPolicy AWS gérée. Cette politique accorde les mêmes autorisations dont Image Builder a besoin pour appeler Services AWS en votre nom. L'utilisation d'un rôle personnalisé vous permet de contrôler totalement les autorisations utilisées par Image Builder. Il permet également de maintenir vos politiques de contrôle des services (SCP) et vos politiques de contrôle des ressources (RCP) en vigueur pour les opérations qu'Image Builder effectue en votre nom.

- Console — Dans l'assistant de pipeline Étape 3 Définissez le processus de création d'image, sélectionnez votre rôle personnalisé dans la liste des rôles IAM du panneau d'accès au service.
- API Image Builder : dans la demande CreateImage d'action, spécifiez votre rôle personnalisé comme valeur du `executionRole` paramètre.

Pour créer un rôle d'exécution personnalisé, consultez la section Création d'un rôle pour déléguer des autorisations à un AWS service dans le Guide de Gestion des identités et des accès AWS l'utilisateur.

Associer des flux de travail à un pipeline

Pour associer des flux de travail personnalisés de création, de test et de distribution d'images à votre pipeline, sélectionnez l'onglet correspondant à la méthode que vous souhaitez utiliser.

Console de gestion AWS

Lorsque vous créez ou mettez à jour un pipeline dans le Console de gestion AWS, l'assistant de création d'un pipeline d'images inclut une étape Flux de travail dans laquelle vous sélectionnez les flux de travail exécutés par votre pipeline. Pour associer des flux de travail d'images personnalisés, procédez comme suit au cours de cette étape :

1. Pour Créer un flux de travail, sélectionnez le flux de travail de génération que vous souhaitez que votre pipeline exécute.
2. Pour les flux de travail de test, ajoutez un ou plusieurs flux de travail de test et organisez-les en groupes de tests afin de contrôler l'ordre dans lequel ils s'exécutent. Pour de plus amples informations, veuillez consulter Définissez des groupes de test pour les flux de travail de test.
3. Pour Workflow de distribution, sélectionnez un workflow de distribution si vous souhaitez modifier la configuration de distribution jointe ou obtenir plus de visibilité sur le processus de distribution. Ce nom est facultatif.
4. Pour Rôle d'exécution, sélectionnez le rôle IAM qu'Image Builder assume pour exécuter les actions de votre flux de travail. Un rôle d'exécution est requis lorsque vous associez des flux de travail d'images personnalisés. Pour de plus amples informations, veuillez consulter Spécifiez le rôle de service IAM qu'Image Builder utilise pour exécuter des actions de flux de travail.
5. Si un flux de travail que vous avez sélectionné définit des paramètres, définissez leurs valeurs dans le panneau Paramètres. Pour de plus amples informations, veuillez consulter Définissez les paramètres du flux de travail dans un pipeline Image Builder depuis la console.

Pour une présentation étape par étape de l'assistant complet de création d'un pipeline d'images, consultezCréation et mise à jour de pipelines d'images AMI.

AWS CLI

Pour associer des flux d'images personnalisés lorsque vous créez un pipeline, utilisez la create-image-pipeline commande. Spécifiez votre rôle d'exécution à l'aide du `--execution-role` paramètre et répertoriez vos flux de travail de génération, de test et de distribution avec `--workflows` ce paramètre. L'exemple suivant crée un pipeline avec un flux de travail de génération, un flux de travail de test et un flux de travail de distribution.

```
aws imagebuilder create-image-pipeline \
    --name example-pipeline \
    --image-recipe-arn arn:aws:imagebuilder:us-west-2:111122223333:image-
recipe/example/1.0.0 \
    --infrastructure-configuration-arn arn:aws:imagebuilder:us-
west-2:111122223333:infrastructure-configuration/example \
    --execution-role
 arn:aws:iam::111122223333:role/ImageBuilderWorkflowExecutionRole \
    --workflows '[
        {"workflowArn":"arn:aws:imagebuilder:us-west-2:111122223333:workflow/
build/example-build/1.0.0"},
```

```
        {"workflowArn":"arn:aws:imagebuilder:us-west-2:111122223333:workflow/
test/example-test/1.0.0","parallelGroup":"group-1","onFailure":"CONTINUE"},
        {"workflowArn":"arn:aws:imagebuilder:us-west-2:111122223333:workflow/
distribution/example-distribute/1.0.0"}
    ]'
```

Utilisation EventBridge avec les pipelines Image Builder

AWS et les services partenaires diffusent les événements sur les bus EventBridge événementiels Amazon en temps quasi réel. Vous pouvez également envoyer vos propres événements personnalisés. Les bus événementiels utilisent des règles pour acheminer chaque événement vers une ou plusieurs cibles.

Image Builder fonctionne EventBridge dans deux directions indépendantes :

- Démarrez la création d'un pipeline à partir d'une règle. Vous pouvez définir un pipeline Image Builder comme cible de règle, afin qu'il EventBridge exécute le pipeline lorsqu'un événement correspond à la règle ou lorsqu'un calendrier se déclenche. Consultez Déclenchez la création d'un pipeline selon un calendrier et Déclencher la création d'un pipeline à partir d'un modèle d'événement.
- Réagissez aux événements envoyés par Image Builder. Image Builder publie des événements sur le bus d'événements par défaut lorsque des modifications importantes se produisent, par exemple lorsqu'une image devient `AVAILABLE` ou qu'un scan détecte des CVE. Vous pouvez associer ces événements à une règle et les acheminer vers des cibles telles qu'une fonction Lambda ou une rubrique Amazon SNS pour démarrer vos propres automatisations. Consultez Réagissez aux événements envoyés par Image Builder.

Pour plus d'informations sur les événements envoyés par Image Builder et les champs de chaque événement, consultezMessages d'événements envoyés par Image Builder.

> **Note**
>
> Les bus événementiels le sont Region-specific. Une règle et les ressources qu'elle invoque se trouvent généralement dans la même région ; pour effectuer un routage entre les régions, ciblez un bus d'événements dans la région de destination. Image Builder publie ses événements sur le bus d'événements par défaut de la région où s'exécute la génération.

Table des matières

- EventBridge termes
- Afficher EventBridge les règles de votre pipeline Image Builder
- Déclenchez la création d'un pipeline selon un calendrier
- Déclencher la création d'un pipeline à partir d'un modèle d'événement
- Réagissez aux événements envoyés par Image Builder

EventBridge termes

Les termes suivants vous aident à comprendre comment EventBridge s'intègre à vos pipelines Image Builder.

Événement

Décrit une modification d'environnement susceptible d'affecter une ou plusieurs ressources de l'application. L'environnement peut être un AWS environnement, un service ou une application partenaire SaaS, ou l'une de vos applications ou services. Vous pouvez également configurer des événements planifiés sur une chronologie.

Bus d'événement

Un pipeline qui reçoit des données d'événements provenant d'applications et de services.

Source

Le service ou l'application qui a envoyé l'événement au bus de l'événement.

Cible

Ressource ou point de terminaison EventBridge invoqué lorsqu'il correspond à une règle, fournissant les données de l'événement à la cible.

Règle

Une règle correspond à des événements entrants et les envoie vers des cibles pour être traités. Une seule règle peut envoyer un événement à plusieurs cibles, qui peuvent ensuite être exécutées en parallèle. Les règles sont basées soit sur un modèle d'événement, soit sur un calendrier.

Modèle

Un modèle d'événement définit la structure de l'événement et les champs auxquels une règle correspond afin de lancer l'action cible.

Planning

Les règles de planification exécutent une action selon un calendrier, comme l'exécution d'un pipeline Image Builder pour actualiser une image tous les trimestres. Il existe deux types d'expressions de planification :

- Expressions Cron : faites correspondre des critères de planification spécifiques à l'aide de la syntaxe cron qui peut définir des critères simples ; par exemple, une exécution hebdomadaire un jour spécifique. Vous pouvez également définir des critères plus complexes, comme une diffusion trimestrielle le cinquième jour du mois, entre 2 h et 4 h.
- Expressions de taux : spécifiez un intervalle régulier lorsque la cible est invoquée, par exemple toutes les 12 heures.

Afficher EventBridge les règles de votre pipeline Image Builder

L'onglet EventBridge Règles de la page détaillée des pipelines d'images d'Image Builder affiche les bus d' EventBridge événements auxquels votre compte a accès, ainsi que les règles du bus d'événements sélectionné qui s'appliquent au pipeline actuel. Cet onglet renvoie également directement à la EventBridge console permettant de créer de nouvelles ressources.

Actions liées à la EventBridge console

- Créer un bus événementiel
- Créer une règle

Pour en savoir plus EventBridge, consultez les rubriques suivantes du guide de EventBridge l'utilisateur Amazon.

- Qu'est-ce qu'Amazon EventBridge
- Bus EventBridge événementiels Amazon
- EventBridge Événements Amazon
- EventBridge Règles d'Amazon

Déclenchez la création d'un pipeline selon un calendrier

Définissez un pipeline Image Builder comme cible d'une règle de EventBridge planification pour actualiser une image à une cadence régulière. L'exemple suivant crée une règle qui lance la création d'un pipeline tous les 90 jours.

> **Note**
>
> Un pipeline peut également comporter son propre calendrier intégré (une expression cron ou rate stockée sur le pipeline). Ce calendrier est géré par Image Builder et est distinct de toute EventBridge règle :
>
> - Utilisez le calendrier du pipeline pour une reconstruction récurrente simple. Il peut également gérer les builds à partir des mises à jour des dépendances et désactiver automatiquement un pipeline défaillant. Consultez `Schedule` la référence de l'API EC2 Image Builder.
> - Utilisez une EventBridge règle pour démarrer un build à partir d'un modèle d'événement ou pour étendre un déclencheur à plusieurs cibles.

Console de gestion AWS

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Pour consulter la liste des pipelines d'images créés sous votre compte, choisissez Pipelines d'images dans le volet de navigation.

   > **Note**
   >
   > La liste des pipelines d'images inclut un indicateur pour le type d'image de sortie créée par le pipeline : AMI ouDocker.

3. Pour afficher les détails ou modifier un pipeline, cliquez sur le lien Nom du pipeline. Cela ouvre la vue détaillée du pipeline.

> **Note**
>
> Vous pouvez également cocher la case à côté du nom du pipeline, puis choisir Afficher les détails.

4. Ouvrez l'onglet EventBridge Règles.
5. Conservez le bus d'événements par défaut qui est présélectionné dans le panneau Bus d'événements.
6. Choisissez Créer une règle. Cela vous amène à la page Créer une règle dans la EventBridge console Amazon.
7. Saisissez un nom et une description pour la règle. Le nom de la règle doit être unique dans le bus d'événements de la région sélectionnée.
8. Dans le panneau Définir le modèle, choisissez l'option Planifier. Cela élargit le panneau, avec le tarif fixe pour chaque option sélectionnée.
9. Entrez `90` dans la première case et sélectionnez Jours dans la liste déroulante.
10. Effectuez les actions suivantes dans le panneau Sélectionner les cibles :
    a. Sélectionnez `EC2 Image Builder` dans la liste déroulante Cible.
    b. Pour appliquer la règle à un pipeline Image Builder, sélectionnez le pipeline cible dans la liste déroulante Image Pipeline.
    c. EventBridge a besoin d'une autorisation pour lancer une compilation pour le pipeline sélectionné. Pour cet exemple, conservez l'option par défaut Créer un nouveau rôle pour cette ressource spécifique.
    d. Sélectionnez Ajouter une cible.
11. Sélectionnez Create (Créer).

> **Note**
>
> La EventBridge console actuelle peut créer des planifications via EventBridge Scheduler, et les libellés et les étapes exacts peuvent changer au fil du temps. Le guide de EventBridge l'utilisateur Amazon lié fait autorité pour l'expérience de console actuelle.

AWS CLI

Ces étapes créent une règle de planification qui lance la construction d'un pipeline tous les 90 jours. EventBridge a besoin d'un rôle IAM qu'il peut assumer pour faire appel `imagebuilder:StartImagePipelineExecution` à votre pipeline.

1. Créez une règle avec une expression de taux :

```
aws events put-rule \
  --name refresh-image-every-90-days \
  --schedule-expression "rate(90 days)"
```

2. Créez un rôle IAM qui EventBridge peut assumer, avec cette politique de confiance :

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": { "Service": "events.amazonaws.com" },
        "Action": "sts:AssumeRole"
    }]
}
```

Joignez une politique d'autorisations qui permet au pipeline de démarrer :

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "imagebuilder:StartImagePipelineExecution",
        "Resource": "arn:aws:imagebuilder:us-west-2:111122223333:image-
pipeline/my-pipeline"
    }]
}
```

3. Ajoutez le pipeline comme cible de la règle, en faisant référence au rôle de l'étape précédente :

```
aws events put-targets \
  --rule refresh-image-every-90-days \
  --targets '[{
    "Id": "1",
```

```
    "Arn": "arn:aws:imagebuilder:us-west-2:111122223333:image-pipeline/my-
pipeline",
    "RoleArn": "arn:aws:iam::111122223333:role/EventBridgeImageBuilderRole"
  }]'
```

4. Vérifiez que la cible est attachée :

```
aws events list-targets-by-rule --rule refresh-image-every-90-days
```

> **Note**
>
> Pour en savoir plus sur les paramètres des règles d'expression de taux qui ne sont pas abordés dans cet exemple, consultez la section Expressions de taux dans le guide de EventBridge l'utilisateur Amazon.

Déclencher la création d'un pipeline à partir d'un modèle d'événement

Au lieu d'un calendrier, vous pouvez démarrer la création d'un pipeline lorsqu'un événement correspond à un modèle. Créez une règle avec un modèle d'événement, puis ajoutez votre pipeline comme cible (avec le même rôle IAM décrit dansDéclenchez la création d'un pipeline selon un calendrier).

L'exemple suivant démarre une génération chaque fois qu'une image de base spécifique devient, par exemple, `AVAILABLE` pour enchaîner une génération d'image dépendante une fois son image de base terminée. Le `resources` préfixe étend la règle à l'image de base afin qu'elle corresponde uniquement à cette image, et non à la propre sortie du pipeline dépendant :

```
{
    "source": ["aws.imagebuilder"],
    "detail-type": ["EC2 Image Builder Image State Change"],
    "detail": {
        "state": { "status": ["AVAILABLE"] }
    },
    "resources": [{
        "prefix": "arn:aws:imagebuilder:us-west-2:111122223333:image/my-base-image/"
    }]
}
```

> **Important**
>
> Étendez le modèle d'événement à l'image de base, comme le montre le `resources` préfixe de cet exemple. Un modèle qui correspond à chaque `AVAILABLE` image du compte correspond également à l'image de sortie du pipeline dépendant, ce qui redémarre le pipeline et peut créer une boucle de construction sans fin. Filtrez en fonction du nom de l'image de base ou du préfixe ARN spécifique afin que la règle corresponde uniquement à l'image souhaitée.

Console de gestion AWS

1. Ouvrez la EventBridge console Amazon à l'adresse https://console.aws.amazon.com/events/.
2. Choisissez Règles, puis Créer une règle. Conservez le bus d'événements par défaut.
3. Entrez un nom et une description. Sous Type de règle, choisissez Règle avec un modèle d'événement.
4. Pour la source de l'événement, choisissez Autre et collez le modèle d'événement ci-dessus dans la zone Modèle d'événement.
5. Dans le panneau Sélectionner les cibles, faites votre choix `EC2 Image Builder` dans la liste déroulante Cible, puis choisissez le pipeline à créer dans la liste déroulante Image Pipeline.
6. EventBridge a besoin d'une autorisation pour démarrer le pipeline. Conservez l'option par défaut pour Créer un nouveau rôle pour cette ressource spécifique.
7. Choisissez Créer une règle.

AWS CLI

Créez la règle et ajoutez le pipeline comme cible :

```
aws events put-rule \
  --name rebuild-on-base-available \
  --event-pattern '{"source":["aws.imagebuilder"],"detail-type":["EC2 Image Builder
 Image State Change"],"detail":{"state":{"status":["AVAILABLE"]}},"resources":
[{"prefix":"arn:aws:imagebuilder:us-west-2:111122223333:image/my-base-image/"}]}'

aws events put-targets \
```

```
  --rule rebuild-on-base-available \
  --targets '[{"Id":"1","Arn":"arn:aws:imagebuilder:us-west-2:111122223333:image-
pipeline/dependent-
pipeline","RoleArn":"arn:aws:iam::111122223333:role/EventBridgeImageBuilderRole"}]'
```

Pour les événements que vous pouvez associer et leurs champs, consultezMessages d'événements envoyés par Image Builder.

Réagissez aux événements envoyés par Image Builder

Image Builder publie des événements sur le bus d'événements par défaut lorsque vos ressources atteignent des points importants, par exemple, lorsqu'une image change d'état, qu'une numérisation est terminée, qu'une étape du flux de travail est interrompue pour la saisie ou qu'un pipeline est automatiquement désactivé. Vous pouvez associer ces événements à une règle et les acheminer vers une cible pour démarrer votre propre automatisation. Les modèles courants incluent :

- Avertissez une équipe via une rubrique Amazon SNS lorsqu'une image devient `AVAILABLE` ou`FAILED`.
- Appelez une fonction Lambda pour enregistrer une nouvelle AMI dans un magasin de paramètres, démarrer des tests de conformité externes ou mettre à jour un pipeline de déploiement.
- Alerte sur les résultats de sécurité lorsqu'un scan signale des CVE critiques.
- Mettez en place un flux de travail d'approbation lorsqu'une étape `WaitForAction` du flux de travail interrompt une création.

Pour agir sur un événement envoyé par Image Builder, créez une règle dont le modèle d'événement correspond à l'événement `source` (`aws.imagebuilder`) et `detail-type` qui restreint éventuellement les champs qu'il contient`detail`. Ajoutez ensuite la cible qui exécute votre automatisation.

Exemple : Notifier lorsque l'image est terminée (**`AVAILABLE`**ou**`FAILED`**)

```
{
    "source": ["aws.imagebuilder"],
    "detail-type": ["EC2 Image Builder Image State Change"],
    "detail": {
        "state": { "status": ["AVAILABLE", "FAILED"] }
    }
}
```

Exemple : alerte uniquement lorsqu'un scan détecte des CVE critiques

Utilisez un comparateur numérique afin que les scans nets ne déclenchent pas la règle :

```
{
    "source": ["aws.imagebuilder"],
    "detail-type": ["EC2 Image Builder CVE Detected"],
    "detail": {
        "finding-severity-counts": {
            "critical": [{ "numeric": [">", 0] }]
        }
    }
}
```

Console de gestion AWS

1. Ouvrez la EventBridge console Amazon à l'adresse https://console.aws.amazon.com/events/.
2. Choisissez Règles, puis Créer une règle. Conservez le bus d'événements par défaut.
3. Entrez un nom et une description. Sous Type de règle, choisissez Règle avec un modèle d'événement.
4. Pour la source de l'événement, choisissez Autre et collez l'un des modèles d'événements ci-dessus dans la zone Modèle d'événement.
5. Choisissez votre cible, par exemple une rubrique Amazon SNS ou une fonction Lambda, puis définissez les paramètres de la cible.
6. Choisissez Créer une règle.

AWS CLI

Créez la règle et envoyez les événements correspondants à une rubrique Amazon SNS :

```
aws events put-rule \
  --name image-available-or-failed \
  --event-pattern '{"source":["aws.imagebuilder"],"detail-type":["EC2 Image Builder
 Image State Change"],"detail":{"state":{"status":["AVAILABLE","FAILED"]}}}'

aws events put-targets \
  --rule image-available-or-failed \
  --targets '[{"Id":"1","Arn":"arn:aws:sns:us-west-2:111122223333:image-status"}]'
```

EventBridge a également besoin d'une autorisation pour envoyer des événements à votre cible. Pour plus d'informations sur les autorisations requises par chaque type de cible, consultez les sections Autorisations pour l'envoi d'événements aux cibles et Utilisation de politiques basées sur les ressources pour Amazon EventBridge dans le Guide de l' EventBridge utilisateur Amazon.

> **Note**
>
> Lorsque vous faites correspondre l'`EC2 Image Builder Workflow Step Waiting`événement, votre cible peut appeler `SendWorkflowStepAction` avec le `workflow-step-execution-id` nom de l'événement et un `action` `RESUME` ou`STOP`. Consultez WaitForAction.

Intégrez des produits et services dans Image Builder

EC2 Image Builder s'intègre à AWS Marketplace d'autres applications pour vous aider à créer des images de machine personnalisées robustes Services AWS et sécurisées.

Produits

Les recettes Image Builder peuvent intégrer des produits d'image AWS Marketplace et des composants gérés par Image Builder afin de fournir des fonctionnalités de création et de test spécialisées, comme suit.

- AWS Marketplace produits illustrés : utilisez une image de produit AWS Marketplace comme image de base de votre recette afin de respecter les normes organisationnelles, telles que CIS Hardening. Lorsque vous créez une recette à partir de la console Image Builder, vous pouvez choisir parmi vos abonnements existants ou rechercher un produit spécifique auprès de AWS Marketplace. Lorsque vous créez une recette à partir de l'API, de l'interface de ligne de commande ou du SDK Image Builder, vous pouvez spécifier un produit d'image Amazon Resource Name (ARN) à utiliser comme image de base.
- Composants Image Builder : les composants que vous spécifiez dans vos recettes peuvent effectuer des actions de génération et de test, par exemple pour installer un logiciel ou effectuer une validation de conformité. Certains produits d'image auxquels vous vous abonnez AWS Marketplace peuvent inclure un composant complémentaire que vous pouvez utiliser dans vos recettes. Les images CIS Hardened incluent un AWSTOE composant correspondant que vous pouvez utiliser dans votre recette pour appliquer les directives CIS Benchmarks de niveau 1 à votre configuration.

> **Note**
>
> Pour plus d'informations sur les produits liés à la conformité, consultez. Produits de conformité pour vos images Image Builder

Services

Image Builder s'intègre aux éléments suivants Services AWS pour fournir des mesures détaillées sur les événements, la journalisation et la surveillance. Ces informations vous permettent de suivre votre

activité, de résoudre les problèmes liés à la création d'images et de créer des automatisations basées sur les notifications d'événements.

- AWS Organizations— vous AWS Organizations permet d'appliquer des politiques de contrôle des services (SCP) aux comptes de votre organisation. Vous pouvez créer, gérer, activer et désactiver des politiques individuelles. Comme tous les autres AWS artefacts et services, Image Builder respecte les politiques définies dans AWS Organizations. AWS fournit des modèles de SCP pour les scénarios courants, tels que l'application de contraintes aux comptes membres afin de lancer des instances avec uniquement des AMI approuvées.
- AWS CloudTrail— Surveillez les événements Image Builder qui sont envoyés à CloudTrail. Pour plus d'informations sur CloudTrail l'intégration à Image Builder, consultezEnregistrez les appels d'API Image Builder à l'aide de CloudTrail.

  Pour en savoir plus CloudTrail, notamment sur la manière de l'activer et de trouver vos fichiers journaux, consultez le Guide de AWS CloudTrail l'utilisateur.
- Amazon CloudWatch Logs : surveillez, stockez et accédez à vos fichiers journaux Image Builder avec CloudWatch. Vous pouvez éventuellement enregistrer vos journaux dans un compartiment S3. Pour en savoir plus sur CloudWatch l'intégration à Image Builder, consultezSurveillez les journaux d'Image Builder avec Amazon CloudWatch Logs.

  Pour plus d'informations sur CloudWatch les journaux, consultez Qu'est-ce qu'Amazon CloudWatch Logs ? dans le guide de l'utilisateur d'Amazon CloudWatch Logs.
- Amazon Elastic Container Registry (Amazon ECR) — Amazon ECR est un service géré de registre d'images de AWS conteneurs sécurisé, évolutif et fiable. Les images de conteneur que vous créez avec Image Builder sont stockées dans Amazon ECR dans votre région source (où s'exécute votre build) et dans toutes les régions où vous distribuez l'image de conteneur. Pour plus d'informations sur Amazon ECR, consultez le guide de l'utilisateur d'Amazon Elastic Container Registry.
- Amazon EventBridge — Connectez-vous à un flux de données d'événements en temps réel provenant des activités d'Image Builder sur votre compte. Pour plus d'informations sur EventBridge, consultez Qu'est-ce qu'Amazon EventBridge ? dans le guide de EventBridge l'utilisateur Amazon.
- Amazon Inspector : découvrez les vulnérabilités de votre logiciel et de vos paramètres réseau grâce à des analyses automatiques de l'instance de test EC2 lancée par Image Builder pour créer une nouvelle image. Image Builder enregistre les résultats pour votre ressource d'image de sortie afin que vous puissiez les étudier et y remédier une fois votre instance de test terminée. Pour plus

d'informations sur les scans et la tarification, consultez Qu'est-ce qu'Amazon Inspector ? dans le guide de l'utilisateur d'Amazon Inspector.

Amazon Inspector peut également analyser vos référentiels ECR si vous configurez une analyse améliorée. Pour plus d'informations, consultez la section Numérisation d'images de conteneurs Amazon ECR dans le guide de l'utilisateur Amazon Inspector.

> **Note**
>
> Amazon Inspector est une fonctionnalité payante.

- AWS License Manager— Vous pouvez associer une licence autogérée License Manager à une AMI de sortie pendant le processus de distribution. La licence que vous spécifiez pour la région de destination doit déjà exister dans cette région. Pour plus d'informations sur les licences autogérées, consultez la section Self-managed Licences dans License Manager.
- AWS Marketplace— Consultez la liste de vos abonnements actuels AWS Marketplace et recherchez des produits illustrés directement depuis Image Builder. Vous pouvez également utiliser un produit d'image auquel vous êtes abonné comme image de base pour une recette Image Builder. Pour plus d'informations sur la gestion des AWS Marketplace abonnements, consultez la section Acheter des produits dans le Guide de AWS Marketplace l'acheteur.
- AWS Resource Access Manager (AWS RAM) — Avec AWS RAM, vous pouvez partager des ressources avec n'importe qui Compte AWS ou via AWS Organizations. Si vous en avez plusieurs Comptes AWS, vous pouvez créer des ressources de manière centralisée et AWS RAM les utiliser pour partager ces ressources avec d'autres comptes. EC2 Image Builder permet de partager les ressources suivantes : composants, images et recettes d'images. Pour plus d'informations à ce sujet AWS RAM, consultez le Guide de AWS Resource Access Manager l'utilisateur. Pour plus d'informations sur le partage des ressources d'Image Builder, consultezPartagez les ressources d'Image Builder avec AWS RAM.
- Amazon Simple Notification Service (Amazon SNS) : si configuré, publiez des messages détaillés concernant l'état de votre image dans une rubrique SNS à laquelle vous êtes abonné. Pour plus d'informations sur Amazon SNS, consultez Qu'est-ce qu'Amazon SNS ? dans le Guide du développeur Amazon Simple Notification Service.

Sujets relatifs à l'intégration des produits et des services

- EventBridge Intégration d'Amazon dans Image Builder
- Intégration d'Amazon Inspector dans Image Builder

- AWS Marketplace intégration dans Image Builder
- Intégration d'Amazon SNS dans Image Builder
- Produits de conformité pour vos images Image Builder

EventBridge Intégration d'Amazon dans Image Builder

Amazon EventBridge est un service de bus d'événements sans serveur que vous pouvez utiliser pour connecter votre application Image Builder à des données connexes provenant d'autres Services AWS applications. Dans EventBridge, une règle fait correspondre les événements entrants et les envoie à des cibles pour traitement. Une seule règle peut envoyer un événement à plusieurs cibles, et ces événements s'exécutent ensuite en parallèle.

Grâce à EventBridge, vous pouvez automatiser vos événements système Services AWS et y répondre automatiquement, tels que les problèmes de disponibilité des applications ou les modifications de ressources. Services AWS diffusez des événements EventBridge en temps quasi réel. Vous pouvez définir des règles qui réagissent aux événements entrants pour lancer des actions. Par exemple, envoyer un événement à une fonction Lambda lorsque l'état d'une instance EC2 passe de « en attente » à « en cours d'exécution ». EventBridge appelle ces définitions correspondantes des modèles d'événements. Pour créer une règle basée sur un modèle d'événement, consultez la section Création de EventBridge règles Amazon qui réagissent aux événements dans le guide de EventBridge l'utilisateur Amazon.

Image Builder fonctionne EventBridge dans deux directions indépendantes. Choisissez celui qui correspond à votre objectif :

- Exécutez un pipeline en réponse à un événement (Image Builder en tant que cible). EventBridge lance la création d'un pipeline lorsqu'un événement correspond à une règle ou lorsqu'un calendrier se déclenche.
- Réagissez à ce que rapporte Image Builder (Image Builder en tant que source). Image Builder publie les événements dans le bus d'événements par défaut que vous pouvez faire correspondre et acheminer vers vos propres cibles.

Pour les procédures étape par étape dans les deux sens, voirUtilisation EventBridge avec les pipelines Image Builder. Pour le catalogue des événements, voirMessages d'événements envoyés par Image Builder.

Les actions qui peuvent être lancées automatiquement sont les suivantes :

- Invoquer une fonction AWS Lambda
- Appeler Run Command (Exécuter la commande) d'Amazon EC2
- Relayer l'événement à Amazon Kinesis Data Streams
- Activer une machine à AWS Step Functions états
- Notifier une rubrique Amazon SNS ou une file d'attente Amazon SQS

Vous pouvez également configurer des règles de planification pour le bus d'événements par défaut afin d'effectuer une action à intervalles réguliers, telle que l'exécution d'un pipeline Image Builder pour actualiser une image tous les trimestres. Il existe deux types d'expressions de planification :

- expressions cron — L'exemple suivant d'expression cron planifie une tâche à exécuter tous les jours à midi UTC+0 :

  ```
  cron(0 12 * * ? *)
  ```

  Pour plus d'informations sur l'utilisation des expressions cron avec EventBridge, consultez la section Expressions Cron dans le guide de l' EventBridge utilisateur Amazon.
- expressions de taux : l'exemple suivant d'expression de taux planifie une tâche à exécuter toutes les 12 heures :

  ```
  rate(12 hour)
  ```

  Pour plus d'informations sur l'utilisation des expressions tarifaires avec EventBridge, consultez la section Expressions tarifaires dans le Guide de EventBridge l'utilisateur Amazon.

Pour plus d'informations sur la manière dont EventBridge les règles s'intègrent aux pipelines d'images Image Builder, consultezUtilisation EventBridge avec les pipelines Image Builder.

> **Note**
>
> Le calendrier intégré d'un pipeline et une règle EventBridge de planification sont des mécanismes différents :
>
> - Un calendrier de pipeline (`schedule.scheduleExpression`) fait partie de la ressource du pipeline. Image Builder l'évalue, peut générer une version sur les mises à jour des dépendances avec `pipelineExecutionStartCondition` et peut désactiver automatiquement un pipeline défaillant avec. `autoDisablePolicy`

- Une règle EventBridge de planification est intégrée EventBridge à votre compte et démarre toujours une génération lorsque la planification se déclenche ; elle n'a aucun contrôle de dépendance et aucun comportement de désactivation automatique.

Messages d'événements envoyés par Image Builder

Image Builder envoie des événements au bus d' EventBridge événements par défaut lorsqu'une ressource Image Builder atteint un point significatif de son cycle de vie, par exemple lorsqu'une image change d'état, qu'une numérisation est terminée, qu'une étape du flux de travail est interrompue pour la saisie ou qu'un pipeline est automatiquement désactivé. Chaque événement utilise la source`aws.imagebuilder`.

Les sections suivantes décrivent chaque événement : quand Image Builder l'envoie, les champs de l'événement `detail` et les ARN des ressources du `resources` tableau. Pour réagir à ces événements, voirRéagissez aux événements envoyés par Image Builder. Les identifiants de compte, les régions, les ARN et les horodatages des exemples sont des espaces réservés. Remplacez-les par les vôtres.

Rubriques

- Modification de l'état de l'image EC2 Image Builder
- EC2 Image Builder CVE détecté
- Étape en attente du flux de travail EC2 Image Builder
- Le pipeline d'images EC2 Image Builder est automatiquement désactivé

Modification de l'état de l'image EC2 Image Builder

Image Builder envoie cet événement lorsque l'état d'une ressource d'image change pendant la création de l'image. Par exemple, lorsque l'état de l'image passe d'un état à un autre, procédez comme suit :

- De `building` à `testing`
- De `testing` à `distribution`
- De `testing` à `failed`
- De `integrating` à `available`

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder Image State Change",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2024-01-18T17:50:56Z",
    "region": "us-west-2",
    "resources": ["arn:aws:imagebuilder:us-west-2:111122223333:image/
cmkencryptedworkflowtest-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222/1.0.0/1"],
    "detail": {
        "previous-state": {
            "status": "TESTING"
        },
        "state": {
            "status": "AVAILABLE"
        }
    }
}
```

Image Builder envoie cet événement chaque fois qu'une ressource d'image change d'état pendant la création de l'image, à travers les transitions de génération illustrées ci-dessus.

L'événement `detail` contient les champs suivants :

Champ	Description
`previous-state.status`	État de l'image avant la modification (par exemple,`TESTING`).
`state.status`	L'état de la nouvelle image (par exemple,`AVAILABLE`).

Les valeurs d'état de construction-transition sont `BUILDINGTESTING`,`DISTRIBUTING`,, `INTEGRATING` `AVAILABLECANCELLED`, et. `FAILED`

Le `resources` tableau contient l'ARN de la ressource d'image qui a changé d'état.

EC2 Image Builder CVE détecté

Si la détection CVE est activée pour votre image, Image Builder envoie un message contenant les résultats chaque fois qu'une numérisation d'image est terminée.

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder CVE Detected",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2023-03-01T16:59:09Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:imagebuilder:us-east-1:111122223333:image/test-image/1.0.0/1",
        "arn:aws:imagebuilder:us-east-1:111122223333:image-pipeline/test-pipeline"
    ],
    "detail": {
        "resource-id": "i-1234567890abcdef0",
        "finding-severity-counts": {
            "all": 0,
            "critical": 0,
            "high": 0,
            "medium": 0
        }
    }
}
```

Image Builder envoie cet événement lorsqu'une numérisation d'image est terminée, et uniquement lorsque vous activez la numérisation d'image (Amazon Inspector) pour l'image. Si la numérisation est désactivée, Image Builder n'envoie pas cet événement. Pour activer la numérisation, consultezConfigurez les scans de sécurité pour les images d'Image Builder dans Console de gestion AWS.

L'événement `detail` contient les champs suivants :

Champ	Description
`resource-id`	ID de l'instance de test Amazon EC2 scannée par Image Builder.
`finding-severity-counts.all`	Nombre total de résultats selon les degrés de gravité signalés.
`finding-severity-counts.critical`	Nombre de résultats relatifs à la gravité critique.

Champ	Description
`finding-severity-counts.high`	Nombre de résultats de gravité élevée.
`finding-severity-counts.medium`	Nombre de résultats de gravité moyenne.

> **Important**
>
> `finding-severity-counts`rapports`critical`,`high`, et `medium` compte plus le `all` total. Il ne fait pas `low` de `informational` rapport ni de `untriaged` résultats. Pour consulter l'ensemble des résultats, consultez les résultats de sécurité dans la console Image Builder ou dans Amazon Inspector.

Le `resources` tableau contient l'ARN de la ressource d'image et l'ARN du pipeline d'images lorsque le scan a été exécuté à partir d'un pipeline.

Étape en attente du flux de travail EC2 Image Builder

Image Builder envoie un message lorsqu'une étape du `WaitForAction` flux de travail s'arrête pour attendre la fin d'une action asynchrone.


```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder Workflow Step Waiting",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2024-01-18T16:54:44Z",
    "region": "us-west-2",
    "resources": ["arn:aws:imagebuilder:us-west-2:111122223333:image/
workflowstepwaitforactionwithvalidsnstopictest-a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222/1.0.0/1", "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
build-workflow-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333/1.0.0/1"],
    "detail": {
        "workflow-execution-id": "wf-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "workflow-step-execution-id": "step-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
        "workflow-step-name": "TestAutoSNSStop"
    }
}
```

Image Builder envoie cet événement lorsqu'une étape `WaitForAction` du flux de travail s'arrête pour attendre votre réponse. Le flux de travail reste suspendu jusqu'à ce que vous répondiez ou que l'étape expire.

L'événement `detail` contient les champs suivants :

Champ	Description
`workflow-execution-id`	L'ID du flux de travail en cours qui contient l'étape suspendue.
`workflow-step-execution-id`	L'ID de l'exécution de l'étape spécifique en attente.
`workflow-step-name`	Le nom de l'`WaitForAction` étape, tel que défini dans le document de flux de travail.

Le `resources` tableau contient l'ARN de la ressource d'image et l'ARN du flux de travail.

Pour répondre, appelez `SendWorkflowStepAction` avec le « from » `workflow-step-execution-id` de l'événement et un « `action` of » `RESUME` pour continuer, ou `STOP` pour mettre fin à la construction. Consultez WaitForAction. Pour le piloter automatiquement à partir d'un événement, consultezRéagissez aux événements envoyés par Image Builder.

Le pipeline d'images EC2 Image Builder est automatiquement désactivé

Si vous l'avez configuré `autoDisablePolicy` pour votre pipeline, Image Builder désactive le pipeline et envoie un message d'événement EventBridge lorsque le nombre d'échecs d'exécution planifiés consécutifs du pipeline dépasse le nombre maximum autorisé par la politique.

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder Image Pipeline Automatically Disabled",
```

```
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2025-09-18T16:54:44Z",
    "region": "us-west-2",
    "resources": ["arn:aws:imagebuilder:us-west-2:111122223333:image-pipeline/disabled-image-pipeline-name"],
    "detail": {
        "consecutive-failures": "5"
    }
}
```

Si vous configurez un `autoDisablePolicy` selon le calendrier d'un pipeline, Image Builder comptabilise les échecs consécutifs des builds planifiés. Lorsque le nombre atteint les limites de la politique`failureCount`, Image Builder désactive le pipeline et envoie cet événement.

L'événement `detail` contient les champs suivants :

Champ	Description
`consecutive-failures`	Le nombre d'échecs consécutifs de génération planifiée qui ont déclenché la désactivation automatique.

Le `resources` tableau contient l'ARN du pipeline qui a été désactivé.

> **Note**
>
> Seules les constructions planifiées sont prises en compte dans le total. Une compilation manuelle qui échoue ne l'augmente pas ; une compilation manuelle ou planifiée qui réussit, ainsi que la création ou la mise à jour du pipeline, la réinitialise à zéro. Consultez `autoDisablePolicy` la référence de l'API EC2 Image Builder.

Intégration d'Amazon Inspector dans Image Builder

Lorsque vous activez l'analyse de sécurité avec Amazon Inspector, celle-ci analyse en permanence les images des machines et les instances en cours d'exécution de votre compte pour détecter les vulnérabilités du système d'exploitation et du langage de programmation. Si cette option est activée, l'analyse de sécurité est automatique et Image Builder peut enregistrer un instantané des résultats

de votre instance de test lorsque vous créez une nouvelle image. Amazon Inspector est un service payant.

Lorsqu'Amazon Inspector découvre des vulnérabilités dans vos paramètres réseau ou logiciels, il prend les mesures suivantes :

- Vous informe qu'une découverte a été faite.
- Évalue la gravité de la découverte. L'indice de gravité classe les vulnérabilités pour vous aider à hiérarchiser vos résultats et inclut les valeurs suivantes :
  - Non trié
  - Informationnel
  - Faible
  - Medium
  - Élevée
  - Critique
- Fournit des informations sur la découverte et des liens vers des ressources supplémentaires pour plus de détails.
- Propose des conseils de remédiation pour vous aider à résoudre les problèmes à l'origine de cette découverte.

Configuration des scans de sécurité

Si vous avez activé Amazon Inspector pour votre compte, Amazon Inspector analyse automatiquement les instances EC2 lancées par Image Builder pour créer et tester une nouvelle image. Ces instances ont une courte durée de vie pendant le processus de création et de test, et leurs résultats expirent normalement dès leur arrêt. Pour vous aider à étudier et à corriger les résultats de votre nouvelle image, Image Builder peut éventuellement enregistrer sous forme de capture instantanée tous les résultats identifiés par Amazon Inspector sur votre instance de test pendant le processus de création.

Pour configurer les analyses de sécurité pour votre pipeline, consultezConfigurez les scans de sécurité pour les images d'Image Builder dans Console de gestion AWS.

Examiner les résultats de sécurité

Dans la console Image Builder, vous pouvez consulter les résultats de sécurité de toutes vos ressources Image Builder en un seul endroit. Vous pouvez consulter tous les résultats sur la page

Résultats de sécurité de la section Aperçu de la sécurité, ou vous pouvez regrouper vos résultats par vulnérabilité, par pipeline d'images ou par image. La console affiche par défaut tous les résultats de sécurité. Le panneau récapitulatif de l'option Tous les résultats de sécurité indique le nombre de résultats dont vous disposez pour chaque niveau de gravité. Pour de plus amples informations, veuillez consulter Gérez les résultats de sécurité pour les images d'Image Builder dans Console de gestion AWS.

Pour en savoir plus sur les résultats relatifs à la vulnérabilité d'Amazon Inspector, consultez la section Comprendre les résultats d'Amazon Inspector dans le guide de l'utilisateur Amazon Inspector.

AWS Marketplace intégration dans Image Builder

AWS Marketplace est un catalogue numérique organisé dans lequel vous pouvez trouver et vous abonner à des logiciels, des données et des services tiers qui vous aident à créer des solutions adaptées aux besoins de votre entreprise. AWS Marketplace réunit les acheteurs authentifiés et les vendeurs enregistrés ainsi que des listes de logiciels appartenant à des catégories populaires telles que la sécurité, la mise en réseau, le stockage, l'apprentissage automatique, etc.

Un AWS Marketplace vendeur peut être un éditeur de logiciels indépendant (ISV), un revendeur ou une personne qui propose quelque chose qui fonctionne avec des AWS produits et services. Lorsque le vendeur soumet un produit AWS Marketplace, il définit le prix du produit et les conditions générales d'utilisation. Les acheteurs acceptent les prix, les termes et conditions définis pour l'offre. Pour en savoir plus AWS Marketplace, consultez Qu'est-ce que c'est AWS Marketplace ?

AWS Marketplace fonctionnalités d'intégration

Image Builder s'intègre AWS Marketplace pour fournir les fonctionnalités suivantes directement depuis la console Image Builder :

- Recherchez des produits d'image disponibles dans AWS Marketplace.
- Recherchez des produits AWS Marketplace d'image qui fournissent des composants.
- Consultez la liste de vos abonnements aux AWS Marketplace produits en cours.
- Utilisez un produit d' AWS Marketplace image auquel vous êtes abonné comme image de base pour une recette Image Builder.
- Utilisez AWS Marketplace les composants auxquels vous êtes abonné dans une recette Image Builder.

Image Builder s'intègre AWS Marketplace pour afficher les produits et composants d'image auxquels vous êtes abonné. Vous pouvez également rechercher des produits et des composants d' AWS Marketplace image sur la page Découvrir les produits sans quitter la console Image Builder.

L'AMI de sortie créée par Image Builder inclut les codes de produit des produits et composants AWS Marketplace d'image. Vous pouvez avoir jusqu'à neuf codes de produit pour votre image personnalisée finale.

AWS Marketplace abonnements dans Image Builder

La page Abonnements de la AWS Marketplace section de la console Image Builder affiche la liste des AWS Marketplace produits auxquels vous êtes actuellement abonné. Chaque produit souscrit affiche les informations suivantes :

- Nom du produit. Ceci est lié à la page détaillée du produit dans AWS Marketplace. La page détaillée du produit que vous avez souscrit s'ouvre dans un nouvel onglet de votre navigateur.
- L'éditeur. Ceci est lié à la page détaillée de l'éditeur dans AWS Marketplace. La page détaillée de l'éditeur s'ouvre dans un nouvel onglet de votre navigateur.
- Version à laquelle vous êtes abonné.

> **Note**
>
> Image Builder extrait la version d'une AWS Marketplace image à partir de la version recommandée pour la liste des produits lorsque cette version est disponible. Si la liste ne fournit pas de version analysable, Image Builder dérive une version basée sur la date à partir de la date de création de l'AMI. Par exemple, la version dérivée peut ressembler à`2026.8.11/1`. Par conséquent, la version d'Image Builder du produit peut refléter une date au lieu de la chaîne de version du fournisseur.

- Si des composants associés sont inclus dans le produit auquel vous êtes abonné, Image Builder affiche un lien vers le détail du composant.

En haut de la page, vous pouvez rechercher un produit spécifique par son nom, ou vous pouvez parcourir vos résultats à l'aide des commandes de pagination. Pour utiliser un produit illustré auquel vous êtes abonné dans une nouvelle recette, sélectionnez un produit auquel vous êtes abonné et choisissez Créer une nouvelle recette. Image Builder présélectionne le premier produit de votre liste par défaut.

> **Note**
>
> Si vous recherchez un produit auquel vous venez de vous abonner et que vous ne le voyez pas dans la liste, utilisez le bouton d'actualisation en haut de l'onglet pour actualiser vos résultats. L'affichage d'un nouvel abonnement dans la liste peut prendre quelques minutes.

Découvrez AWS Marketplace produits d'image depuis la console Image Builder

Cette section se concentre sur les produits d' AWS Marketplace image à utiliser comme image de base dans votre recette. Pour les produits qui incluent des composants logiciels associés, vous pouvez filtrer en fonction du propriétaire du produit dans la console et dans l'API, le SDK et l'interface de ligne de commande. Pour de plus amples informations, veuillez consulter Liste des composants d'Image Builder. Pour plus d'informations sur la recherche, l'abonnement et l'utilisation de AWS Marketplace composants, consultezUtilisation AWS Marketplace composants pour personnaliser votre image.

Découvrez les produits

Pour rechercher un produit d' AWS Marketplace image depuis la console Image Builder, procédez comme suit :

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Dans le volet de navigation, choisissez Découvrir les produits dans la AWS Marketplace section.
3. Vous pouvez rechercher des produits illustrés dans l'onglet Produits illustrés de la page Découvrir les produits.

   Image Builder préfiltre les produits AWS Marketplace pour se concentrer sur les images de machines que vous pouvez utiliser dans vos recettes Image Builder. Pour plus d'informations sur AWS Marketplace l'intégration à Image Builder, choisissez l'onglet correspondant à ce que vous souhaitez voir.

   Cet onglet contient deux panneaux. Sur la gauche, le panneau Affiner les résultats vous permet de filtrer vos résultats pour trouver les produits auxquels vous souhaitez vous abonner. Sur la

droite, le panneau Rechercher des produits affiche les produits qui répondent à vos critères de filtrage et vous permet également d'effectuer une recherche par nom de produit.

Affiner les résultats

La liste suivante ne répertorie que quelques-uns des filtres que vous pouvez appliquer à votre recherche de produits :

- Sélectionnez une ou plusieurs catégories de produits, telles que les logiciels d'infrastructure ou l'apprentissage automatique.
- Choisissez les systèmes d'exploitation pour votre produit d'image ou choisissez tous les produits pour une plate-forme de système d'exploitation spécifique, par exemple Tous Linux/Unix.
- Choisissez un ou plusieurs éditeurs pour afficher leurs produits disponibles. Cliquez sur le lien Afficher tout pour afficher tous les éditeurs dont les produits correspondent aux filtres que vous avez appliqués.

> **Note**
>
> Les noms des éditeurs ne sont pas classés par ordre alphabétique. Si vous recherchez un éditeur en particulier`Center for Internet Security`, vous pouvez saisir une partie de son nom dans le champ de recherche en haut de la boîte de dialogue Tous les éditeurs. Vous devez épeler le nom sous forme d'abréviation, par exemple, `CIS` il risque de ne pas produire les résultats que vous recherchez.
> Vous pouvez également parcourir les noms des éditeurs page par page.

Les choix de filtres sont dynamiques. Chaque choix que vous faites affecte vos options pour toutes les autres catégories. Il existe des milliers de produits disponibles dans AWS Marketplace, donc plus vous pouvez filtrer, plus vous avez de chances de trouver ce que vous recherchez.

Rechercher des produits

Pour trouver un produit spécifique par son nom, vous pouvez saisir une partie du nom dans la barre de recherche en haut de ce panneau. Chaque résultat de produit inclut les informations suivantes :

- Le nom et le logo du produit. Ces deux éléments sont liés à la page détaillée du produit dans AWS Marketplace. La page détaillée s'ouvre dans un nouvel onglet de votre navigateur. À partir de là, vous pouvez vous abonner au produit d'image si vous souhaitez l'utiliser dans une recette Image Builder. Pour plus d'informations, consultez la section Acheter des produits dans le Guide de AWS Marketplace l'acheteur.

  Si vous vous abonnez au produit d'image dans AWS Marketplace, revenez à l'onglet Image Builder de votre navigateur et actualisez la liste des produits d'image auxquels vous êtes abonné pour le voir.

  > **Note**
  >
  > La disponibilité de votre nouvel abonnement peut prendre quelques minutes.

- Le nom de l'éditeur. Ceci est lié à la page détaillée de l'éditeur dans AWS Marketplace. La page détaillée de l'éditeur s'ouvre dans un nouvel onglet de votre navigateur.
- La version du produit.
- Le classement par étoiles du produit et les liens directs vers la section d'évaluation de la page détaillée du produit dans AWS Marketplace. La page détaillée s'ouvre dans un nouvel onglet de votre navigateur.
- Les premières lignes de la description du produit.

Juste en dessous de la barre de recherche, vous pouvez voir combien de résultats votre recherche a produits et quel sous-ensemble de ces résultats est actuellement affiché. Vous pouvez utiliser des commandes supplémentaires sur le côté droit du panneau pour ajuster vos paramètres concernant le nombre de produits à afficher simultanément et l'ordre de tri à appliquer à vos résultats. Vous pouvez également utiliser le contrôle de pagination pour parcourir vos résultats.

Utilisez un AWS Marketplace produit d'image dans les recettes Image Builder

Ouvrez la page Créer une recette et sélectionnez un produit d' AWS Marketplace image à utiliser comme image de base, comme suit.

1. Ouvrez la console EC2 Image Builder à https://console.aws.amazon.com/imagebuilder/ l'adresse.
2. Dans le volet de navigation, choisissez Recettes d'images dans la AWS Marketplace section. Cela vous montre la liste des recettes d'images que vous avez créées.
3. Choisissez Créer une recette d'image. Cela ouvre la page Créer une recette.
4. Entrez le nom et la version de votre recette dans la section Détails de la recette comme d'habitude.
5. Dans la section Image de base, choisissez l'option AWS Marketplace images. Cela vous montre la liste des produits AWS Marketplace d'image auxquels vous êtes abonné dans l'onglet Abonnements. Vous pouvez choisir votre image de base dans la liste.

   Vous pouvez également rechercher d'autres produits d'image disponibles AWS Marketplace directement à partir de l'AWS Marketplaceonglet. Choisissez Ajouter des produits ou ouvrez directement l'AWS Marketplaceonglet. Pour plus d'informations sur la définition des filtres et la recherche dans le AWS Marketplace, consultezDécouvrez AWS Marketplace produits d'image depuis la console Image Builder.
6. Entrez les informations restantes comme d'habitude. Si l'un de vos abonnements à des produits inclut des composants de build, vous pouvez les sélectionner dans la liste des composants de build. Sélectionnez `AWS Marketplace` dans la liste des types de propriétaires de composants `Third party managed` pour les voir, ou sélectionnez le composant CIS.

   > **Note**
   >
   > Si votre image de base est un produit d' AWS Marketplace image CIS publié, Image Builder vérifie la compatibilité des composants de CIS renforcement. Choisissez le CIS composant qui correspond au benchmark et au système d'exploitation de votre image de base pour éviter une erreur de validation de recette.
7. Choisissez Créer une recette.

   Votre image finale peut contenir jusqu'à neuf codes de produits issus de produits et de composants AWS Marketplace d'image. Si l'image de base et les composants que vous avez sélectionnés contiennent plus de neuf codes de produit, Image Builder renvoie un message d'erreur lorsque vous essayez de créer la recette.

Intégration d'Amazon SNS dans Image Builder

Amazon Simple Notification Service (Amazon SNS) est un service géré qui fournit des messages asynchrones entre les éditeurs et les abonnés (également appelés producteurs et consommateurs).

Vous pouvez spécifier une rubrique SNS dans la configuration de votre infrastructure. Lorsque vous créez une image ou exécutez un pipeline, Image Builder peut publier des messages détaillés concernant l'état de votre image dans cette rubrique. Lorsque l'état de l'image atteint l'un des états suivants, Image Builder publie un message :

- `AVAILABLE`
- `FAILED`

Pour un exemple de message SNS d'Image Builder, consultezFormat de message SNS. Si vous souhaitez créer une nouvelle rubrique SNS, consultez la section Premiers pas avec Amazon SNS dans le guide du développeur Amazon Simple Notification Service.

Rubriques SNS cryptées

Si votre rubrique SNS est cryptée, vous devez autoriser le rôle de service Image Builder à effectuer les actions suivantes dans la AWS KMS key politique :

- `kms:Decrypt`
- `kms:GenerateDataKey`

> **Note**
>
> Si votre rubrique SNS est chiffrée, la clé qui chiffre cette rubrique doit résider dans le compte sur lequel s'exécute le service Image Builder. Image Builder ne peut pas envoyer de notifications à des sujets SNS chiffrés à l'aide de clés provenant d'autres comptes.

Exemple d'ajout d'une politique de clé KMS

L'exemple suivant montre la section supplémentaire que vous ajoutez à la politique de clé KMS. Utilisez l'Amazon Resource Name (ARN) pour le rôle lié au service IAM qu'Image Builder a créé sous votre compte lorsque vous avez créé une image Image Builder pour la première fois. Pour en savoir

plus sur le rôle lié au service Image Builder, consultez. Utiliser des rôles liés à un service IAM pour Image Builder

```
{
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
    },
    "Action": [
      "kms:GenerateDataKey*",
      "kms:Decrypt"
    ],
    "Resource": "*"
  }]
}
```

Vous pouvez utiliser l'une des méthodes suivantes pour obtenir l'ARN.

Console de gestion AWS

Pour obtenir l'ARN du rôle lié à un service qu'Image Builder a créé sous votre compte depuis le Console de gestion AWS, procédez comme suit :

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de gauche, choisissez Rôles.
3. Recherchez `ImageBuilder` et choisissez le nom de rôle suivant dans les résultats :`AWSServiceRoleForImageBuilder`. La page détaillée du rôle s'affiche.
4. Pour copier l'ARN dans votre presse-papiers, choisissez l'icône à côté du nom de l'ARN.

AWS CLI

Pour obtenir l'ARN du rôle lié à un service qu'Image Builder a créé sous votre compte à partir de AWS CLI, utilisez la commande IAM get-role comme suit.

```
aws iam get-role --role-name AWSServiceRoleForImageBuilder
```

Sortie d'échantillon partielle :

```
{
    "Role": {
        "Path": "/aws-service-role/imagebuilder.amazonaws.com/",
        "RoleName": "AWSServiceRoleForImageBuilder",
        ...
        "Arn": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
        ...
}
```

Format de message SNS

Une fois qu'Image Builder a publié un message dans votre rubrique Amazon SNS, les autres services abonnés à cette rubrique peuvent filtrer le message en fonction du format du message et déterminer s'il répond aux critères pour une action ultérieure. Par exemple, un message de réussite peut lancer une tâche visant à mettre à jour un magasin de AWS Systems Manager paramètres ou à lancer un flux de travail de test de conformité externe pour l'AMI en sortie.

L'exemple suivant montre la charge utile JSON d'un message typique qu'Image Builder publie lorsqu'une compilation de pipeline est terminée et crée une image Linux.

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image",
  "semver": 123794003928538027489912 4227,
  "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3",
  "name": "example-linux-image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 3,
  "state": {
    "status": "AVAILABLE"
  },
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-linux-
image/1.0.0",
    "name": "amjule-barebones-linux",
    "version": "1.0.0",
    "components": [
```

```
      {
        "componentArn": "arn:aws:imagebuilder:us-west-1:123456789012:component/update-
linux/1.0.2/1"
      }
    ],
    "platform": "Linux",
    "parentImage": "arn:aws:imagebuilder:us-west-1:987654321098:image/amazon-linux-2-
x86/2022.6.14/1",
    "blockDeviceMappings": [
      {
        "deviceName": "/dev/xvda",
        "ebs": {
          "encrypted": false,
          "deleteOnTermination": true,
          "volumeSize": 8,
          "volumeType": "gp2"
        }
      }
    ],
    "dateCreated": "Feb 24, 2021 12:31:54 AM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-
linux-image/1.0.0"
    },
    "workingDirectory": "/tmp",
    "accountId": "462045008730"
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-pipeline/
example-linux-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-configuration/
example-linux-infra-config-uswest1",
    "name": "example-linux-infra-config-uswest1",
    "instanceProfileName": "example-linux-ib-baseline-admin",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-
configuration/example-linux-infra-config-uswest1"
    },
    "logging": {
      "s3Logs": {
        "s3BucketName": "amzn-s3-demo-bucket"
      }
```

```
    },
    "keyPair": "example-linux-key-pair-uswest1",
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-1:123456789012:example-linux-ibnotices-
uswest1",
    "dateCreated": "Feb 24, 2021 12:31:55 AM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-configuration/
example-linux-distribution",
    "name": "example-linux-distribution",
    "dateCreated": "Feb 24, 2021 12:31:56 AM",
    "distributions": [
      {
        "region": "us-west-1",
        "amiDistributionConfiguration": {}
      }
    ],
    "tags": {
      "internalId": "345abc67-8910-12d3-4ef5-67a8b90c12de",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-
configuration/example-linux-distribution"
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 28, 2022 1:13:45 AM",
  "outputResources": {
    "amis": [
      {
        "region": "us-west-1",
        "image": "ami-01a23bc4def5a6789",
        "name": "example-linux-image 2022-07-28T01-14-17.416Z",
        "accountId": "123456789012"
      }
    ]
  },
  "buildExecutionId": "ab0cd12e-34fa-5678-b901-2c3456d789e0",
  "testExecutionId": "6a7b8901-cdef-234a-56b7-8cd89ef01234",
  "distributionJobId": "1f234567-8abc-9d0e-1234-fa56b7c890de",
```

```
  "integrationJobId": "432109b8-afe7-6dc5-4321-0ba98f7654e3",
  "accountId": "123456789012",
  "osVersion": "Amazon Linux 2",
  "enhancedImageMetadataEnabled": true,
  "buildType": "USER_INITIATED",
  "tags": {
    "internalId": "901e234f-a567-89bc-0123-d4e567f89a01",
    "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3"
  }
}
```

L'exemple suivant montre la charge utile JSON d'un message typique qu'Image Builder publie en cas d'échec de la génération d'un pipeline pour une image Linux.

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image",
  "semver": 123794003928538027489912 4231,
  "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-image/1.0.0/7",
  "name": "My Example Image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 7,
  "state": {
    "status": "FAILED",
    "reason": "Image Failure reason."
  },
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-
image/1.0.0",
    "name": "My Example Image",
    "version": "1.0.0",
    "description": "Testing Image recipe",
    "components": [
      {
        "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-
example-image-component/1.0.0/1"
      }
    ],
    "platform": "Linux",
    "parentImage": "ami-0cd12345db678d90f",
```

```
    "dateCreated": "Jun 21, 2022 11:36:14 PM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
example-image/1.0.0"
    },
    "accountId": "123456789012"
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-
example-image-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/
my-example-infra-config",
    "name": "SNS topic Infra config",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
      "t2.micro"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/my-example-infra-config"
    },
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:example-pipeline-notification-
topic",
    "dateCreated": "Jul 5, 2022 7:31:53 PM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-
example-distribution-config",
    "name": "New distribution config",
    "dateCreated": "Dec 3, 2021 9:24:22 PM",
    "distributions": [
      {
        "region": "us-west-2",
        "amiDistributionConfiguration": {},
        "fastLaunchConfigurations": [
```

```
          {
            "enabled": true,
            "snapshotConfiguration": {
              "targetResourceCount": 2
            },
            "maxParallelLaunches": 2,
            "launchTemplate": {
              "launchTemplateId": "lt-01234567890"
            },
            "accountId": "123456789012"
          }
        ]
      }
    ],
    "tags": {
      "internalId": "1fecd23a-4f56-7f89-01e2-345678abbe90",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/my-example-distribution-config"
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 5, 2022 7:40:15 PM",
  "outputResources": {
    "amis": []
  },
  "accountId": "123456789012",
  "enhancedImageMetadataEnabled": true,
  "buildType": "SCHEDULED",
  "tags": {
    "internalId": "456c78b9-0e12-3f45-afb6-7e89b0f1a23b",
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image/1.0.0/7"
  }
}
```

Produits de conformité pour vos images Image Builder

Les normes de sécurité étant en constante évolution, il peut être difficile de maintenir la conformité et de protéger votre organisation contre les cybermenaces. Pour garantir la conformité de vos images personnalisées et le rester grâce à des mises à jour automatiques lorsque les éditeurs publient de nouvelles versions, Image Builder s'intègre AWS Marketplace aux produits de conformité et aux composants d'Image Builder.

Image Builder s'intègre aux produits de conformité suivants :

- Renforcement des critères de référence du Centre pour la sécurité Internet (CIS)

  Vous pouvez utiliser les images renforcées CIS et les composants de renforcement CIS associés pour créer des images personnalisées conformes aux dernières directives CIS Benchmarks de niveau 1. Les images CIS Hardened sont disponibles en AWS Marketplace. Pour en savoir plus sur la configuration et l'utilisation des images renforcées CIS et des composants de renforcement, consultez le Guide de démarrage rapide : Composants de renforcement CIS pour EC2 Image Builder dans le wiki de sécurité CIS.

  > **Note**
  >
  > Lorsque vous vous abonnez à une image renforcée CIS, vous avez également accès au composant de génération associé qui exécute un script pour appliquer les directives CIS Benchmark de niveau 1 pour votre configuration. Pour de plus amples informations, veuillez consulter composants de durcissement CIS.

- Guides techniques de mise en œuvre de la sécurité (STIG)

  Pour vous conformer aux normes STIG, vous pouvez utiliser Amazon-managed AWS Task Orchestrator and Executor (AWSTOE) les composants STIG dans vos recettes Image Builder. Les composants STIG analysent votre instance de build pour détecter les erreurs de configuration et exécutent un script de correction pour corriger les problèmes détectés. Nous ne pouvons pas garantir la conformité STIG pour les images que vous créez avec Image Builder. Vous devez travailler avec l'équipe de conformité de votre organisation pour vérifier que votre image finale est conforme. Pour une liste complète des composants AWSTOE STIG que vous pouvez utiliser dans vos recettes Image Builder, consultezAmazon a géré les composants de renforcement STIG pour Image Builder.

Surveillez les événements et les journaux dans Image Builder

Pour maintenir la fiabilité, la disponibilité et les performances de vos pipelines EC2 Image Builder, il est important de surveiller les événements et les journaux. Les événements et les journaux vous aident à avoir une vue d'ensemble et à approfondir les détails en cas d'échec d'un appel d'API. Image Builder s'intègre à des services capables d'envoyer des alertes et de déclencher des réponses automatisées lorsque les événements répondent aux critères que vous avez configurés.

Les rubriques suivantes décrivent les techniques de surveillance que vous pouvez utiliser via les services intégrés à Image Builder.

Surveiller les événements et les journaux

- Enregistrez les appels d'API Image Builder à l'aide de CloudTrail
- Surveillez les journaux d'Image Builder avec Amazon CloudWatch Logs

Enregistrez les appels d'API Image Builder à l'aide de CloudTrail

EC2 Image Builder est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Image Builder. CloudTrail capture tous les appels d'API pour Image Builder sous forme d'événements. Les appels capturés incluent des appels provenant de la console Image Builder et des appels de code vers les opérations de l'API Image Builder. Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour Image Builder. 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 faite à Image Builder, l'adresse IP à partir de laquelle la demande a été faite, qui a fait la demande, quand elle a été faite et des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Informations sur Image Builder dans CloudTrail

CloudTrail est activé sur votre compte Compte AWS lorsque vous créez le compte. Lorsqu'une activité se produit dans Image Builder, 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 Compte AWS. 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 site Compte AWS, y compris des événements pour Image Builder, créez un parcours. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal d'activité dans la console, il s'applique à toutes les régions Régions AWS. 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 les ressources suivantes :

- Présentation de la création d'un journal de suivi
- 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 d'Image Builder sont enregistrées CloudTrail et documentées dans le Guide de référence de l'API EC2 Image Builder. Par exemple, les appels aux `CreateImagePipeline`, `UpdateInfrastructureConfiguration`, et `StartImagePipelineExecution` les actions 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 les éléments suivants :

- Si la demande a été faite avec les informations d'identification de l'utilisateur root ou Gestion des identités et des accès AWS (IAM).
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour de plus amples informations, veuillez consulter l'élément userIdentity CloudTrail .

Surveillez les journaux d'Image Builder avec Amazon CloudWatch Logs

CloudWatch La prise en charge des journaux est activée par défaut. Image Builder conserve les journaux de génération sur l'instance pendant le processus de génération, puis les diffuse dans CloudWatch Logs une fois la génération terminée. Avant de créer l'image de construction finale, Image Builder supprime les journaux locaux de l'instance.

Journaux de création d'images

Les journaux de génération d'image contiennent les journaux détaillés d'une génération d'image individuelle. Image Builder écrit les journaux de création d'images dans le groupe Image Builder CloudWatch Logs et le flux suivants :

LogGroup: `/aws/imagebuilder/`*`ImageName`*

LogStream (x.x.x/x) : *`ImageVersion/ImageBuildVersion`*

Journaux d'exécution du pipeline

Image Builder écrit des journaux d'exécution du pipeline lorsqu'un pipeline d'images s'exécute ou lorsqu'un pipeline ignore une exécution planifiée, ainsi que des informations expliquant pourquoi cela s'est produit. Par exemple, lorsque les mises à jour des dépendances sont configurées pour un pipeline planifié, les scénarios suivants seront associés à une messagerie :

- Image Builder écrit une entrée de journal lorsqu'il ignore l'exécution du pipeline car aucune mise à jour des dépendances n'a été effectuée.
- Lorsqu'une dépendance est mise à jour, Image Builder enregistre l'ARN de la ressource modifiée.

Image Builder écrit les journaux d'exécution du pipeline dans le groupe et le flux Image Builder CloudWatch Logs suivants :

LogGroup: `/aws/imagebuilder/pipeline/`*`pipeline-name`*

LogStream: *`2025/09/01`* (la date d'exécution du pipeline au YYYY/MM/DD format)

Chaque journal de pipeline est ajouté au flux pour ce jour.

Groupes de journaux personnalisés

Si vous avez spécifié des groupes de journaux personnalisés pour la journalisation des images ou des pipelines, Image Builder écrit les journaux dans ceux LogGroup que vous avez spécifiés

dans votre pipeline, ou dans l'une des commandes de création d'image si vous avez exécuté la génération manuellement. Pour plus d'informations sur la façon dont cela fonctionne dans Image Builder, consultezConfigurer la journalisation du pipeline.

Se désinscrire

Vous pouvez désactiver le streaming des CloudWatch journaux en supprimant les autorisations suivantes associées au rôle personnalisé qui s'applique à votre cas d'utilisation.

- journaux : CreateLogGroup
- journaux : CreateLogStream
- journaux : PutLogEvents

Type de journal	Rôle personnalisé
Création d'images	Rôle d'exécution
Exécution du pipeline	Rôle d'exécution
Journaux des composants	Rôle du profil de l'instance

Pour un dépannage avancé, vous pouvez exécuter des commandes et des scripts prédéfinis à l'aide de AWS Systems Manager Run Command. Pour de plus amples informations, veuillez consulter Résoudre les problèmes liés à Image Builder.

Sécurité dans Image Builder

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 organisations les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit ceci comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : elle AWS est responsable de la protection de l'infrastructure qui s'exécute Services AWS dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Third-partyles auditeurs testent et vérifient régulièrement l'efficacité de notre sécurité dans le AWS cadre des programmes de . Pour en savoir plus sur les programmes de conformité qui s'appliquent à EC2 Image Builder, consultez Services AWS Scope by Compliance Program .
- 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'Image Builder. Les rubriques suivantes vous montrent comment configurer Image Builder pour répondre à vos objectifs de sécurité et de conformité. Vous apprenez également à en utiliser d'autres Services AWS qui vous aident à surveiller et à sécuriser vos ressources Image Builder.

Rubriques

- Protection des données et AWS modèle de responsabilité partagée dans Image Builder
- Intégration de la gestion des identités et des accès pour Image Builder
- Validation de conformité pour Image Builder
- Redondance et résilience des données dans Image Builder
- Sécurité de l'infrastructure dans Image Builder
- Gestion des correctifs pour les images d'Image Builder
- Meilleures pratiques de sécurité pour Image Builder

Protection des données et AWS modèle de responsabilité partagée dans Image Builder

Le modèle de responsabilité AWS https://aws.amazon.com/compliance/shared-responsibility-model/ s'applique à la protection des données dans EC2 Image Builder. Comme décrit dans ce modèle, AWS est responsable de la protection de l'infrastructure mondiale qui gère l'ensemble des AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez la FAQ sur la confidentialité des données et les conditions légales de la région . Pour plus d'informations sur la protection des données en Europe, consultez le Centre du règlement général sur la protection des données (RGPD).

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou Gestion des identités et des accès AWS (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- SSL/TLS À utiliser pour communiquer avec AWS les ressources. Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation des activités des utilisateurs avec AWS CloudTrail. Pour plus d'informations sur l'utilisation des CloudTrail sentiers pour enregistrer AWS des activités, consultez la section Utilisation des CloudTrail sentiers dans le Guide de AWS CloudTrail l'utilisateur.
- Utilisez des solutions de AWS cryptage, ainsi que tous les contrôles de sécurité par défaut qui s'y trouvent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-3 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez Norme FIPS (Federal Information Processing Standard) 140-3.

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Nom. Cela inclut lorsque vous travaillez avec Image Builder ou autre à Services AWS l'aide de la console, de l'API ou des AWS kits de développement logiciel. AWS CLI Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Votre responsabilité en ce qui concerne les images de disque ISO importées

Lorsque vous importez une image disque ISO Windows, l'ISO que vous fournissez fournit le système d'exploitation pour l'AMI de sortie. Dans le cadre du modèle de responsabilité partagée, il est de votre responsabilité de vérifier l'authenticité des médias.

L'importation s'exécute vous-même Compte AWS, sur une instance de génération qui utilise le rôle de profil d'instance défini dans la configuration de votre infrastructure. Lors de l'importation, Image Builder exécute des exécutables à partir de l'ISO et de l'image Windows appliquée avec les privilèges SYSTEM associés à ce rôle.

Une image ISO provenant d'une source non fiable peut affecter l'AMI de sortie et toutes les instances que vous lancez à partir de celle-ci. Procurez-vous des fichiers ISO auprès de Microsoft ou d'un revendeur agréé. Pour de plus amples informations, veuillez consulter Importez des images de disque ISO Windows vérifiées avec Image Builder.

Chiffrement et gestion des clés dans Image Builder

Image Builder chiffre les données en transit et au repos par défaut à l'aide d'une clé KMS appartenant au service, sauf dans les cas suivants :

- Composants personnalisés : Image Builder chiffre les composants personnalisés à l'aide de votre clé KMS par défaut ou d'une clé KMS appartenant à un service.
- Flux de travail liés aux images : Image Builder peut chiffrer vos flux de travail liés aux images à l'aide d'une clé gérée par le client si vous spécifiez cette clé lors de la création du flux de travail. Image Builder gère le chiffrement et le déchiffrement à l'aide de votre clé afin d'exécuter les flux de travail que vous avez configurés pour vos images.

Vous pouvez gérer vos propres clés via AWS KMS. Cependant, vous n'êtes pas autorisé à gérer la clé KMS Image Builder détenue par Image Builder. Pour plus d'informations sur la gestion de vos clés KMS avec AWS Key Management Service, consultez Getting Started dans le Guide du AWS Key Management Service développeur.

Contexte de chiffrement

Pour fournir un contrôle supplémentaire de l'intégrité et de l'authenticité de vos données chiffrées, vous pouvez inclure un contexte de cryptage lorsque vous chiffrez les données. Lorsqu'une ressource est chiffrée à l'aide d'un contexte de chiffrement, le contexte est lié AWS KMS cryptographiquement au texte chiffré. La ressource ne peut être déchiffrée que si le demandeur fournit une correspondance exacte et sensible aux majuscules et minuscules pour le contexte.

Les exemples de politiques présentés dans cette section utilisent un contexte de chiffrement qui ressemble à l'Amazon Resource Name (ARN) d'une ressource de flux de travail Image Builder.

Chiffrez les flux d'images à l'aide d'une clé gérée par le client

Pour ajouter une couche de protection, vous pouvez chiffrer les ressources de votre flux de travail Image Builder avec votre propre clé gérée par le client. Si vous utilisez votre clé gérée par le client pour chiffrer les flux de travail Image Builder que vous créez, vous devez autoriser l'accès dans la politique de clé pour qu'Image Builder utilise votre clé lorsqu'il chiffre et déchiffre les ressources du flux de travail. Vous pouvez supprimer cet accès à tout moment. Toutefois, Image Builder n'aura accès à aucun flux de travail déjà chiffré si vous révoquez l'accès à la clé.

Le processus permettant d'accorder à Image Builder l'accès à l'utilisation de votre clé gérée par le client comporte deux étapes, comme suit :

Étape 1 : ajouter des autorisations de politique clés pour les flux de travail d'Image Builder

Pour permettre à Image Builder de chiffrer et de déchiffrer les ressources de flux de travail lorsqu'il crée ou utilise ces flux de travail, vous devez spécifier des autorisations dans la politique de clé KMS.

Cet exemple de politique clé autorise l'accès aux pipelines Image Builder afin de chiffrer les ressources du flux de travail pendant le processus de création et de déchiffrer les ressources du flux de travail pour les utiliser. La politique accorde également l'accès aux principaux administrateurs. Le contexte de chiffrement et la spécification des ressources utilisent un caractère générique pour couvrir toutes les régions dans lesquelles vous disposez de ressources de flux de travail.

Comme condition préalable à l'utilisation des flux de travail d'imagerie, vous avez créé un rôle d'exécution de flux de travail IAM qui autorise Image Builder à exécuter des actions de flux de travail.

Le principal de la première instruction présentée dans l'exemple de politique clé présenté ici doit spécifier votre rôle d'exécution du flux de travail IAM.

Pour plus d'informations sur les clés gérées par le client, consultez la section Gestion de l'accès aux clés gérées par le client dans le Guide du AWS Key Management Service développeur.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "Allow access to build images with encrypted workflow",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/YourImageBuilderExecutionRole"
   },
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "*",
   "Condition": {
    "StringLike": {
     "kms:EncryptionContext:aws:imagebuilder:arn":
 "arn:aws:imagebuilder:*:111122223333:workflow/*"
    }
   }
  },
  {
   "Sid": "Allow access for key administrators",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:root"
   },
   "Action": [
    "kms:*"
   ],
   "Resource": "arn:aws:kms:*:111122223333:key/*"
  }
 ]
}
```

Étape 2 : accordez un accès clé à votre rôle d'exécution du flux de travail

Le rôle IAM assumé par Image Builder pour exécuter vos flux de travail nécessite l'autorisation d'utiliser votre clé gérée par le client. Sans accès à votre clé, Image Builder ne sera pas en mesure de chiffrer ou de déchiffrer les ressources de votre flux de travail à l'aide de celle-ci.

Modifiez la politique de votre rôle d'exécution du flux de travail pour ajouter la déclaration de stratégie suivante.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "AllowAccessToWorkflowKey",
   "Effect": "Allow",
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "arn:aws:kms:us-west-2:111122223333:key/key_ID",
   "Condition": {
    "StringLike": {
     "kms:EncryptionContext:aws:imagebuilder:arn":
 "arn:aws:imagebuilder:*:111122223333:workflow/*"
    }
   }
  }
 ]
}
```

AWS CloudTrail événements pour les flux de travail liés aux images

Les exemples suivants présentent des AWS CloudTrail entrées typiques pour le chiffrement et le déchiffrement des flux d'images qui sont stockés avec une clé gérée par le client.

Exemple : GenerateDataKey

Cet exemple montre à quoi peut ressembler un CloudTrail événement lorsqu'Image Builder invoque l'action d' AWS KMS GenerateDataKeyAPI à partir de l'action d'CreateWorkflowAPI Image Builder. Image Builder doit chiffrer un nouveau flux de travail avant de créer la ressource de flux de travail.


```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
   }
  },
  "invokedBy": "imagebuilder.amazonaws.com"
 },
 "eventTime": "2023-11-21T20:31:03Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "GenerateDataKey",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "key value"
  },
  "keyId": "arn:aws:kms:us-west-2:111122223333:alias/ExampleKMSKey",
  "numberOfBytes": 32
 },
```

```
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
   "type": "AWS::KMS::Key",
   "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
  }
 ],
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "111122223333",
 "eventCategory": "Management"
}
```

Exemple : décrypter

Cet exemple montre à quoi peut ressembler un CloudTrail événement lorsqu'Image Builder invoque l'action d' AWS KMS DecryptAPI à partir de l'action d'GetWorkflowAPI Image Builder. Les pipelines Image Builder doivent déchiffrer une ressource de flux de travail avant de pouvoir l'utiliser.

```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
```

```
    }
   },
   "invokedBy": "imagebuilder.amazonaws.com"
 },
 "eventTime": "2023-11-21T20:34:25Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "Decrypt",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
  "keyId": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLEzzzzz",
  "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "ABC123def4567890abc12345678/90dE/F123abcDEF+4567890abc123D
+ef1=="
  }
 },
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
   "type": "AWS::KMS::Key",
   "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
  }
 ],
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "111122223333",
 "eventCategory": "Management"
}
```

Stockage des données dans Image Builder

Image Builder ne stocke aucun de vos journaux dans le service. Tous les journaux sont enregistrés sur votre instance Amazon EC2 utilisée pour créer l'image, ou dans les journaux d'automatisation de Systems Manager.

Inter-network Confidentialité du trafic dans Image Builder

Les connexions sont sécurisées entre Image Builder et les sites locaux, entre les zones de zone de zone au sein d'une même AWS région et entre les AWS régions via HTTPS. Il n'existe aucun lien direct entre les comptes.

Intégration de la gestion des identités et des accès pour Image Builder

Rubriques

- Public ciblé
- Authentification par des identités
- Comment Image Builder fonctionne avec les politiques et les rôles IAM
- Gérez les périmètres de données pour l'accès au téléchargement du bucket S3 dans Image Builder
- Politiques basées sur l'identité d'Image Builder
- Autorisations IAM pour les flux de travail personnalisés
- Politiques basées sur les ressources d'Image Builder
- Utilisation AWS politiques gérées pour EC2 Image Builder
- Utiliser des rôles liés à un service IAM pour Image Builder
- Résoudre les problèmes d'IAM dans Image Builder

Public ciblé

La façon dont vous utilisez Gestion des identités et des accès AWS (IAM) varie en fonction de votre rôle :

- Utilisateur du service : demandez des autorisations à votre administrateur si vous ne pouvez pas accéder aux fonctionnalités (voir Résoudre les problèmes d'IAM dans Image Builder)
- Administrateur du service : déterminez l'accès des utilisateurs et soumettez les demandes d'autorisation (voir Comment Image Builder fonctionne avec les politiques et les rôles IAM)
- Administrateur IAM : rédigez des politiques pour gérer l'accès (voir Politiques basées sur l'identité d'Image Builder)

Authentification par des identités

Pour obtenir des informations détaillées sur la manière de fournir une authentification aux personnes et aux processus de votre entreprise Compte AWS, consultez la section Identités dans le guide de l'utilisateur IAM.

Comment Image Builder fonctionne avec les politiques et les rôles IAM

Avant d'utiliser IAM pour gérer l'accès à Image Builder, découvrez quelles fonctionnalités IAM peuvent être utilisées avec Image Builder.

Pour obtenir une vue d'ensemble de la façon dont Image Builder 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 Image Builder

Prend en charge les politiques basées sur l'identité : oui

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. 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 l'identité pour Image Builder

Pour consulter des exemples de politiques basées sur l'identité d'Image Builder, consultez. Politiques basées sur l'identité d'Image Builder

Politiques basées sur les ressources dans Image Builder

Prend en charge les politiques basées sur les ressources : oui

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Actions politiques pour Image Builder

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. Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour consulter la liste des actions d'Image Builder, consultez la section Actions définies par EC2 Image Builder dans le Service Authorization Reference.

Les actions de stratégie dans Image Builder utilisent le préfixe suivant avant l'action :

```
imagebuilder
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
 "imagebuilder:action1",
 "imagebuilder:action2"
 ]
```

Pour consulter des exemples de politiques basées sur l'identité d'Image Builder, consultez. Politiques basées sur l'identité d'Image Builder

Ressources relatives aux politiques pour Image Builder

Prend en charge les ressources 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 de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Pour consulter la liste des types de ressources Image Builder et leurs caractéristiques ARNs, consultez la section Ressources définies par EC2 Image Builder dans le Service Authorization Reference. Pour savoir avec quelles actions vous pouvez spécifier l'ARN de chaque ressource, voir Actions définies par EC2 Image Builder.

Pour consulter des exemples de politiques basées sur l'identité d'Image Builder, consultez. Politiques basées sur l'identité d'Image Builder

Clés de conditions de politique pour Image Builder

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` indique à quel moment les instructions s'exécutent en fonction de critères définis. 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. 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.

Pour consulter la liste des clés de condition d'Image Builder, consultez la section Clés de condition pour EC2 Image Builder dans le Service Authorization Reference. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, voir Actions définies par EC2 Image Builder.

Pour consulter des exemples de politiques basées sur l'identité d'Image Builder, consultez. Politiques basées sur l'identité d'Image Builder

ACLs dans Image Builder

Supports ACLs : Non

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

ABAC avec Image Builder

Prend en charge ABAC (identifications dans les politiques) : partiellement

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction des attributs nommés balise. Vous pouvez associer des balises aux entités et aux AWS ressources IAM, puis concevoir des politiques ABAC pour autoriser les opérations lorsque la balise du principal correspond à la balise de la ressource.

Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur ABAC, consultez Définition d'autorisations avec l'autorisation ABAC dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès par attributs (ABAC) dans le Guide de l'utilisateur IAM.

Utilisation d'informations d'identification temporaires avec Image Builder

Prend en charge les informations d'identification temporaires : oui

Les informations d'identification temporaires fournissent un accès à court terme aux AWS ressources et sont automatiquement créées lorsque vous utilisez la fédération ou que vous changez de rôle. 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 et Services AWS compatibles avec IAM dans le Guide de l'utilisateur IAM.

Autorisations principales interservices pour Image Builder

Prend en charge les sessions d'accès direct (FAS) : oui

Les sessions d'accès direct (FAS) utilisent les autorisations du principal appelant et Service AWS, combinées Service AWS à la demande d'envoi de demandes aux services en aval. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez la section Sessions de transmission d'accès.

Rôles de service pour Image Builder

Prend en charge les rôles de service : oui

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

> **Warning**
>
> La modification des autorisations associées à un rôle de service peut perturber les fonctionnalités d'Image Builder. Modifiez les rôles de service uniquement lorsque Image Builder fournit des instructions à cet effet.

Rôles liés à un service pour Image Builder

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 de détails sur le rôle lié au service Image Builder, consultez. Utiliser des rôles liés à un service IAM pour Image Builder

Politiques basées sur l'identité d'Image Builder

Avec les politiques basées sur l'identité IAM, vous pouvez spécifier des actions et des ressources autorisées ou refusées ainsi que les conditions selon lesquelles les actions sont autorisées ou refusées. Image Builder prend en charge des actions, des ressources et des clés de condition spécifiques. Pour plus d'informations sur tous les éléments que vous utilisez dans une politique JSON, consultez la section Actions, ressources et clés de condition pour Amazon EC2 Image Builder dans le guide de l'utilisateur IAM.

Actions

Les actions de politique dans Image Builder utilisent le préfixe suivant avant l'action : `imagebuilder:` Les déclarations de politique doivent inclure un élément `Action` ou `NotAction`. Image Builder définit son propre ensemble d'actions décrivant les tâches que vous pouvez effectuer avec ce service.

Pour spécifier plusieurs actions dans une seule déclaration, séparez-les par des virgules comme suit :

```
"Action": [
 "imagebuilder:action1",
 "imagebuilder: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 `List`, incluez l'action suivante :

```
"Action": "imagebuilder:List*"
```

Pour consulter la liste des actions d'Image Builder, reportez-vous à la section Actions, ressources et clés de condition Services AWS dans le guide de l'utilisateur d'IAM.

Gestion de l'accès à l'aide de politiques

Pour obtenir des informations détaillées sur la façon de gérer l'accès en AWS créant des politiques et en les associant aux identités ou aux AWS ressources IAM, consultez la section Politiques et autorisations du guide de l'utilisateur IAM.

Le rôle IAM que vous associez à votre profil d'instance doit être autorisé à exécuter les composants de génération et de test inclus dans votre image. Les politiques de rôle IAM suivantes doivent être associées au rôle IAM associé au profil d'instance :

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- AmazonSSMManagedInstanceCore

Ressources

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

L'ARN est composé de plusieurs nœuds qui permettent d'identifier la ressource et de garantir que le nom est unique. Les derniers nœuds du nom incluent plusieurs variantes de mise en forme pour le type, le nom et l'ID de la ressource. Lorsque Image Builder crée une ressource, il utilise le format suivant :

`arn:aws:imagebuilder:`*`region`*`:`*`owner`*`:`*`resource-type/resource-name/version/build-version`*

> **Note**
>
> La version de compilation n'est pas toujours incluse dans l'ARN de la ressource. Cependant, certaines opérations d'API, telles que GetComponent, nécessitent la version de compilation pour identifier de manière unique une ressource à récupérer.

Pour les ressources utilisées par Image Builder dans ses recettes, telles que l'image de base ou les composants, le nœud propriétaire peut être l'un des suivants :

- Le numéro de compte du propriétaire de la ressource
- Pour les ressources gérées par Amazon : `aws`
- Pour les AWS Marketplace ressources : `aws-marketplace`

L'exemple suivant montre l'ARN d'un composant géré pour installer l' CloudWatch agent Amazon sous Linux :

```
arn:aws:imagebuilder:us-east-1:aws:component/amazon-cloudwatch-agent-linux/1.0.1/1
```

Cet exemple montre l'ARN d'un composant géré fictif provenant du : AWS Marketplace

```
arn:aws:imagebuilder:us-east-1:aws-marketplace:component/example-linux-software-component/1.0.1
```

Pour plus d'informations sur l'obtention d'une liste de composants, notamment sur l'utilisation d'un filtre de propriété, consultezListe des composants d'Image Builder.

Exemple ARNs

Voici quelques exemples de ressources ARNs que vous pouvez spécifier dans une politique IAM :

- ARN de l'instance

  ```
  "Resource": "arn:aws:imagebuilder:us-east-1:111122223333:instance/i-1234567890abcdef0"
  ```

- Exemple de code générique (*) pour spécifier toutes les instances d'un compte donné

  ```
  "Resource": "arn:aws:imagebuilder:us-east-1:111122223333:instance/*"
  ```

- Exemple de caractère générique (*) pour spécifier toutes les versions d'un flux de travail d'images géré

  ```
  "Resource": "arn:aws:imagebuilder:us-east-1:aws:workflow/build/build-image/*"
  ```

- ARN de l'image gérée

  ```
  "Resource": "arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2-arm64/2024.12.17/1"
  ```

- Exemple de caractère générique (*) pour spécifier toutes les versions d'une image gérée

```
"Resource": "arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2-arm64/x.x.x"
```

De nombreuses actions de l'API EC2 Image Builder impliquent plusieurs ressources. Pour spécifier plusieurs ressources dans une seule instruction, séparez-les ARNs par des virgules.

```
"Resource": [
   "resource1",
   "resource2"
]
```

Clés de condition

Image Builder fournit des clés de condition spécifiques au service et prend en charge l'utilisation de certaines clés de condition globales. Pour voir toutes les clés de condition AWS globales, consultez la section Clés contextuelles de condition AWS globale dans le guide de l'utilisateur IAM. Les clés de condition spécifiques au service suivantes sont fournies.

générateur d'images : CreatedResourceTagKeys

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour filtrer l'accès en fonction de la présence de clés de balise dans la demande. Cela vous permet de gérer les ressources créées par Image Builder.

Disponibilité — Cette clé n'est disponible que pour le `CreateInfrastrucutreConfiguration` et `UpdateInfrastructureConfiguration` APIs.

générateur d'images :/CreatedResourceTag<key>

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour filtrer l'accès en fonction des paires clé-valeur de balise associées à la ressource créée par Image Builder. Cela vous permet de gérer les ressources d'Image Builder via des balises définies.

Disponibilité — Cette clé n'est disponible que pour le `CreateInfrastrucutreConfiguration` et `UpdateInfrastructureConfiguration` APIs.

générateur d'images : LifecyclePolicyResourceType

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour filtrer l'accès en fonction du type de ressource Lifecycle spécifié dans la demande.

La valeur de cette clé peut être `AMI_IMAGE` soit`CONTAINER_IMAGE`.

Disponibilité — Cette clé n'est disponible que pour le `CreateLifecyclePolicy` et `UpdateLifecyclePolicy`APIs.

Générateur d'images : EC2 MetadataHttpTokens

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour filtrer l'accès en fonction de l'exigence de jeton HTTP de métadonnées d'instance EC2 spécifiée dans la demande.

La valeur de cette clé peut être l'une `optional` ou l'autre`required`.

Disponibilité — Cette clé n'est disponible que pour le `CreateInfrastrucutreConfiguration` et `UpdateInfrastructureConfiguration` APIs.

générateur d'images : StatusTopicArn

Fonctionne avec des opérateurs de chaîne.

Utilisez cette touche pour filtrer l'accès par l'ARN de la rubrique SNS dans la demande à laquelle les notifications d'état du terminal seront publiées.

Disponibilité — Cette clé n'est disponible que pour le `CreateInfrastrucutreConfiguration` et `UpdateInfrastructureConfiguration` APIs.

Exemples

Pour consulter des exemples de politiques basées sur l'identité d'Image Builder, consultez. Politiques basées sur l'identité d'Image Builder

Politiques basées sur les ressources d'Image Builder

Les politiques basées sur les ressources spécifient les actions qu'un principal spécifié peut effectuer sur la ressource Image Builder et dans quelles conditions. Image Builder prend en charge les politiques d'autorisation basées sur les ressources pour les composants, les images et les recettes

d'images. Les politiques basées sur les ressources permettent d'accorder une autorisation à d'autres comptes en fonction des ressources. Vous pouvez également utiliser une politique basée sur les ressources pour autoriser un AWS service à accéder à vos composants, images et recettes d'images.

Pour plus d'informations sur la façon d'associer une politique basée sur les ressources à un composant, à une image ou à une recette d'image, consultez. Partagez les ressources d'Image Builder avec AWS RAM

> **Note**
>
> Lorsque vous mettez à jour une politique de ressources à l'aide d'Image Builder, la mise à jour apparaît dans la console RAM.

Autorisation basée sur les balises Image Builder

Vous pouvez associer des balises aux ressources d'Image Builder ou transmettre des balises dans une demande à Image Builder. 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 `imagebuilder:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`. Pour plus d'informations sur le balisage des ressources Image Builder, consultezMarquez une ressource depuis AWS CLI.

Rôles IAM dans Image Builder

Un rôle IAM est une entité au sein de vous Compte AWS qui possède des autorisations spécifiques.

Utilisation d'informations d'identification temporaires avec Image Builder

Vous pouvez utiliser des informations d'identification temporaires pour vous connecter à l'aide de la fédération, endosser un rôle IAM ou encore pour endosser un rôle intercompte. Vous obtenez des informations d'identification de sécurité temporaires en appelant des opérations d' AWS STS API telles que AssumeRoleou GetFederationToken.

Rôles liés à un service

Les rôles liés à un service permettent Services AWS d'accéder aux ressources d'autres services pour effectuer une action en votre nom. Les rôles liés à un service s'affichent dans votre compte IAM et sont la propriété du service. Un utilisateur disposant d'un accès administratif peut consulter mais pas modifier les autorisations pour les rôles liés à un service.

Image Builder prend en charge les rôles liés à un service. Pour plus d'informations sur la création ou la gestion des rôles liés au service Image Builder, consultez. Utiliser des rôles liés à un service IAM pour Image Builder

Rôles du service

Cette fonction permet à un service d'endosser une fonction du service en votre nom. Ce rôle autorise le service à accéder à des ressources d'autres services pour effectuer une action en votre nom. Les rôles de service s'affichent dans votre compte IAM et sont la propriété du compte. Cela signifie qu'un utilisateur disposant d'un accès administratif peut modifier les autorisations associées à ce rôle. Toutefois, une telle action peut perturber le bon fonctionnement du service.

Gérez les périmètres de données pour l'accès au téléchargement du bucket S3 dans Image Builder

EC2 Image Builder gère deux catégories AWS de compartiments S3 appartenant au service qui contiennent les ressources téléchargeables nécessaires pour exécuter les charges de travail Image Builder sur votre compte. Si vous utilisez des périmètres de données pour contrôler l'accès à Amazon S3 dans votre environnement, vous devrez peut-être autoriser explicitement l'accès à ces compartiments. Vous pouvez utiliser l'ARN ou l'URL du compartiment pour autoriser la mise en liste de ces compartiments, en fonction de la manière dont vous contrôlez l'accès à Amazon S3.

Scripts d'amorçage pour la gestion des composants (obligatoire)

Ce compartiment S3 contient des scripts d'amorçage permettant de configurer l' AWSTOE application sur les instances EC2 utilisées pour créer des images. Image Builder a besoin d'un accès pour télécharger les scripts nécessaires à la création et au test de nouvelles images.

- ARN du compartiment S3 : `arn:<AWS partition>:s3:::ec2imagebuilder-managed-resources-<AWS Region>-prod`
- URL du compartiment S3 : `https://ec2imagebuilder-managed-resources-<AWS Region>.s3.<AWS Region>.<AWS partition-specific domain name>`

Composants gérés

Ce compartiment S3 contient les charges utiles des packages pour les composants gérés par Amazon. Image Builder a besoin d'un accès pour télécharger tous les composants gérés configurés dans vos recettes.

- ARN du compartiment S3 : `arn:<AWS partition>:s3:::ec2imagebuilder-toe-<AWS Region>-prod`

- URL du compartiment S3 : `https://ec2imagebuilder-toe-<AWS Region>.s3.<AWS Region>.<AWS partition-specific domain name>`

Politiques basées sur l'identité d'Image Builder

Rubriques

- Bonnes pratiques en matière de politiques basées sur l'identité
- Utilisation de la console Image Builder

Bonnes pratiques en matière de politiques basées sur l'identité

Les politiques basées sur l'identité déterminent si quelqu'un peut créer, accéder ou supprimer des ressources Image Builder dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accordez les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utilisez des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

- Utilisez l'Analyseur d'accès IAM pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : l'Analyseur d'accès IAM valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politiques avec IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue. Compte AWS Pour exiger la MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Sécurisation de l'accès aux API avec MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Utilisation de la console Image Builder

Pour accéder à la console EC2 Image Builder, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations vous permettent de répertorier et d'afficher des informations détaillées sur les ressources Image Builder de votre Compte AWS. Si vous créez une politique basée sur l'identité qui est plus restrictive que les autorisations minimales requises, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs et rôles IAM) tributaires de cette politique.

Pour garantir que vos entités IAM peuvent utiliser la console Image Builder, vous devez leur associer l'une des politiques AWS gérées suivantes :

- AWSImageBuilderReadOnlyAccess policy
- AWSImageBuilderFullAccess policy

Pour plus d'informations sur les politiques gérées par Image Builder, consultezUtilisation AWS politiques gérées pour EC2 Image Builder.

> **Important**
>
> La AWSImageBuilderFullAccesspolitique est requise pour créer le rôle lié au service Image Builder. Lorsque vous associez cette politique à une entité IAM, vous devez également

associer la stratégie personnalisée suivante et inclure les ressources que vous souhaitez utiliser et qui ne figurent pas `imagebuilder` dans le nom de la ressource :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:*:*:*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::*:instance-profile/*imagebuilder*",
                "arn:aws:iam::*:role/*imagebuilder*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3::*:*imagebuilder*"
```

```
        }
    ]
}
```

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui appellent uniquement l'API AWS CLI ou l' AWS API. Autorisez plutôt l'accès à uniquement aux actions qui correspondent à l'opération d'API que vous tentez d'effectuer.

Autorisations IAM pour les flux de travail personnalisés

Lorsque vous utilisez des flux de travail personnalisés comportant des étapes spécifiquesRegisterImage, par exemple, des autorisations IAM supplémentaires peuvent être requises au-delà des politiques gérées par Image Builder standard. Cette section décrit les autorisations supplémentaires nécessaires pour les actions d'étapes de flux de travail personnalisées.

RegisterImage autorisations relatives aux actions par étapes

L'action de l'`RegisterImage`étape nécessite des autorisations Amazon EC2 spécifiques pour enregistrer AMIs et éventuellement récupérer les balises de capture instantanée. Lorsque vous utilisez le `includeSnapshotTags` paramètre, des autorisations supplémentaires sont nécessaires pour décrire les instantanés.

Autorisations requises pour effectuer une action par RegisterImage étapes :

Pour toutes les ressources, autorisez les actions suivantes :

- `ec2:RegisterImage`
- `ec2:DescribeSnapshots`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RegisterImage",
                "ec2:DescribeSnapshots"
            ],
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*::image/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "RegisterImage"
                }
            }
        }
    ]
}
```

Détails de l'autorisation :

- `ec2:RegisterImage`- Nécessaire pour enregistrer de nouvelles données AMIs à partir de snapshots
- `ec2:DescribeSnapshots`- Nécessaire lors de l'utilisation `includeSnapshotTags: true` pour récupérer des balises de capture en vue de les fusionner avec des balises AMI
- `ec2:CreateTags`- Nécessaire pour appliquer des balises à l'AMI enregistrée, y compris les balises par défaut d'Image Builder et les balises de capture d'écran fusionnées

> **Note**
>
> L'`ec2:DescribeSnapshots`autorisation n'est utilisée que lorsque le `includeSnapshotTags` paramètre est défini sur`true`. Si vous n'utilisez pas cette fonctionnalité, vous pouvez omettre cette autorisation.

Comportement de fusion de balises :

Lorsque cette option `includeSnapshotTags` est activée, l'action de l' RegisterImage étape permet de :

- Récupérez les balises à partir du premier instantané spécifié dans le mappage des périphériques en mode bloc

- Exclure toutes les balises AWS réservées (celles dont les clés commencent par « aws : »)
- Fusionnez les balises de capture d'écran avec les balises d'enregistrement AMI par défaut d'Image Builder
- Donnez la priorité aux balises Image Builder en cas de conflit entre les clés de balise

Politiques basées sur les ressources d'Image Builder

Pour plus d'informations sur la création d'un composant, consultezUtilisez des composants pour personnaliser l'image de votre Image Builder.

Restreindre l'accès aux composants Image Builder à des adresses IP spécifiques

L'exemple suivant autorise tout utilisateur à effectuer des opérations Image Builder sur des composants. Toutefois, la demande doit provenir de la plage d'adresses IP indiquée dans la condition.

La condition contenue dans cette déclaration identifie la plage 54.240.143.* d'adresses IP autorisées du protocole Internet version 4 (IPv4), à une exception près : 54.240.143.188.

Le `Condition` bloc utilise les `NotIpAddress` conditions `IpAddress` et et la clé de `aws:SourceIp` condition, qui est une clé AWS de condition étendue. Pour plus d'informations sur ces clés de condition, consultez la section Spécification de conditions dans une politique. Les `aws:sourceIp` IPv4 valeurs utilisent la notation CIDR standard. Pour plus d'informations, consultez Opérateurs de condition d'adresse IP dans le Guide de l'utilisateur IAM.

JSON

```
{
  "Version":"2012-10-17",
  "Id": "IBPolicyId1",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Action": "imagebuilder:GetComponent",
      "Resource": "arn:aws:imagebuilder:*::examplecomponent/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"},
```

```
        "NotIpAddress": {"aws:SourceIp": "54.240.143.188/32"}
      }
    }
  ]
}
```

Utilisation AWS politiques gérées pour EC2 Image Builder

Une politique AWS gérée est une politique autonome créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous les AWS clients. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle politique Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d'informations, consultez Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

AWSImageBuilderFullAccess policy

La AWSImageBuilderFullAccesspolitique accorde un accès complet aux ressources d'Image Builder pour le rôle auquel il est rattaché, ce qui permet au rôle de répertorier, de décrire, de créer, de mettre à jour et de supprimer les ressources d'Image Builder. La politique accorde également des autorisations ciblées aux personnes associées Services AWS qui sont nécessaires, par exemple, pour vérifier les ressources ou pour afficher les ressources actuelles du compte dans le Console de gestion AWS.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes :

- Image Builder : l'accès administratif est accordé afin que le rôle puisse répertorier, décrire, créer, mettre à jour et supprimer des ressources Image Builder.
- Amazon EC2 — L'accès est accordé pour les actions Amazon EC2 Describez nécessaires pour vérifier l'existence des ressources ou obtenir des listes de ressources appartenant au compte.
- IAM — L'accès est accordé pour obtenir et utiliser des profils d'instance dont le nom contient « imagebuilder », pour vérifier l'existence du rôle lié au service Image Builder via l'action `iam:GetRole` API et pour créer le rôle lié au service Image Builder.
- License Manager — L'accès est accordé pour répertorier les configurations de licence ou les licences pour une ressource.
- Amazon S3 — L'accès est accordé pour répertorier les buckets appartenant au compte, ainsi que les buckets Image Builder dont le nom contient « imagebuilder ».
- Amazon SNS — Des autorisations d'écriture sont accordées à Amazon SNS pour vérifier la propriété des rubriques contenant « imagebuilder ».

Pour voir les autorisations de cette stratégie, consultez AWSImageBuilderFullAccess dans le AWS Guide de référence des stratégies gérées par.

AWSImageBuilderReadOnlyAccess policy

La AWSImageBuilderReadOnlyAccesspolitique fournit un accès en lecture seule à toutes les ressources d'Image Builder. Des autorisations sont accordées pour vérifier que le rôle lié au service Image Builder existe via l'action d'`iam:GetRole`API.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes :

- Image Builder : l'accès est accordé pour un accès en lecture seule aux ressources Image Builder.
- IAM — L'accès est accordé pour vérifier l'existence du rôle lié au service Image Builder via l'`iam:GetRole`action API.

Pour voir les autorisations de cette stratégie, consultez AWSImageBuilderReadOnlyAccess dans le AWS Guide de référence des stratégies gérées par.

AWSServiceRoleForImageBuilder policy

La AWSServiceRoleForImageBuilderpolitique autorise Image Builder à appeler Services AWS en votre nom.

Détails de l'autorisation

Cette politique est attachée au rôle lié au service Image Builder lorsque le rôle est créé via Systems Manager. Pour plus d'informations sur le rôle lié au service Image Builder, consultez. Utiliser des rôles liés à un service IAM pour Image Builder

La stratégie inclut les autorisations suivantes :

- CloudWatch Journaux — L'accès est accordé pour créer et télécharger CloudWatch des journaux dans tout groupe de journaux dont le nom commence par`/aws/imagebuilder/`.
- Amazon EC2 — L'accès est accordé à Image Builder pour créer, prendre des instantanés et enregistrer des images (AMI) qu'il crée et lancer des instances EC2 sur votre compte. Image Builder utilise les instantanés, les volumes, les interfaces réseau, les sous-réseaux, les groupes de sécurité, la configuration de licence et les paires de clés associés selon les besoins, à condition que l'image, l'instance et les volumes créés ou utilisés soient balisés avec `CreatedBy: EC2 Image Builder` ou. `CreatedBy: EC2 Fast Launch`

  Image Builder peut obtenir des informations sur les images Amazon EC2, les attributs des instances, le statut des instances, les types d'instances disponibles pour votre compte, les modèles de lancement, les sous-réseaux, les hôtes et les balises de vos ressources Amazon EC2.

  Image Builder peut mettre à jour les paramètres de l'image pour activer ou désactiver le lancement plus rapide des instances Windows dans votre compte, où l'image est étiquetée avec`CreatedBy: EC2 Image Builder`.

  En outre, Image Builder peut démarrer, arrêter et mettre fin aux instances qui s'exécutent sur votre compte, partager des instantanés Amazon EBS, créer et mettre à jour des images et des modèles de lancement, annuler l'enregistrement d'images existantes, ajouter des balises et répliquer des images sur des comptes auxquels vous avez accordé des autorisations via la politique. Ec2ImageBuilderCrossAccountDistributionAccess Le balisage Image Builder est obligatoire pour toutes ces actions, comme décrit précédemment.

  Image Builder peut également associer des filigranes d'AMI aux images balisées`CreatedBy: EC2 Image Builder`, afin que les filigranes se propagent vers des AMI dérivées.

- Amazon ECR — L'accès est accordé à Image Builder pour créer un référentiel si nécessaire pour les analyses de vulnérabilité des images de conteneurs, et pour étiqueter les ressources qu'il crée afin de limiter la portée de ses opérations. Image Builder est également autorisé à supprimer les images de conteneur qu'il a créées pour les scans après avoir pris des instantanés des vulnérabilités.
- EventBridge— L'accès est accordé à Image Builder pour créer et gérer EventBridge des règles.
- IAM — L'accès est accordé à Image Builder pour transmettre n'importe quel rôle de votre compte à Amazon EC2 et à VM. Import/Export
- Image Builder — L'accès est accordé à Image Builder pour déclencher automatiquement des pipelines au nom de l'utilisateur selon un calendrier défini par l'utilisateur et appliquer des balises aux images obtenues. Utilisé uniquement lorsque la `imageTags` propriété est définie sur un pipeline et que ce pipeline est configuré pour s'exécuter selon un calendrier.
- Amazon Inspector — L'accès est accordé à Image Builder pour déterminer quand Amazon Inspector termine les scans des instances de build et pour recueillir les résultats des images configurées pour l'autoriser.
- AWS KMS— L'accès est accordé à Amazon EBS pour chiffrer, déchiffrer ou rechiffrer les volumes Amazon EBS. Cela est essentiel pour garantir que les volumes chiffrés fonctionnent lorsque Image Builder crée une image.
- License Manager — L'accès est accordé à Image Builder pour mettre à jour les spécifications du License Manager via`license-manager:UpdateLicenseSpecificationsForResource`.
- Amazon SNS — Les autorisations d'écriture sont accordées pour toutes les rubriques Amazon SNS de votre compte.
- Systems Manager — L'accès est accordé à Image Builder pour répertorier les commandes de Systems Manager et leurs invocations, les entrées d'inventaire, décrire les informations sur les instances et les états d'exécution de l'automatisation, décrire les hôtes pour l'assistance au placement des instances et obtenir des informations détaillées sur l'invocation des commandes. Image Builder peut également envoyer des signaux d'automatisation et arrêter les exécutions automatisées pour toutes les ressources de votre compte.

  Image Builder est capable d'envoyer des appels de commande d'exécution à toute instance étiquetée `"CreatedBy": "EC2 Image Builder"` pour les fichiers de script suivants : `AWS-RunPowerShellScriptAWS-RunShellScript`, ou. `AWSEC2-RunSysprep` Image Builder est capable de lancer une exécution automatisée de Systems Manager dans votre compte pour les documents d'automatisation dont le nom commence par`ImageBuilder`.

Image Builder est également capable de créer ou de supprimer des associations State Manager pour n'importe quelle instance de votre compte, à condition que le document d'association le soit`AWS-GatherSoftwareInventory`, et de créer le rôle lié au service Systems Manager dans votre compte.

Image Builder est capable de lire les paramètres publics du magasin de paramètres, ainsi que de lire et de mettre à jour les paramètres privés préfixés par un préfixe `/imagebuilder/` afin de pouvoir mettre à jour la valeur du paramètre avec les ID d'AMI de sortie créés par Image Builder à partir d'une nouvelle version.

- AWS STS— L'accès est accordé à Image Builder pour assumer les rôles nommés EC2ImageBuilderDistributionCrossAccountRoledepuis votre compte sur n'importe quel compte lorsque la politique de confiance relative au rôle l'autorise. Ceci est utilisé pour la distribution d'images entre comptes.

Pour voir les autorisations de cette stratégie, consultez AWSServiceRoleForImageBuilder dans le AWS Guide de référence des stratégies gérées par.

EC2ImageBuilderExecutionPolicy policy

La EC2ImageBuilderExecutionPolicypolitique accorde des autorisations qui permettent à Image Builder d'appeler Services AWS en votre nom. Vous pouvez associer cette politique aux rôles IAM que vous transmettez à Image Builder en tant que rôle d'exécution au lieu d'utiliser le rôle lié à un service.

Détails de l'autorisation

La stratégie inclut les autorisations suivantes :

- CloudWatch Journaux — L'accès est accordé pour créer et télécharger CloudWatch des journaux dans tout groupe de journaux dont le nom commence par`/aws/imagebuilder/`.
- Amazon EC2 — L'accès est accordé à Image Builder pour créer, prendre des instantanés et enregistrer les images (AMI) qu'il crée, puis lancer des instances EC2 sur votre compte. Image Builder utilise les instantanés, les volumes, les interfaces réseau, les sous-réseaux, les groupes de sécurité, les configurations de licence et les paires de clés associés selon les besoins, à condition que l'image, l'instance et les volumes créés ou utilisés soient balisés avec `CreatedBy: EC2 Image Builder` ou. `CreatedBy: EC2 Fast Launch`

Image Builder peut obtenir des informations sur les images Amazon EC2, les attributs des instances, le statut des instances, les types d'instances disponibles pour votre compte, les modèles de lancement, les sous-réseaux, les hôtes et les balises de vos ressources Amazon EC2.

Image Builder peut mettre à jour les paramètres de l'image pour activer ou désactiver le lancement plus rapide des instances Windows dans votre compte, où l'image est étiquetée avec`CreatedBy: EC2 Image Builder`.

En outre, Image Builder peut démarrer, arrêter et mettre fin à des instances qui s'exécutent sur votre compte, partager des instantanés Amazon EBS, créer et mettre à jour des images, mettre à jour des modèles de lancement, annuler l'enregistrement d'images existantes, ajouter des balises et répliquer des images sur des comptes auxquels vous avez accordé des autorisations via la politique. Ec2ImageBuilderCrossAccountDistributionAccess Le balisage Image Builder est obligatoire pour toutes ces actions, comme décrit précédemment.

Image Builder peut également associer des filigranes d'AMI aux images balisées`CreatedBy: EC2 Image Builder`, afin que les filigranes se propagent vers des AMI dérivées.

- Amazon ECR — L'accès est accordé à Image Builder pour créer un référentiel si nécessaire pour les analyses de vulnérabilité des images de conteneurs, et pour étiqueter les ressources qu'il crée afin de limiter la portée de ses opérations. Image Builder est également autorisé à supprimer les images de conteneur qu'il a créées pour les scans après avoir pris des instantanés des vulnérabilités.
- EventBridge— L'accès est accordé à Image Builder pour créer et gérer EventBridge des règles.
- IAM — L'accès est accordé à Image Builder pour transmettre n'importe quel rôle de votre compte à Amazon EC2 et à VM. Import/Export
- Image Builder — L'accès est accordé à Image Builder pour déclencher automatiquement des pipelines au nom de l'utilisateur selon un calendrier défini par l'utilisateur et appliquer des balises aux images obtenues. Utilisé uniquement lorsque la `imageTags` propriété est définie sur un pipeline et que ce pipeline est configuré pour s'exécuter selon un calendrier.
- Amazon Inspector — L'accès est accordé à Image Builder pour déterminer quand Amazon Inspector termine les scans des instances de build et pour recueillir les résultats des images configurées pour l'autoriser.
- AWS KMS— L'accès est accordé à Amazon EBS pour chiffrer, déchiffrer ou rechiffrer les volumes Amazon EBS. Cela est essentiel pour garantir que les volumes chiffrés fonctionnent lorsque Image Builder crée une image.

- License Manager — L'accès est accordé à Image Builder pour mettre à jour les spécifications du License Manager via`license-manager:UpdateLicenseSpecificationsForResource`.
- Amazon SNS — Les autorisations d'écriture sont accordées pour toutes les rubriques Amazon SNS de votre compte.
- Systems Manager — L'accès est accordé à Image Builder pour répertorier les appels de commandes et les entrées d'inventaire de Systems Manager, décrire les informations sur les instances et les statuts d'exécution des associations, décrire les hôtes pour l'assistance au placement des instances et obtenir des informations détaillées sur les appels de commandes.

  Image Builder peut envoyer des appels de commande d'exécution à toute instance étiquetée `"CreatedBy": "EC2 Image Builder"` pour les fichiers de script suivants : `AWS-RunPowerShellScriptAWS-RunShellScript`, ou. `AWSEC2-RunSysprep`

  Image Builder peut également créer ou supprimer des associations State Manager pour toute instance de votre compte balisée`"CreatedBy": "EC2 Image Builder"`, à condition que le document d'association le soit`AWS-GatherSoftwareInventory`, et pour créer le rôle lié au service Systems Manager dans votre compte.

  Image Builder peut lire les paramètres publics du magasin de paramètres, ainsi que lire et mettre à jour les paramètres privés préfixés par un préfixe `/imagebuilder/` afin de pouvoir mettre à jour la valeur du paramètre avec les ID d'AMI de sortie créés par Image Builder à partir d'une nouvelle version.
- AWS STS— L'accès est accordé à Image Builder pour assumer les rôles nommés EC2ImageBuilderDistributionCrossAccountRoledepuis votre compte sur n'importe quel compte lorsque la politique de confiance relative au rôle l'autorise. Ceci est utilisé pour la distribution d'images entre comptes.

Pour voir les autorisations de cette stratégie, consultez EC2ImageBuilderExecutionPolicy dans le AWS Guide de référence des stratégies gérées par.

Ec2ImageBuilderCrossAccountDistributionAccess policy

La Ec2ImageBuilderCrossAccountDistributionAccesspolitique autorise Image Builder à distribuer des images entre les comptes des régions cibles. Image Builder peut également décrire, copier et appliquer des balises à n'importe quelle image Amazon EC2 du compte. La politique donne également la possibilité de modifier les autorisations de l'AMI via l'action de l'`ec2:ModifyImageAttribute`API.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes :

- Amazon EC2 — L'accès est accordé à Amazon EC2 pour décrire, copier et modifier les attributs d'une image, et pour créer des balises pour toutes les images Amazon EC2 présentes dans le compte.

Pour voir les autorisations de cette stratégie, consultez Ec2ImageBuilderCrossAccountDistributionAccess dans le AWS Guide de référence des stratégies gérées par.

EC2ImageBuilderLifecycleExecutionPolicy policy

La EC2ImageBuilderLifecycleExecutionPolicypolitique autorise Image Builder à effectuer des actions telles que la dépréciation, la désactivation ou la suppression des ressources d'image Image Builder et de leurs ressources sous-jacentes (AMI, instantanés) afin de prendre en charge les règles automatisées pour les tâches de gestion du cycle de vie des images.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes :

- Amazon EC2 — L'accès est accordé à Amazon EC2 pour effectuer les actions suivantes pour les Amazon Machine Images (AMI) du compte associé au tag. `CreatedBy: EC2 Image Builder`
  - Activez et désactivez une AMI.
  - Activez et désactivez la dépréciation des images.
  - Décrivez et annulez l'enregistrement d'une AMI.
  - Décrivez et modifiez les attributs d'image de l'AMI.
  - Supprimez les instantanés de volume associés à l'AMI.
  - Récupérez les balises d'une ressource.
  - Ajoutez ou supprimez des balises dans une AMI par souci de dépréciation.
- Amazon ECR — L'accès est accordé à Amazon ECR pour effectuer les actions par lots suivantes sur les référentiels ECR dotés de la balise. `LifecycleExecutionAccess: EC2 Image Builder` Les actions par lots prennent en charge les règles de cycle de vie automatisées des images des conteneurs.
  - `ecr:BatchGetImage`

- `ecr:BatchDeleteImage`

  L'accès est accordé au niveau du référentiel pour les référentiels ECR étiquetés avec. `LifecycleExecutionAccess: EC2 Image Builder`
- AWS Groupes de ressources — L'accès est accordé à Image Builder pour obtenir des ressources en fonction de balises.
- EC2 Image Builder — L'accès est accordé à Image Builder pour supprimer les ressources d'image Image Builder.

Pour voir les autorisations de cette stratégie, consultez EC2ImageBuilderLifecycleExecutionPolicy dans le AWS Guide de référence des stratégies gérées par.

EC2InstanceProfileForImageBuilder policy

La EC2InstanceProfileForImageBuilderpolitique accorde les autorisations minimales requises pour qu'une instance EC2 fonctionne avec Image Builder. Cela n'inclut pas les autorisations requises pour utiliser l'agent Systems Manager.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes :

- CloudWatch Journaux — L'accès est accordé pour créer et télécharger CloudWatch des journaux dans tout groupe de journaux dont le nom commence par`/aws/imagebuilder/`.
- Amazon EC2 — L'accès est accordé pour décrire les volumes et les instantanés, pour créer des instantanés des ressources de volume ou de capture d'écran créées par Image Builder et pour créer des balises pour les ressources Image Builder.
- Image Builder — L'accès est accordé pour obtenir n'importe quel générateur ou AWS Marketplace composant d'Image Builder.
- AWS KMS— L'accès est accordé pour déchiffrer un composant Image Builder, s'il a été chiffré via AWS KMS.
- Amazon S3 — L'accès est accordé pour obtenir des objets stockés dans un compartiment Amazon S3 dont le nom commence par`ec2imagebuilder-`, ou des ressources ayant une extension de fichier ISO.

Pour voir les autorisations de cette stratégie, consultez EC2InstanceProfileForImageBuilder dans le AWS Guide de référence des stratégies gérées par.

EC2InstanceProfileForImageBuilderECRContainerBuilds policy

La EC2InstanceProfileForImageBuilderECRContainerBuildspolitique accorde les autorisations minimales requises pour une instance EC2 lorsque vous travaillez avec Image Builder pour créer des images Docker, puis enregistrer et stocker les images dans un référentiel de conteneurs Amazon ECR. Cela n'inclut pas les autorisations requises pour utiliser l'agent Systems Manager.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes :

- CloudWatch Journaux — L'accès est accordé pour créer et télécharger CloudWatch des journaux dans tout groupe de journaux dont le nom commence par`/aws/imagebuilder/`.
- Amazon ECR — L'accès est accordé à Amazon ECR pour obtenir, enregistrer et stocker une image de conteneur, ainsi que pour obtenir un jeton d'autorisation.
- Image Builder — L'accès est accordé pour obtenir un composant Image Builder ou une recette de conteneur.
- AWS KMS— L'accès est accordé pour déchiffrer un composant ou une recette de conteneur Image Builder, s'il a été chiffré via AWS KMS.
- Amazon S3 — L'accès est accordé pour obtenir des objets stockés dans un compartiment Amazon S3 dont le nom commence par`ec2imagebuilder-`.

Pour voir les autorisations de cette stratégie, consultez EC2InstanceProfileForImageBuilderECRContainerBuilds dans le AWS Guide de référence des stratégies gérées par.

Mise à jour d'Image Builder vers AWS stratégies gérées

Cette section fournit des informations sur les mises à jour apportées aux politiques AWS gérées pour Image Builder depuis que ce service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques concernant les modifications apportées à cette page, abonnez-vous au flux RSS sur la page d'historique du document Image Builder.

Modifier	Description	Date
EC2ImageBuilderExecutionPolicy – Mise à jour d'une stratégie existante	Image Builder a été ajouté `ec2:AttachImageWatermark` pour permettre à Image Builder d'associer des filigranes AMI aux images de sortie `CreatedBy: EC2 Image Builder` balisées.	22 juin 2026
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a été ajouté `ec2:AttachImageWatermark` pour permettre à Image Builder d'associer des filigranes AMI aux images de sortie `CreatedBy: EC2 Image Builder` balisées.	22 juin 2026
EC2ImageBuilderExecutionPolicy : nouvelle politique	Image Builder a ajouté la nouvelle `EC2ImageBuilderExecutionPolicy` politique qui contient les autorisations permettant à Image Builder d'appeler Services AWS en votre nom. Vous pouvez associer cette politique aux rôles IAM que vous transmettez à Image Builder en tant que rôle d'exécution au lieu d'utiliser le rôle lié à un service.	4 juin 2026
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service afin de prendre en charge le balisage automatique des images de	18 mars 2026

Modifier	Description	Date
	sortie lorsque la `imageTags` propriété est définie sur un pipeline. • Ajouté `imagebuilder:StartImagePipelineExecution` pour permettre à Image Builder de déclencher des pipelines au nom de l'utilisateur selon un calendrier fourni par l'utilisateur avec les balises à appliquer. • Ajouté `imagebuilder:TagResource` pour permettre à Image Builder de baliser les images de sortie des pipelines au nom de l'utilisateur.	
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service : • `ssm:StartAutomationExecution` Supprimé, car le service a terminé sa migration hors de SSM Automation en 2023 et n'a plus besoin de cette autorisation.	26 février 2026

Modifier	Description	Date
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service afin de prendre en charge l'utilisation des paramètres de magasin de paramètres AWS Systems Manager (SSM) dans les recettes et lors de la distribution d'images. • Ajout de ssm : GetParameter pour permettre à Image Builder de lire les paramètres SSM publics et les paramètres SSM privés préfixés par un préfixe `/imagebuilder/` afin qu'ils puissent être utilisés dans des recettes. • Ajout de ssm : PutParameter pour permettre à Image Builder de mettre à jour les paramètres SSM privés préfixés par les ID `/imagebuilder/` d'AMI de sortie créés par Image Builder à partir d'une nouvelle version.	23 juillet 2025

Modifier	Description	Date
EC2InstanceProfileForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes à la politique de profil d'instance afin de prendre en charge davantage d'extensions de fichiers pour les téléchargements de fichiers ISO. • Deux extensions de fichier ont été ajoutées à la liste des ressources `s3:GetObject` d'action : `"arn:aws:s3:::*/*.iso"` et`"arn:aws:s3:::*/*.Iso"` .	19 mai 2025
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service afin de prendre en charge l'importation de fichiers ISO du système d'exploitation client Microsoft en tant qu'image de base. • Ajout de ec2 : RegisterImage pour permettre à Image Builder de créer un instantané et de créer et d'enregistrer une AMI dont le système d'exploitation de base a été importé à partir de fichiers de disque ISO vérifiés.	30 décembre 2024

Modifier	Description	Date
EC2InstanceProfileForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes à la politique de profil d'instance afin de prendre en charge la création d'images à partir de fichiers image disque. • Ajout des actions ec2 suivantes : DescribeVolumes et DescribeSnapshots sur toutes les ressources pour récupérer les détails. Les actions suivantes ont également été ajoutées pour les ressources créées par Image Builder : CreateSnapshot pour les ressources de volume et de capture d'écran, et CreateTags. Ajout de s3 : GetObject pour les ressources avec une extension de fichier « ISO ».	30 décembre 2024
EC2InstanceProfileForImageBuilder— Politique mise à jour	Image Builder a mis à jour la `EC2InstanceProfileForImageBuilder` politique pour permettre à Image Builder d'obtenir AWS Marketplace des composants.	2 décembre 2024

Modifier	Description	Date
EC2ImageBuilderLifecycleExecutionPolicy : nouvelle politique	Image Builder a ajouté la nouvelle `EC2ImageBuilderLifecycleExecutionPolicy` politique qui contient des autorisations pour la gestion du cycle de vie des images.	17 novembre 2023
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service afin de fournir une assistance au placement d'instances. • Ajout de ec2 : DescribeHosts permet à Image Builder d'interroger l'HostID afin de déterminer s'il est dans un état valide pour lancer une instance. • Ajout de l'action d'API ssm :GetCommandInvocation, pour améliorer la méthode utilisée par Image Builder pour obtenir les détails de l'appel de commande.	19 octobre 2023

Modifier	Description	Date
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service afin de fournir une assistance au placement d'instances. • Ajout d'ec2 : DescribeHosts permet à Image Builder d'interroger l'HostID afin de déterminer s'il est dans un état valide pour lancer une instance. • Ajout de l'action d'API ssm :GetCommandInvocation, pour améliorer la méthode utilisée par Image Builder pour obtenir les détails de l'appel de commande.	28 septembre 2023

Modifier	Description	Date
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service afin de permettre aux flux de travail Image Builder de collecter des informations sur les vulnérabilités pour les builds d'images de conteneur AMI et ECR. Les nouvelles autorisations prennent en charge la fonctionnalité de détection et de signalement des CVE. • Ajout de inspector2 : ListCoverage et inspector2 : pour permettre à ListFindings Image Builder de déterminer à quel moment Amazon Inspector effectue les scans des instances de test et de recueillir les résultats des images configurées pour l'autoriser. • Ajout de ecr :CreateRepository, avec l'obligation pour Image Builder de baliser le référentiel avec `CreatedBy: EC2 Image Builder` (tag-on-create). Nous avons également ajouté ecr : TagResource (obligatoire pour tag-on-create)	30 mars 2023

Modifier	Description	Date
	avec la même contrainte de CreatedBy balise, et une contrainte supplémentaire qui nécessite de commencer par le nom du dépôt. `image-builder-*` La contrainte de nom empêche l'augmentation des privilèges et empêche les modifications des référentiels qu'Image Builder n'a pas créés. • Ajout de ecr : BatchDeleteImage pour les référentiels ECR étiquetés avec. `CreatedBy: EC2 Image Builder` Cette autorisation nécessite de commencer par le nom du référentiel`image-builder-*` . • Ajout d'autorisations d'événement permettant à Image Builder de créer et de gérer les règles EventBridge gérées par Amazon qui figurent `ImageBuilder-*` dans le nom.	

Modifier	Description	Date
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service : • Ajout de licences License Manager en tant que ressource pour l'RunInstance appel ec2 : afin de permettre aux clients d'utiliser des AMI d'image de base associées à une configuration de licence.	22 mars 2022
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service : • Des autorisations ont été ajoutées pour EnableFastLaunch l'action de l'API EC2, afin d'activer et de désactiver le lancement plus rapide pour les instances Windows. • Champ d'application plus restreint pour ec2 : conditions des balises CreateTags d'action et de ressource.	21 février 2022

Modifier	Description	Date
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle de service : • Ajout d'autorisations permettant d'appeler le service VMIE pour importer une machine virtuelle et créer une AMI de base à partir de celle-ci. • Champ d'application renforcé pour ec2 : conditions CreateTags d'action et de balise de ressource.	20 novembre 2021
AWSServiceRoleForImageBuilder : mise à jour d'une politique existante	Image Builder a ajouté de nouvelles autorisations pour résoudre les problèmes liés au blocage de la création de l'image par plusieurs associations d'inventaire.	11 août 2021

Modifier	Description	Date
AWSImageBuilderFullAccess : mise à jour d'une politique existante	Image Builder a apporté les modifications suivantes au rôle d'accès complet : • Autorisations ajoutées à autoriser `ec2:DescribeInstanceTypeOfferings` . • Des autorisations d'appel ont été ajoutées `ec2:DescribeInstanceTypeOffereings` pour permettre à la console Image Builder de refléter avec précision les types d'instances disponibles dans le compte.	13 avril 2021
Image Builder a commencé à suivre les modifications	Image Builder a commencé à suivre les modifications apportées AWS à ses politiques gérées.	2 avril 2021

Utiliser des rôles liés à un service IAM pour Image Builder

EC2 Image Builder Gestion des identités et des accès AWS utilise des rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM directement lié à Image Builder. Les rôles liés au service sont prédéfinis par Image Builder et incluent toutes les autorisations dont le service a besoin pour appeler d'autres personnes en votre Services AWS nom.

Un rôle lié à un service rend la configuration d'Image Builder plus efficace, car il n'est pas nécessaire d'ajouter les autorisations nécessaires manuellement. Image Builder définit les autorisations associées à ses rôles liés aux services et, sauf indication contraire, seul Image Builder peut

assumer ses rôles. Les autorisations définies comprennent la stratégie d'approbation et la stratégie d'autorisations. La stratégie d'autorisations ne peut pas être attachée à une autre entité IAM.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS That Work with IAM et recherchez les services dont la valeur est Oui dans la colonne Rôle lié au service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations de rôle liées à un service pour Image Builder

Image Builder utilise le rôle AWSServiceRoleForImageBuilderlié au service pour permettre à EC2 Image Builder d'accéder aux AWS ressources en votre nom. Le rôle lié au service fait confiance au service imagebuilder.amazonaws.com pour assumer le rôle.

Vous n'avez pas besoin de créer manuellement ce rôle lié à un service. Lorsque vous créez votre première image Image Builder dans la console AWS de gestion AWS CLI, ou dans l' AWS API, Image Builder crée pour vous le rôle lié au service.

Les actions suivantes créent une nouvelle image :

- Exécutez l'assistant de pipeline dans la console Image Builder pour créer une image personnalisée.
- Utilisez l'une des actions d'API suivantes ou la AWS CLI commande correspondante :
  - L'action de l'CreateImageAPI (create-imagedans le AWS CLI).
  - L'action de l'ImportVmImageAPI (import-vm-imagedans le AWS CLI).
  - L'action de l'StartImagePipelineExecutionAPI (start-image-pipeline-executiondans le AWS CLI).

> **Important**
>
> Si le rôle lié au service est supprimé de votre compte, vous pouvez suivre le même processus pour le créer à nouveau. Lorsque vous créez votre première ressource EC2 Image Builder, Image Builder crée à nouveau le rôle lié au service pour vous.

Pour voir les autorisations pour le AWSServiceRoleForImageBuilder, consultez la AWSServiceRoleForImageBuilder policy page. Pour en savoir plus sur la configuration des autorisations pour un rôle lié à un service, consultez la section Autorisations relatives aux rôles liés à un service dans le guide de l'utilisateur IAM.

Supprimer un rôle lié au service Image Builder de votre compte

Vous pouvez utiliser la console IAM, le AWS CLI, ou l' AWS API pour supprimer manuellement le rôle lié à un service pour Image Builder de votre compte. Toutefois, avant de procéder, vous devez vous assurer qu'aucune ressource Image Builder n'y fait référence.

> **Note**
>
> Si le service Image Builder utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression risque d'échouer. Si cela se produit, patientez quelques minutes et réessayez.

Nettoyez les ressources Image Builder utilisées par le **`AWSServiceRoleForImageBuilder`** rôle

1. Vérifiez qu'aucune version de pipeline n'est en cours d'exécution avant de commencer. Pour annuler une compilation en cours, utilisez la `cancel-image-creation` commande du AWS CLI.

   ```
   aws imagebuilder cancel-image-creation --image-build-version-
   arn arn:aws:imagebuilder:us-east-1:123456789012:image-pipeline/sample-pipeline
   ```

2. Modifiez tous les plannings de pipeline pour utiliser un processus de génération manuel, ou supprimez-les si vous ne les réutilisez pas. Pour plus d'informations sur la suppression de ressources, consultezSupprimer les ressources d'Image Builder obsolètes ou inutilisées.

Supprimer le rôle lié à un service à l'aide d'IAM

Vous pouvez utiliser la console IAM, le AWS CLI, ou l' AWS API pour supprimer le `AWSServiceRoleForImageBuilder` rôle de votre compte. 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 au service Image Builder

Image Builder prend en charge l'utilisation de rôles liés à un service dans toutes les AWS régions où le service est disponible. Pour consulter la liste des AWS régions prises en charge, consultezAWS Régions et points de terminaison.

Résoudre les problèmes d'IAM dans Image Builder

Rubriques

- Je ne suis pas autorisé à effectuer une action dans Image Builder
- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite autoriser des personnes extérieures à moi Compte AWS à accéder à mes ressources Image Builder

Je ne suis pas autorisé à effectuer une action dans Image Builder

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 `imagebuilder:`*`GetWidget`* fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 imagebuilder: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 `imagebuilder:`*`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 un message d'erreur indiquant que vous n'êtes pas autorisé à effectuer l'`iam:PassRole`action, vos politiques doivent être mises à jour pour vous permettre de transmettre un rôle à Image Builder.

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, vous devez 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 Image Builder. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle de service. Mary n'est pas autorisée à transmettre le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d'exécuter l'action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je souhaite autoriser des personnes extérieures à moi Compte AWS à accéder à mes ressources Image Builder

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACLs), vous pouvez utiliser ces politiques pour autoriser les utilisateurs à accéder à vos ressources.

Pour plus d'informations, consultez les éléments suivants :

- Pour savoir si Image Builder prend en charge ces fonctionnalités, consultezComment Image Builder fonctionne avec les politiques et les rôles IAM.
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section Fournir un accès à des ressources Comptes AWS détenues par des tiers dans le guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.
- Pour en savoir plus sur la différence entre l'utilisation des rôles et des politiques basées sur les ressources pour l'accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Validation de conformité pour Image Builder

EC2 Image Builder ne fait l'objet d'aucun programme de AWS conformité.

Notre nouvelle expérience AWS d'inscription n'est pas conçue pour les charges de travail réglementées. Si vous utilisez notre nouvelle interface d' AWS inscription, mais que vous souhaitez l'utiliser AWS pour des charges de travail réglementées, vous pouvez vous inscrire à des fonctionnalités AWS (avancées) ou activer des fonctionnalités avancées pour votre AWS environnement.

Pour savoir si un s' Service AWS inscrit dans le cadre de programmes de conformité spécifiques, consultez Services AWS la section Étendue par programme de conformité et choisissez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes .

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 dépend de la sensibilité de vos données, des objectifs de conformité de votre entreprise et des lois et réglementations applicables. Pour plus d'informations sur votre responsabilité en matière de conformité lors de l'utilisation Services AWS, consultez AWS la documentation de sécurité.

Vous pouvez intégrer des produits de conformité provenant de AWS Task Orchestrator and Executor (AWSTOE) AWS Marketplace ou des composants provenant de () dans vos images Image Builder afin de garantir la conformité de vos images. Pour de plus amples informations, veuillez consulter Produits de conformité pour vos images Image Builder.

Redondance et résilience des données dans Image Builder

L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité. AWS Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, qui sont connectées à un réseau à faible latence, haut débit et hautement redondant. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone à l'autre sans interruption. Les zones de disponibilité sont davantage disponibles, tolérantes aux pannes et ont une plus grande capacité de mise à l'échelle que les infrastructures traditionnelles à un ou plusieurs centres de données.

Le service EC2 Image Builder vous permet de distribuer des images créées dans une région avec d'autres régions, ce qui leur confère une résilience multirégionale pour les AMI. Il n'existe aucun mécanisme permettant de « sauvegarder » les pipelines d'images, les recettes ou les composants. Vous pouvez stocker les documents de recette et de composants en dehors du service Image Builder, par exemple dans un compartiment Amazon S3.

L'EC2 Image Builder ne peut pas être configuré pour la haute disponibilité (HA). Vous pouvez distribuer des images dans plusieurs régions afin de les rendre plus disponibles.

Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS globale.

Sécurité de l'infrastructure dans Image Builder

Le réseau AWS mondial fournit des fonctionnalités de sécurité et contrôle l'accès au réseau pour des services tels que EC2 Image Builder. Pour plus d'informations sur la sécurité de l'infrastructure qui AWS assure ses services, consultez la section Sécurité de l'infrastructure du livre blanc Introduction à AWS la sécurité.

Pour envoyer des demandes via le réseau AWS mondial pour les actions de l'API Image Builder, votre logiciel client doit respecter les consignes de sécurité suivantes :

- Pour envoyer des demandes d'actions d'API Image Builder, le logiciel client doit utiliser une version prise en charge de Transport Layer Security (TLS).

  > **Note**
  >
  > AWS supprime progressivement la prise en charge des versions 1.0 et 1.1 de TLS. Nous vous recommandons vivement de mettre à jour votre logiciel client pour utiliser la version 1.2 ou ultérieure de TLS afin de pouvoir toujours vous connecter. Pour plus d'informations, consultez ce billet AWS de blog sur la sécurité.

- Le logiciel client doit prendre en charge les suites de chiffrement avec une confidentialité avancée parfaite (PFS), telles que Ephemeral (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Diffie-Hellman La plupart des systèmes actuels, tels que Java 7 et versions ultérieures, prennent en charge ces modes.
- Vous devez signer vos demandes d'API à l'aide d'un identifiant de clé d'accès et d'une clé d'accès secrète associée à un principal Gestion des identités et des accès AWS (IAM). Vous pouvez

également utiliser le AWS Security Token Service (AWS STS) pour générer des informations de sécurité temporaires pour vos demandes.

En outre, les instances EC2 utilisées par Image Builder pour créer et tester des images doivent avoir accès AWS Systems Manager.

Gestion des correctifs pour les images d'Image Builder

AWS fournit des AMI gérées mises à jour chaque mois qui intègrent les dernières mises à jour et correctifs de sécurité appliqués aux systèmes d'exploitation suivants. Vous pouvez utiliser ces AMI comme image de base pour vos personnalisations. Pour de plus amples informations, veuillez consulter Systèmes d'exploitation pris en charge.

- Distributions Linux, notamment Amazon Linux 2, AL2023, Red Hat Enterprise Linux (RHEL), CentOS, Ubuntu, SUSE Linux Enterprise Server
- Windows Server 2016 et versions ultérieures
- macOS 10.14.x et versions ultérieures

Après avoir créé une image personnalisée, vous êtes responsable de l'application des correctifs au système Amazon EC2, conformément au modèle de responsabilité partagée. Si les instances EC2 de la charge de travail de votre application peuvent être facilement remplacées, il peut être plus efficace de mettre à jour l'AMI de base et de redéployer tous les nœuds de calcul en fonction de cette image.

> **Note**
>
> Pour l'application de correctifs à macOS, nous vous recommandons de créer une nouvelle version de votre recette qui utilise la dernière AMI gérée pour l'image de base, puis de créer une image personnalisée mise à jour à partir de la recette et de vos autres ressources de création d'image. Si vos instances Mac ne sont pas faciles à remplacer, consultez la page Mettre à jour le système d'exploitation et les logiciels sur les instances Mac du guide de l'utilisateur Amazon EC2 pour plus d'informations.

Vous pouvez maintenir à jour vos AMI Image Builder de deux manières différentes.

- AWS-composants de correction fournis — EC2 Image Builder fournit les composants de compilation suivants qui installent toutes les mises à jour du système d'exploitation en attente :
  - `update-linux`
  - `update-windows`

  Ces composants utilisent le module `UpdateOS` d'action. Pour de plus amples informations, veuillez consulter Mettre à jour le système d'exploitation. Les composants peuvent être ajoutés à vos pipelines de création d'images en les sélectionnant dans la liste des composants AWS fournis.
- Composants de génération personnalisés avec opérations d'application de correctifs : pour installer ou mettre à jour des correctifs de manière sélective sur les systèmes d'exploitation des AMI prises en charge, vous pouvez créer un composant Image Builder pour installer les correctifs requis. Un composant personnalisé peut installer des correctifs à l'aide de scripts shell (Bash ou PowerShell), ou il peut utiliser le module `UpdateOS` d'action pour spécifier des correctifs à installer ou à exclure. Pour de plus amples informations, veuillez consulter Modules d'action pris en charge par AWSTOE gestionnaire de composants.

  Composant utilisant le module `UpdateOS` d'action (Linux et Windows uniquement). Le module `UpdateOS` d'action n'est pas pris en charge pour macOS.)

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: UpdateOS
  action: UpdateOS
```

  Composant qui utilise Bash pour installer les mises à jour de Yum

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: InstallYumUpdates
  action: ExecuteBash
  inputs:
    commands:
   - sudo yum update -y
```

Meilleures pratiques de sécurité pour Image Builder

EC2 Image Builder fournit un certain nombre de fonctionnalités de sécurité à prendre en compte lors de l'élaboration et de la mise en œuvre de vos propres politiques de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou suffisantes pour votre environnement, considérez-les comme des remarques utiles plutôt que comme des recommandations.

- N'utilisez pas de groupes de sécurité trop permissifs dans les recettes d'Image Builder.
- Ne partagez pas d'images avec des comptes auxquels vous n'avez pas confiance.
- Ne rendez pas publiques des images contenant des données privées ou sensibles.
- Appliquez tous les correctifs de sécurité Windows ou Linux disponibles lors de la création de l'image.
- Appliquez régulièrement des mises à jour d'AMI gérées à vos recettes macOS et créez de nouvelles images pour lancer des instances dotées des derniers correctifs de sécurité.
- Lorsque vous importez une image disque ISO Windows, procurez-vous l'ISO auprès de Microsoft ou d'un revendeur agréé. En outre, n'accordez au rôle de profil d'instance que les autorisations requises pour l'importation. Pour de plus amples informations, veuillez consulter Considérations de sécurité pour l'importation d'images de disque ISO.

Nous vous recommandons vivement de tester vos images pour valider la posture de sécurité et les niveaux de conformité de sécurité applicables. Des solutions telles qu'Amazon Inspector peuvent aider à valider le niveau de sécurité et de conformité des images.

Pipelines IMDSv2 pour Image Builder

Lorsque votre pipeline Image Builder s'exécute, il envoie des requêtes HTTP pour lancer des instances EC2 qu'Image Builder utilise pour créer et tester votre image. Pour configurer la version d'IMDS que votre pipeline utilise pour les demandes de lancement, définissez le `httpTokens` paramètre dans les paramètres de métadonnées de votre instance de configuration d'infrastructure Image Builder.

> **Note**
>
> Nous vous recommandons de configurer toutes les instances EC2 qu'Image Builder lance à partir d'une génération de pipeline pour utiliser IMDSv2 afin que les demandes de récupération des métadonnées d'instance nécessitent un en-tête de jeton signé.

Pour plus d'informations sur la configuration de l'infrastructure d'Image Builder, consultezGérer la configuration de l'infrastructure d'Image Builder. Pour plus d'informations sur les options de métadonnées d'instance EC2 pour les images Linux, voir Configurer les options de métadonnées d'instance dans le guide de l'utilisateur Amazon EC2. Pour les images Windows, voir Configurer les options de métadonnées de l'instance dans le guide de l'utilisateur Amazon EC2.

Nettoyage requis après la construction

Une fois qu'Image Builder a terminé toutes les étapes de création de votre image personnalisée, Image Builder prépare l'instance de génération pour les tests et la création d'images. Avant d'arrêter l'instance de génération pour créer l'instantané, Image Builder effectue le nettoyage suivant pour garantir la sécurité de votre image :

Linux

Le pipeline Image Builder exécute un script de nettoyage pour garantir que l'image finale respecte les meilleures pratiques de sécurité et pour supprimer tous les artefacts ou paramètres de génération qui ne devraient pas être transférés sur votre instantané. Vous pouvez toutefois ignorer des sections du script ou remplacer complètement les données utilisateur. Par conséquent, les images produites par les pipelines Image Builder ne sont pas nécessairement conformes à des critères réglementaires spécifiques.

Lorsque le pipeline a terminé ses étapes de construction et de test, Image Builder exécute automatiquement le script de nettoyage suivant juste avant de créer l'image de sortie.

> **Important**
>
> Si vous remplacez les données utilisateur dans votre recette, le script ne s'exécute pas. Dans ce cas, assurez-vous d'inclure une commande dans vos données utilisateur qui crée un fichier vide nomméperform_cleanup. Image Builder détecte ce fichier et exécute le script de nettoyage avant de créer la nouvelle image.

Une fois le nettoyage du fichier terminé, Image Builder ajoute des étapes au script pour désinstaller l'agent Systems Manager et supprimer le `cronie` package. Image Builder suit ce qu'il a installé lors de la génération dans le répertoire `/tmp/imagebuilder_service` de travail du service. Image Builder utilise ces marqueurs pour décider des éléments à supprimer :

- Agent Systems Manager : la désinstallation de l'agent par Image Builder dépend du `systemsManagerAgent.uninstallAfterBuild` paramètre défini dans votre recette d'image et de l'installation de l'agent par Image Builder. Pour plus d'informations, consultez le `uninstallAfterBuild` paramètre dansCréez une recette imagée à l'aide du AWS CLI.
- **`cronie`**package : cette étape s'applique à Amazon Linux 1, Amazon Linux 2 et Amazon Linux 2023. Si Image Builder a installé un crontab pendant la génération, il enregistre un `crontab_installed` marqueur dans le répertoire de travail du service. Image Builder supprime ensuite le `cronie` package dans le cadre du nettoyage. Pour conserver `cronie` votre image finale, supprimez le fichier de marqueurs avant le nettoyage en ajoutant la commande suivante à un composant ou à vos données utilisateur :

```
rm -f /tmp/imagebuilder_service/crontab_installed
```

```
#!/bin/bash
if [[ ! -f {{workingDirectory}}/perform_cleanup ]]; then
    echo "Skipping cleanup"
    exit 0
else
    sudo rm -f {{workingDirectory}}/perform_cleanup
fi

function cleanup() {
    FILES=("$@")
    for FILE in "${FILES[@]}"; do
        if [[ -f "$FILE" ]]; then
            echo "Deleting $FILE";
            sudo shred -zuf $FILE;
        fi;
        if [[ -f $FILE ]]; then
            echo "Failed to delete '$FILE'. Failing."
            exit 1
        fi;
    done
};
```

```
# Clean up for cloud-init files
CLOUD_INIT_FILES=(
    "/etc/sudoers.d/90-cloud-init-users"
    "/etc/locale.conf"
    "/var/log/cloud-init.log"
    "/var/log/cloud-init-output.log"
)
if [[ -f {{workingDirectory}}/skip_cleanup_cloudinit_files ]]; then
    echo "Skipping cleanup of cloud init files"
else
    echo "Cleaning up cloud init files"
    cleanup "${CLOUD_INIT_FILES[@]}"
    if [[ -d "/var/lib/cloud" ]]; then
        if [[ $( sudo find /var/lib/cloud -type f | sudo wc -l ) -gt 0 ]]; then
            echo "Deleting files within /var/lib/cloud/*"
            sudo find /var/lib/cloud -type f -exec shred -zuf {} \;
        fi;

        if [[ $( sudo ls /var/lib/cloud | sudo wc -l ) -gt 0 ]]; then
            echo "Deleting /var/lib/cloud/*"
            sudo rm -rf /var/lib/cloud/* || true
        fi;
    fi;
fi;


# Clean up for temporary instance files
INSTANCE_FILES=(
    "/etc/.updated"
    "/etc/aliases.db"
    "/etc/hostname"
    "/var/lib/misc/postfix.aliasesdb-stamp"
    "/var/lib/postfix/master.lock"
    "/var/spool/postfix/pid/master.pid"
    "/var/.updated"
    "/var/cache/yum/x86_64/2/.gpgkeyschecked.yum"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_files ]]; then
    echo "Skipping cleanup of instance files"
else
    echo "Cleaning up instance files"
    cleanup "${INSTANCE_FILES[@]}"
```

```
fi;


# Clean up for ssh files
SSH_FILES=(
    "/etc/ssh/ssh_host_rsa_key"
    "/etc/ssh/ssh_host_rsa_key.pub"
    "/etc/ssh/ssh_host_ecdsa_key"
    "/etc/ssh/ssh_host_ecdsa_key.pub"
    "/etc/ssh/ssh_host_ed25519_key"
    "/etc/ssh/ssh_host_ed25519_key.pub"
    "/root/.ssh/authorized_keys"
)
if [[ -f {{workingDirectory}}/skip_cleanup_ssh_files ]]; then
    echo "Skipping cleanup of ssh files"
else
    echo "Cleaning up ssh files"
    cleanup "${SSH_FILES[@]}"
    USERS=$(ls /home/)
    for user in $USERS; do
        echo Deleting /home/"$user"/.ssh/authorized_keys;
        sudo find /home/"$user"/.ssh/authorized_keys -type f -exec shred -zuf {} \;
    done
    for user in $USERS; do
        if [[ -f /home/"$user"/.ssh/authorized_keys ]]; then
            echo Failed to delete /home/"$user"/.ssh/authorized_keys;
            exit 1
        fi;
    done;
fi;


# Clean up for instance log files
INSTANCE_LOG_FILES=(
    "/var/log/audit/audit.log"
    "/var/log/boot.log"
    "/var/log/dmesg"
    "/var/log/cron"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_log_files ]]; then
    echo "Skipping cleanup of instance log files"
else
    echo "Cleaning up instance log files"
    cleanup "${INSTANCE_LOG_FILES[@]}"
```

```
fi;

# Clean up for TOE files
if [[ -f {{workingDirectory}}/skip_cleanup_toe_files ]]; then
    echo "Skipping cleanup of TOE files"
else
    echo "Cleaning TOE files"
    shopt -s nullglob
    TOE_MATCHES=({{workingDirectory}}/TOE_*)
    shopt -u nullglob
    if [[ ${#TOE_MATCHES[@]} -gt 0 ]]; then
        if [[ $( sudo find "${TOE_MATCHES[@]}" -type f | sudo wc -l) -gt 0 ]]; then
            echo "Deleting files within {{workingDirectory}}/TOE_*"
            sudo find "${TOE_MATCHES[@]}" -type f -exec shred -zuf {} \;
        fi
        shopt -s nullglob
        TOE_REMAINING=({{workingDirectory}}/TOE_*)
        shopt -u nullglob
        if [[ ${#TOE_REMAINING[@]} -gt 0 ]]; then
            if [[ $( sudo find "${TOE_REMAINING[@]}" -type f | sudo wc -l) -gt 0 ]];
 then
                echo "Failed to delete {{workingDirectory}}/TOE_*"
                exit 1
            fi
            echo "Deleting {{workingDirectory}}/TOE_*"
            sudo rm -rf "${TOE_REMAINING[@]}"
        fi
        shopt -s nullglob
        TOE_FINAL=({{workingDirectory}}/TOE_*)
        shopt -u nullglob
        if [[ ${#TOE_FINAL[@]} -gt 0 ]]; then
            echo "Failed to delete {{workingDirectory}}/TOE_*"
            exit 1
        fi
    fi
fi

# Clean up for ssm log files
if [[ -f {{workingDirectory}}/skip_cleanup_ssm_log_files ]]; then
    echo "Skipping cleanup of ssm log files"
else
    echo "Cleaning up ssm log files"
    if [[ -d "/var/log/amazon/ssm" ]]; then
        if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
```

```
            echo "Deleting files within /var/log/amazon/ssm/*"
            sudo find /var/log/amazon/ssm -type f -exec shred -zuf {} \;
        fi
        if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
            echo "Failed to delete /var/log/amazon/ssm"
            exit 1
        fi
        echo "Deleting /var/log/amazon/ssm/*"
        sudo rm -rf /var/log/amazon/ssm
        if [[ -d "/var/log/amazon/ssm" ]]; then
            echo "Failed to delete /var/log/amazon/ssm"
            exit 1
        fi
    fi
fi


shopt -s nullglob
SA_MATCHES=(/var/log/sa/sa*)
shopt -u nullglob
if [[ ${#SA_MATCHES[@]} -gt 0 ]]; then
    echo "Deleting /var/log/sa/sa*"
    sudo shred -zuf "${SA_MATCHES[@]}"
    shopt -s nullglob
    SA_REMAINING=(/var/log/sa/sa*)
    shopt -u nullglob
    if [[ ${#SA_REMAINING[@]} -gt 0 ]]; then
        echo "Failed to delete /var/log/sa/sa*"
        exit 1
    fi
fi

shopt -s nullglob
DHCLIENT_MATCHES=(/var/lib/dhclient/dhclient*.lease)
shopt -u nullglob
if [[ ${#DHCLIENT_MATCHES[@]} -gt 0 ]]; then
    echo "Deleting /var/lib/dhclient/dhclient*.lease"
    sudo shred -zuf "${DHCLIENT_MATCHES[@]}"
    shopt -s nullglob
    DHCLIENT_REMAINING=(/var/lib/dhclient/dhclient*.lease)
    shopt -u nullglob
    if [[ ${#DHCLIENT_REMAINING[@]} -gt 0 ]]; then
        echo "Failed to delete /var/lib/dhclient/dhclient*.lease"
        exit 1
```

```
    fi
fi

if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
    echo "Deleting files within /var/tmp/*"
    sudo find /var/tmp -type f -exec shred -zuf {} \;
fi
if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
    echo "Failed to delete /var/tmp"
    exit 1
fi
if [[ $( sudo ls /var/tmp | sudo wc -l ) -gt 0 ]]; then
    echo "Deleting /var/tmp/*"
    sudo rm -rf /var/tmp/*
fi

if [[ -f "/var/lib/systemd/random-seed" ]]; then
    echo "Deleting /var/lib/systemd/random-seed"
    sudo shred -zuf /var/lib/systemd/random-seed
    sudo rm -f /var/lib/systemd/random-seed
fi

# Shredding is not guaranteed to work well on rolling logs

if [[ -f "/var/lib/rsyslog/imjournal.state" ]]; then
    echo "Deleting /var/lib/rsyslog/imjournal.state"
    sudo shred -zuf /var/lib/rsyslog/imjournal.state
    sudo rm -f /var/lib/rsyslog/imjournal.state
fi

if [[ -d "/var/log/journal" ]] && [[ $( sudo ls /var/log/journal/ | sudo wc -l ) -gt
 0 ]]; then
    echo "Deleting /var/log/journal/*"
    sudo find /var/log/journal/ -type f -exec shred -zuf {} \;
    sudo rm -rf /var/log/journal/*
fi

if [[ -f "/etc/machine-id" ]]; then
    echo "Truncating /etc/machine-id"
    sudo truncate -s 0 /etc/machine-id
fi

if [[ -f "/var/lib/dbus/machine-id" ]]; then
    echo "Truncating /var/lib/dbus/machine-id"
```

```
    sudo truncate -s 0 /var/lib/dbus/machine-id
fi

sudo touch /etc/machine-id

# Flush all pending writes to disk before instance shutdown and snapshot
sync

echo "Sanitize OK"


################################################################################
# Image Builder appends the following steps to the clean up script to uninstall
# the Systems Manager (SSM) agent and, on Amazon Linux 1, Amazon Linux 2, and
# Amazon Linux 2023, to remove the cronie package when Image Builder installed a
# crontab during the build. Image Builder uses the /tmp/imagebuilder_service
# working directory to track what it installed.
################################################################################

SERVICE_ROOT_WORKING_DIR="/tmp/imagebuilder_service"

# SSM_UNINSTALL_CONDITION reflects the systemsManagerAgent.uninstallAfterBuild
# recipe setting and how the SSM agent was installed:
#   SSM_INSTALLED_BY_CUSTOMER      - always uninstall the SSM agent
#   SSM_INSTALLED_BY_IMAGE_BUILDER - uninstall only if Image Builder installed it
#   (any other value)              - leave the SSM agent in the final image
SSM_UNINSTALL_CONDITION="<uninstallAfterBuild condition>"

function error_exit {
  echo "$1" 1>&2
  exit 1
}

function ssm_exists() {
  eval "$1"  > /dev/null 2>&1
  echo $?
}

function cleanup_image() {
  rm -rf "${SERVICE_ROOT_WORKING_DIR}"
}

function uninstall_ssm_agent() {
```

```
 uninstall_package="$1"
 uninstall_all=""
 uninstall_success="false"

 if [ "${uninstall_package}" == "" ]; then
   uninstall_all="true"
 fi

 yum="sudo yum search amazon-ssm-agent | grep amazon-ssm-agent"
 snap="sudo snap list amazon-ssm-agent"
 rpm="sudo rpm -qa amazon-ssm-agent | grep amazon-ssm-agent"
 dpkg="sudo dpkg --get-selections | grep amazon-ssm-agent"
 pkg="su -m root -c \"pkg info -l amazon-ssm-agent | grep amazon-ssm-agent\""


 if [[ ("${uninstall_all}" == "true" || "${uninstall_package}" == "snap") &&
$(ssm_exists "${snap}") -eq 0 ]]; then
   echo "Package found in Snap.... Uninstalling"
   (sleep 30 ; sudo snap remove amazon-ssm-agent) &>/dev/null &
   uninstall_success="true"
 fi

 if [[ ("${uninstall_all}" == "true" || "${uninstall_package}" == "yum") &&
$(ssm_exists "${yum}") -eq 0 ]]; then
   echo "Package found in Yum.... Uninstalling"
   (sleep 30 ; sudo yum remove -y amazon-ssm-agent) &>/dev/null &
   uninstall_success="true"
 fi

 if [[ ("${uninstall_all}" == "true" || "${uninstall_package}" == "rpm") &&
$(ssm_exists "${rpm}") -eq 0 ]]; then
   echo "Package found in Rpm.... Uninstalling"
   (sleep 30 ; sudo rpm -e amazon-ssm-agent) &>/dev/null &
   uninstall_success="true"
 fi

 if [[ ("${uninstall_all}" == "true" || "${uninstall_package}" == "dpkg") &&
$(ssm_exists "${dpkg}") -eq 0 ]]; then
   echo "Package found in Dpkg.... Uninstalling"
   (sleep 30 ; sudo dpkg -r --force-all amazon-ssm-agent) &>/dev/null &
   uninstall_success="true"
 fi
```

```
  if [[ ("${uninstall_all}" == "true" || "${uninstall_package}" == "pkg") &&
 $(ssm_exists "${pkg}") -eq 0 ]]; then
    echo "Package found in FreeBSD.... Uninstalling"
    (sleep 30 ; su -m root -c "pkg remove -y amazon-ssm-agent") &> /dev/null &
    uninstall_success="true"
  fi

  if [ "${uninstall_success}" == "false" ] ; then
    error_exit "Unable to uninstall an SSM agent"
  fi
}

# Amazon Linux releases where Image Builder installs cronie to provide a crontab:
# Amazon Linux AMI (AL1), AL2, and AL2023.
function is_cronie_supported_amazon_linux() {
    if [ "$(get_os_type)" != "amzn" ]; then
        return 1
    fi
    case "$(get_os_version)" in
        2|2023) return 0 ;;
        # AL1 releases use date-based versions from 2010 through 2018.
        201[0-8].[0-9][0-9]) return 0 ;;
        *) return 1 ;;
    esac
}

function uninstall_crontab() {
    if is_cronie_supported_amazon_linux ; then
        echo "Uninstalling cronie package"
        sudo yum remove -y cronie
    fi
}

function get_os_type() {
    FILE=/etc/os-release
    if [ -e $FILE ]; then
        . $FILE
        echo $ID
    else
        echo ""
    fi
}

function get_os_version() {
```

```
    FILE=/etc/os-release
    if [ -e $FILE ]; then
        . $FILE
        echo $VERSION_ID
    else
        echo ""
    fi
}

if [ "${SSM_UNINSTALL_CONDITION}" == "SSM_INSTALLED_BY_CUSTOMER" ] ; then
  echo "Uninstall after build set to true. Uninstalling SSM agent."
  uninstall_ssm_agent

elif [ "${SSM_UNINSTALL_CONDITION}" == "SSM_INSTALLED_BY_IMAGE_BUILDER" ] ; then
  echo "Checking if the SSM agent was installed by Image Builder"
  if [[ -f ${SERVICE_ROOT_WORKING_DIR}/ssm_installed ]] ; then
    package_manager="$(cat ${SERVICE_ROOT_WORKING_DIR}/ssm_installed)"
    echo "Uninstalling the SSM agent installed by Image Builder using
 ${package_manager}"
    uninstall_ssm_agent "${package_manager}"
  fi
else
  echo "Uninstall after build set to false. Skipping SSM agent uninstall."
fi

# When Image Builder installs a crontab during the build (on Amazon Linux 1,
# Amazon Linux 2, or Amazon Linux 2023), it records a crontab_installed marker
# and removes the cronie package here. To keep cronie in your final image,
# delete the marker before clean up runs:
#   rm -f /tmp/imagebuilder_service/crontab_installed
if [[ -f ${SERVICE_ROOT_WORKING_DIR}/crontab_installed ]] ; then
  echo "Uninstalling crontab installed by Image Builder"
  uninstall_crontab
fi

cleanup_image
```

Windows

Une fois que le pipeline Image Builder a personnalisé les images Windows, il exécute l'utilitaire Microsoft Sysprep. Ces actions suivent les AWS meilleures pratiques pour durcir et nettoyer l'image .

macOS

Le pipeline Image Builder exécute un script de nettoyage pour garantir que l'image finale respecte les meilleures pratiques de sécurité et pour supprimer tous les artefacts ou paramètres de génération qui ne devraient pas être transférés sur votre instantané. Vous pouvez toutefois ignorer des sections du script ou remplacer complètement les données utilisateur. Par conséquent, les images produites par les pipelines Image Builder ne sont pas nécessairement conformes à des critères réglementaires spécifiques.

Lorsque le pipeline a terminé ses étapes de construction et de test, Image Builder exécute automatiquement le script de nettoyage suivant juste avant de créer l'image de sortie.

> **Important**
>
> Si vous remplacez les données utilisateur dans votre recette, le script ne s'exécute pas. Dans ce cas, assurez-vous d'inclure une commande dans vos données utilisateur qui crée un fichier vide nommé`perform_cleanup`. Image Builder détecte ce fichier et exécute le script de nettoyage avant de créer la nouvelle image.

```
#!/bin/bash
if [[ ! -f {{workingDirectory}}/perform_cleanup ]]; then
  echo "Skipping cleanup"
  exit 0
else
  sudo rm -f {{workingDirectory}}/perform_cleanup
fi

function cleanup() {
  FILES=("$@")
  for FILE in "${FILES[@]}"; do
      if [[ -f "$FILE" ]]; then
          echo "Deleting $FILE";
          sudo rm -f $FILE;
      fi;
      if [[ -f $FILE ]]; then
          echo "Failed to delete '$FILE'. Failing."
          exit 1
      fi;
  done
};
```

```
# Reset EC2 macOS Init instance history so the image behaves as a first boot
if [[ -f {{workingDirectory}}/skip_cleanup_ec2_macos_init_files ]]; then
  echo "Skipping cleanup of ec2-macos-init instance history"
else
  echo "Cleaning up ec2-macos-init instance history"
  if [[ -x /usr/local/bin/ec2-macos-init ]]; then
      sudo /usr/local/bin/ec2-macos-init clean -all
  fi
fi

# Clean up for temporary instance files
INSTANCE_FILES=(
  "/Library/Preferences/SystemConfiguration/NetworkInterfaces.plist"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_files ]]; then
  echo "Skipping cleanup of instance files"
else
  echo "Cleaning up instance files"
  cleanup "${INSTANCE_FILES[@]}"
fi;


# Clean up for ssh files
SSH_FILES=(
  "/etc/ssh/ssh_host_rsa_key"
  "/etc/ssh/ssh_host_rsa_key.pub"
  "/etc/ssh/ssh_host_ecdsa_key"
  "/etc/ssh/ssh_host_ecdsa_key.pub"
  "/etc/ssh/ssh_host_ed25519_key"
  "/etc/ssh/ssh_host_ed25519_key.pub"
  "/var/root/.ssh/authorized_keys"
)
if [[ -f {{workingDirectory}}/skip_cleanup_ssh_files ]]; then
  echo "Skipping cleanup of ssh files"
else
  echo "Cleaning up ssh files"
  cleanup "${SSH_FILES[@]}"
  USERS=$(ls /Users/)
  for user in $USERS; do
      if [[ -f /Users/"$user"/.ssh/authorized_keys ]]; then
          echo Deleting /Users/"$user"/.ssh/authorized_keys;
          sudo rm -f /Users/"$user"/.ssh/authorized_keys;
      fi;
```

```
  done
  for user in $USERS; do
      if [[ -f /Users/"$user"/.ssh/authorized_keys ]]; then
          echo Failed to delete /Users/"$user"/.ssh/authorized_keys;
          exit 1
      fi;
  done;
fi;


# Clean up for instance log files
INSTANCE_LOG_FILES=(
  "/var/log/amazon/ec2/ec2-macos-init.log"
  "/var/log/amazon/ec2/ena-ethernet.log"
  "/var/log/amazon/ec2/system-monitoring.log"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_log_files ]]; then
  echo "Skipping cleanup of instance log files"
else
  echo "Cleaning up instance log files"
  cleanup "${INSTANCE_LOG_FILES[@]}"
fi;

# Clean up for TOE files
if [[ -f {{workingDirectory}}/skip_cleanup_toe_files ]]; then
  echo "Skipping cleanup of TOE files"
else
  echo "Cleaning TOE files"
  shopt -s nullglob
  TOE_MATCHES=({{workingDirectory}}/TOE_*)
  shopt -u nullglob
  if [[ ${#TOE_MATCHES[@]} -gt 0 ]]; then
      if [[ $( sudo find "${TOE_MATCHES[@]}" -type f | sudo wc -l) -gt 0 ]]; then
          echo "Deleting files within {{workingDirectory}}/TOE_*"
          sudo find "${TOE_MATCHES[@]}" -type f -exec rm -f {} \;
      fi
      shopt -s nullglob
      TOE_REMAINING=({{workingDirectory}}/TOE_*)
      shopt -u nullglob
      if [[ ${#TOE_REMAINING[@]} -gt 0 ]]; then
          if [[ $( sudo find "${TOE_REMAINING[@]}" -type f | sudo wc -l) -gt 0 ]];
 then
              echo "Failed to delete {{workingDirectory}}/TOE_*"
              exit 1
```

```
          fi
          echo "Deleting {{workingDirectory}}/TOE_*"
          sudo rm -rf "${TOE_REMAINING[@]}"
      fi
      shopt -s nullglob
      TOE_FINAL=({{workingDirectory}}/TOE_*)
      shopt -u nullglob
      if [[ ${#TOE_FINAL[@]} -gt 0 ]]; then
          echo "Failed to delete {{workingDirectory}}/TOE_*"
          exit 1
      fi
  fi
fi

# Clean up for ssm log files
if [[ -f {{workingDirectory}}/skip_cleanup_ssm_log_files ]]; then
  echo "Skipping cleanup of ssm log files"
else
  echo "Cleaning up ssm log files"
  if [[ -d "/var/log/amazon/ssm" ]]; then
      if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
          echo "Deleting files within /var/log/amazon/ssm/*"
          sudo find /var/log/amazon/ssm -type f -exec rm -f {} \;
      fi
      if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
          echo "Failed to delete /var/log/amazon/ssm"
          exit 1
      fi
      echo "Deleting /var/log/amazon/ssm/*"
      sudo rm -rf /var/log/amazon/ssm
      if [[ -d "/var/log/amazon/ssm" ]]; then
          echo "Failed to delete /var/log/amazon/ssm"
          exit 1
      fi
  fi
fi


# Clean up for DHCP lease files
shopt -s nullglob
DHCP_LEASE_MATCHES=(/var/db/dhcpclient/leases/*)
shopt -u nullglob
if [[ ${#DHCP_LEASE_MATCHES[@]} -gt 0 ]]; then
  echo "Deleting /var/db/dhcpclient/leases/*"
```

```
  sudo rm -f "${DHCP_LEASE_MATCHES[@]}"
  shopt -s nullglob
  DHCP_LEASE_REMAINING=(/var/db/dhcpclient/leases/*)
  shopt -u nullglob
  if [[ ${#DHCP_LEASE_REMAINING[@]} -gt 0 ]]; then
      echo "Failed to delete /var/db/dhcpclient/leases/*"
      exit 1
  fi
fi

if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
      echo "Deleting files within /var/tmp/*"
      sudo find /var/tmp -type f -exec rm -f {} \;
fi
if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
      echo "Failed to delete /var/tmp"
      exit 1
fi
if [[ $( sudo ls /var/tmp | sudo wc -l ) -gt 0 ]]; then
      echo "Deleting /var/tmp/*"
      sudo rm -rf /var/tmp/*
fi

# Flush all pending writes to disk before instance shutdown and snapshot
sync
```

Remplacez le script de nettoyage de Linux

Image Builder crée des images sécurisées par défaut et conformes à nos meilleures pratiques de sécurité. Toutefois, dans certains cas d'utilisation plus avancés, vous devrez peut-être ignorer une ou plusieurs sections du script de nettoyage intégré. Si vous devez ignorer une partie du nettoyage, nous vous recommandons vivement de tester votre AMI de sortie pour garantir la sécurité de votre image.

> **Important**
>
> Le fait de sauter des sections dans le script de nettoyage peut entraîner l'inclusion d'informations sensibles, telles que les détails du compte du propriétaire ou les clés SSH, dans l'image finale et, dans tous les cas, lancées à partir de cette image. Vous

pouvez également rencontrer des problèmes lors du lancement dans différentes zones de disponibilité, régions ou comptes.

Le tableau suivant décrit les sections du script de nettoyage, les fichiers supprimés dans cette section et les noms de fichiers que vous pouvez utiliser pour signaler une section qu'Image Builder doit ignorer. Pour ignorer une section spécifique du script de nettoyage, vous pouvez utiliser le module d'action du CreateFile composant ou une commande de vos données utilisateur (en cas de remplacement) pour créer un fichier vide avec le nom spécifié dans la colonne Nom de fichier de la section Ignorer.

> **Note**
>
> Les fichiers que vous créez pour ignorer une section du script de nettoyage ne doivent pas inclure d'extension de fichier. Par exemple, si vous souhaitez ignorer la `CLOUD_INIT_FILES` section du script, mais que vous créez un fichier nommé`skip_cleanup_cloudinit_files.txt`, Image Builder ne reconnaîtra pas le fichier ignoré.

Input

Section de nettoyage	Fichiers supprimés	Ignorer le nom du fichier de section

`CLOUD_INIT_FILES`	`/etc/sudoers.d/90-cloud-init-users`
	`/etc/locale.conf`
	`/var/log/cloud-init.log`
	`/var/log/cloud-init-output.log`
	Tous les fichiers de `/var/lib/cloud/`
		`skip_cleanup_cloudinit_files`

Section de nettoyage	Fichiers supprimés	Ignorer le nom du fichier de section
INSTANCE_FILES	/etc/.updated /etc/aliases.db /etc/hostname /var/lib/misc/postfix.aliasesdb-stamp /var/lib/postfix/master.lock /var/spool/postfix/pid/master.pid /var/.updated /var/cache/yum/x86_64/2/.gpgkeyschecked.yum	skip_cleanup_instance_files

Section de nettoyage	Fichiers supprimés	Ignorer le nom du fichier de section
SSH_FILES	/etc/ssh/ssh_host_rsa_key /etc/ssh/ssh_host_rsa_key.pub /etc/ssh/ssh_host_ecdsa_key /etc/ssh/ssh_host_ecdsa_key.pub /etc/ssh/ssh_host_ed25519_key /etc/ssh/ssh_host_ed25519_key.pub /root/.ssh/authorized_keys /home/<all users>/.ssh/authorized_keys	skip_cleanup_ssh_files
INSTANCE_LOG_FILES	/var/log/audit/audit.log /var/log/boot.log /var/log/dmesg /var/log/cron	skip_cleanup_instance_log_files
TOE_FILES	{{workingDirectory}}/TOE_*	skip_cleanup_toe_files

Section de nettoyage	Fichiers supprimés	Ignorer le nom du fichier de section
SSM_LOG_FILES	/var/log/amazon/ssm/*	skip_cleanup_ssm_log_files

Étapes de nettoyage qui fonctionnent toujours

Les sections du tableau précédent sont les seules parties du script de nettoyage que vous pouvez ignorer. Une fois que le script a traité ces sections, il supprime également les éléments suivants, et vous ne pouvez pas ignorer ces étapes :

- /var/log/sa/sa*
- /var/lib/dhclient/dhclient*.lease
- Tous les fichiers de /var/tmp/
- /var/lib/systemd/random-seed
- /var/lib/rsyslog/imjournal.state
- Tous les fichiers de /var/log/journal/

Le script tronque également les fichiers /etc/machine-id d'identification de la machine et. /var/lib/dbus/machine-id Cela garantit que chaque instance lancée à partir de l'image génère un identifiant de machine unique.

Ignorer le script de nettoyage de macOS

Image Builder crée des images sécurisées par défaut et conformes à nos meilleures pratiques de sécurité. Toutefois, dans certains cas d'utilisation plus avancés, vous devrez peut-être ignorer une ou plusieurs sections du script de nettoyage intégré. Si vous devez ignorer une partie du nettoyage, nous vous recommandons vivement de tester votre AMI de sortie pour garantir la sécurité de votre image.

Important

Le fait de sauter des sections dans le script de nettoyage peut entraîner l'inclusion d'informations sensibles, telles que les détails du compte du propriétaire ou les clés SSH, dans l'image finale et, dans tous les cas, lancées à partir de cette image. Vous pouvez également rencontrer des problèmes lors du lancement dans différentes zones de disponibilité, régions ou comptes.

Le tableau suivant décrit les sections du script de nettoyage, les fichiers supprimés dans cette section et les noms de fichiers que vous pouvez utiliser pour signaler une section qu'Image Builder doit ignorer. Pour ignorer une section spécifique du script de nettoyage, vous pouvez utiliser le module d'action du CreateFile composant ou une commande de vos données utilisateur (en cas de remplacement) pour créer un fichier vide avec le nom spécifié dans la colonne Nom de fichier de la section Ignorer.

Note

Les fichiers que vous créez pour ignorer une section du script de nettoyage ne doivent pas inclure d'extension de fichier. Par exemple, si vous souhaitez ignorer la `INSTANCE_FILES` section du script, mais que vous créez un fichier nommé`skip_cleanup_instance_files.txt`, Image Builder ne reconnaîtra pas le fichier ignoré.

Input

Section de nettoyage	Fichiers supprimés	Ignorer le nom du fichier de section
`EC2_MACOS_INIT_FILES`	Exécute `ec2-macos-init clean -all` pour réinitialiser l'historique de l'instance EC2 macOS Init afin que l'image se comporte comme au premier démarrage.	`skip_cleanup_ec2_macos_init_files`

Section de nettoyage	Fichiers supprimés	Ignorer le nom du fichier de section
INSTANCE_FILES	/Library/Preferences/SystemConfiguration/NetworkInterfaces.plist	skip_cleanup_instance_files
SSH_FILES	/etc/ssh/ssh_host_rsa_key /etc/ssh/ssh_host_rsa_key.pub /etc/ssh/ssh_host_ecdsa_key /etc/ssh/ssh_host_ecdsa_key.pub /etc/ssh/ssh_host_ed25519_key /etc/ssh/ssh_host_ed25519_key.pub /var/root/.ssh/authorized_keys /Users/<all users>/.ssh/authorized_keys	skip_cleanup_ssh_files

Section de nettoyage	Fichiers supprimés	Ignorer le nom du fichier de section
INSTANCE_LOG_FILES	/var/log/amazon/ec2/ec2-macos-init.log /var/log/amazon/ec2/ena-ethernet.log /var/log/amazon/ec2/system-monitoring.log	skip_cleanup_instance_log_files
TOE_FILES	{{workingDirectory}}/TOE_*	skip_cleanup_toe_files
SSM_LOG_FILES	/var/log/amazon/ssm/*	skip_cleanup_ssm_log_files

Résoudre les problèmes liés à Image Builder

EC2 Image Builder s'intègre à des Services AWS fins de surveillance et de dépannage afin de vous aider à résoudre les problèmes liés à la création d'images. Image Builder suit et affiche la progression de chaque étape du processus de création d'images. En outre, Image Builder peut exporter des journaux vers un emplacement Amazon S3 que vous fournissez.

Pour un dépannage avancé, vous pouvez exécuter des commandes et des scripts prédéfinis à l'aide de l'option AWS Systems Manager Exécuter la commande.

> **Tip**
>
> Vous pouvez utiliser la fonction de résolution des problèmes, optimisée par AWS DevOps l'Agent, pour diagnostiquer les versions ayant échoué et obtenir une analyse des AI-powered causes premières avec des étapes de correction. Pour de plus amples informations, veuillez consulter Résoudre les problèmes liés aux builds ayant échoué avec l'IA - Aperçu.

Table des matières

- Résoudre les problèmes liés aux builds de pipelines
- Scénarios de résolution des problèmes
- Résoudre les problèmes liés aux builds ayant échoué avec l'IA - Aperçu
- Résoudre les problèmes liés aux builds à l'aide d'un agent de codage IA

Résoudre les problèmes liés aux builds de pipelines

Si la création d'un pipeline Image Builder échoue, Image Builder renvoie un message d'erreur décrivant l'échec. Image Builder renvoie également un `workflow execution ID` message d'échec, tel que celui de l'exemple de sortie suivant :

```
Workflow Execution ID: wf-12345abc-6789-0123-abc4-567890123abc failed with reason: …
```

Image Builder organise et dirige les actions de création d'images selon une série d'étapes définies pour les étapes d'exécution de son processus de création d'image standard. Les étapes de création et de test du processus sont chacune associées à un flux de travail. Lorsqu'Image Builder exécute un

flux de travail pour créer ou tester une nouvelle image, il génère une ressource de métadonnées de flux de travail qui enregistre les détails d'exécution.

Les images de conteneurs comportent un flux de travail supplémentaire qui s'exécute pendant la distribution.

Recherchez des informations sur les défaillances des instances d'exécution pour votre flux de travail

Pour résoudre un échec d'exécution de votre flux de travail, vous pouvez appeler les actions d'ListWorkflowStepExecutionsAPI GetWorkflowExecution et avec votre`workflow execution ID`.

Consulter les journaux d'exécution des flux de travail

- CloudWatch Journaux Amazon

  Image Builder publie des journaux d'exécution de flux de travail détaillés dans le groupe et le flux Image Builder CloudWatch Logs suivants :

  Avec CloudWatch Logs, vous pouvez rechercher des données de journal à l'aide de modèles de filtre. Pour plus d'informations, consultez la section Rechercher des données de journal à l'aide de modèles de filtre dans le guide de l'utilisateur Amazon CloudWatch Logs.
- AWS CloudTrail

  Toutes les activités de création sont également enregistrées CloudTrail si elles sont activées dans votre compte. Vous pouvez filtrer CloudTrail les événements en fonction de leur source`imagebuilder.amazonaws.com`. Vous pouvez également rechercher l'ID d'instance Amazon EC2 renvoyé dans le journal d'exécution pour obtenir plus de détails sur l'exécution du pipeline.
- Amazon Simple Storage Service (S3)

  Si vous avez spécifié un nom de compartiment S3 et un préfixe de clé dans la configuration de votre infrastructure, le chemin du journal d'exécution des étapes du workflow suit le modèle suivant :

  ```
  S3://S3BucketName/KeyPrefix/ImageName/ImageVersion/ImageBuildVersion/
  WorkflowExecutionId/StepName
  ```

  Les journaux que vous envoyez à votre compartiment S3 indiquent les étapes et les messages d'erreur relatifs à l'activité sur l'instance EC2 pendant le processus de création de l'image. Les

journaux incluent les sorties de journal du gestionnaire de composants, les définitions des composants qui ont été exécutés et la sortie détaillée (en JSON) de toutes les étapes effectuées sur l'instance. Si vous rencontrez un problème, vous devez passer en revue ces fichiers, en commençant par`application.log`, pour diagnostiquer la cause du problème sur l'instance.

Par défaut, Image Builder arrête l'instance de compilation ou de test Amazon EC2 qui est en cours d'exécution lorsque le pipeline échoue. Vous pouvez modifier les paramètres d'instance pour la ressource de configuration d'infrastructure utilisée par votre pipeline, afin de conserver votre instance de génération ou de test à des fins de dépannage.

Pour modifier les paramètres de l'instance dans la console, vous devez désactiver la case à cocher Terminer l'instance en cas de panne située dans la section Paramètres de dépannage de votre ressource de configuration d'infrastructure.

Vous pouvez également modifier les paramètres de l'instance à l'aide de la update-infrastructure-configuration commande du AWS CLI. Définissez la `terminateInstanceOnFailure` valeur `false` dans le fichier JSON auquel la commande fait référence avec le `--cli-input-json` paramètre. Pour en savoir plus, consultez Mettre à jour une configuration d'infrastructure.

> **Tip**
>
> Pour plus d'informations sur un exemple qui déploie une version qui échoue intentionnellement, avec une procédure pas à pas qui montre où chaque classe de défaillance laisse des traces, consultez l'exemple de débogage sur. GitHub Pour de plus amples informations, veuillez consulter Découvrez des exemples de projets Image Builder sur GitHub.

Scénarios de résolution des problèmes

Cette section répertorie les scénarios de dépannage détaillés suivants :

- Accès refusé — code d'état 403
- Expire les délais de génération lors de la vérification de la disponibilité de l'agent Systems Manager sur l'instance de génération
- Le disque secondaire Windows est hors ligne au lancement
- La compilation échoue avec l'image de base renforcée CIS

- AssertInventoryCollection échoue (Systems Manager Automation)

Pour voir les détails d'un scénario, choisissez le titre du scénario pour le développer. Vous pouvez étendre plusieurs titres en même temps.

Accès refusé — code d'état 403

Description

La compilation du pipeline échoue avec « AccessDenied : Code d'état Accès refusé : 403 ».

Cause

Les causes possibles incluent :

- Le profil d'instance ne dispose pas des autorisations requises pour accéder aux API ou aux ressources des composants.
- Le rôle de profil d'instance ne dispose pas des autorisations requises pour la connexion à Amazon S3. Cela se produit le plus souvent lorsque le rôle de profil d'instance ne dispose pas PutObject d'autorisations pour vos compartiments S3.

Solution

Selon la cause, ce problème peut être résolu comme suit :

- Le profil d'instance ne dispose pas de politiques gérées : ajoutez les politiques manquantes à votre rôle de profil d'instance. Ensuite, relancez le pipeline.
- Le profil d'instance ne dispose pas d'autorisations d'écriture pour le compartiment S3 : ajoutez une politique à votre rôle de profil d'instance qui accorde PutObject des autorisations d'écriture dans votre compartiment S3. Ensuite, relancez le pipeline.

Expire les délais de génération lors de la vérification de la disponibilité de l'agent Systems Manager sur l'instance de génération

Description

La génération du pipeline échoue avec « status = 'TimedOut' » et « message d'échec = 'L'étape a expiré alors que l'étape vérifie la disponibilité de l'agent Systems Manager sur la ou les instances cibles '».

Cause

Les causes possibles incluent :

- L'instance qui a été lancée pour effectuer les opérations de génération et exécuter des composants n'a pas pu accéder au point de terminaison Systems Manager.
- Le profil d'instance ne dispose pas des autorisations requises.

Solution

En fonction de la cause possible, ce problème peut être résolu comme suit :

- Problème d'accès, sous-réseau privé — Si vous créez dans un sous-réseau privé, assurez-vous d'avoir configuré des PrivateLink points de terminaison pour Systems Manager, Image Builder et, si vous souhaitez enregistrer, Amazon. S3/CloudWatch Pour plus d'informations sur la configuration des PrivateLink terminaux, voir Accéder aux AWS services via AWS PrivateLink.
- Autorisations manquantes : ajoutez les politiques gérées suivantes à votre rôle lié au service IAM pour Image Builder :
  - EC2InstanceProfileForImageBuilder
  - EC2InstanceProfileForImageBuilderECRContainerBuilds
  - AmazonSSMManagedInstanceCore

  Pour plus d'informations sur le rôle lié au service Image Builder, consultez. Utiliser des rôles liés à un service IAM pour Image Builder

Le disque secondaire Windows est hors ligne au lancement

Description

Lorsque le type d'instance utilisé pour créer une AMI Windows Image Builder ne correspond pas au type d'instance utilisé pour le lancement à partir de l'AMI, un problème peut survenir lorsque des volumes non root sont hors ligne au lancement. Cela se produit principalement lorsque l'instance de génération utilise une architecture plus récente que l'instance de lancement.

L'exemple suivant montre ce qui se passe lorsqu'une AMI Image Builder est créée sur un type d'instance EC2 Nitro et lancée sur une instance EC2 Xen :

Type d'instance de compilation : m5.large (Nitro)

Type d'instance de lancement : t2.medium (Xen)

```
PS C:\Users\Administrator> get-disk
Number  Friendly Name  Serial Number         Health Status  Operational Status  Total
 Size  Partition Style
------  -------------  -------------         -------------  ------------------
 ----------  ---------------
0       AWS PVDISK     vol0abc12d34e567f8a9  Healthy        Online                  30
 GB  MBR
1       AWS PVDISK     vol1bcd23e45f678a9b0  Healthy        Offline                  8
 GB  MBR
```

Cause

En raison des paramètres par défaut de Windows, les disques récemment découverts ne sont pas automatiquement mis en ligne et formatés. Lorsque le type d'instance est modifié sur EC2, Windows le considère comme de nouveaux disques découverts. Cela est dû au changement de moteur sous-jacent.

Solution

Nous vous recommandons d'utiliser le même système de types d'instances lors de la création de l'AMI Windows à partir de laquelle vous souhaitez le lancer. N'incluez pas les types d'instances créés sur différents systèmes dans la configuration de votre infrastructure. Si l'un des types d'instances que vous spécifiez utilise le système Nitro, ils doivent tous utiliser le système Nitro.

Pour plus d'informations sur les instances créées sur le système Nitro, consultez la section Instances créées sur le système Nitro dans le guide de l'utilisateur Amazon EC2.

La compilation échoue avec l'image de base renforcée CIS

Description

Vous utilisez une image de base renforcée CIS et la compilation échoue.

Cause

Lorsque le `/tmp` répertoire est classé comme tel`noexec`, cela peut entraîner l'échec d'Image Builder.

Solution

Choisissez un autre emplacement pour votre répertoire de travail dans le `workingDirectory` champ de la recette d'image. Pour plus d'informations, consultez la description du type de ImageRecipe données.

AssertInventoryCollection échoue (Systems Manager Automation)

Description

Systems Manager Automation indique un échec lors de l'étape `AssertInventoryCollection` d'automatisation.

Cause

Vous ou votre organisation avez peut-être créé une association Systems Manager State Manager qui collecte des informations d'inventaire pour les instances EC2. Si la collecte améliorée de métadonnées d'image est activée pour votre pipeline Image Builder (c'est la valeur par défaut), Image Builder tente de créer une nouvelle association d'inventaire pour l'instance de génération. Toutefois, Systems Manager n'autorise pas plusieurs associations d'inventaire pour les instances gérées et empêche toute nouvelle association s'il en existe déjà une. Cela entraîne l'échec de l'opération et entraîne l'échec de la construction du pipeline.

Solution

Pour résoudre ce problème, désactivez la collecte améliorée de métadonnées d'image à l'aide de l'une des méthodes suivantes :

- Mettez à jour votre pipeline d'images dans la console pour désactiver la case à cocher Activer la collecte améliorée de métadonnées. Enregistrez vos modifications et lancez une génération de pipeline.

  Pour plus d'informations sur la mise à jour de votre pipeline d'images AMI à l'aide de la console EC2 Image Builder, consultezMettre à jour les pipelines d'images AMI depuis la console. Pour plus d'informations sur la mise à jour de votre pipeline d'images de conteneur à l'aide de la console EC2 Image Builder, consultezMettre à jour un pipeline d'images de conteneur depuis la console.
- Vous pouvez également mettre à jour votre pipeline d'images à l'aide de la update-image-pipeline commande du AWS CLI. Pour ce faire, incluez la `EnhancedImageMetadataEnabled` propriété dans votre fichier JSON, en lui attribuant la valeur`false`. L'exemple suivant montre la propriété définie sur`false`.

```
{
    "name": "MyWindows2019Pipeline",
    "description": "Builds Windows 2019 Images",
    "enhancedImageMetadataEnabled": false,
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-infrastructure-configuration",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-distribution-configuration",
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 60
    },
    "schedule": {
        "scheduleExpression": "cron(0 0 * * SUN *)",
        "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "ENABLED"
}
```

Pour éviter que cela ne se produise pour les nouveaux pipelines, désactivez la case à cocher Activer la collecte améliorée de métadonnées lorsque vous créez un nouveau pipeline à l'aide de la console EC2 Image Builder, ou définissez la valeur de la `EnhancedImageMetadataEnabled` propriété dans votre fichier JSON sur `false` lorsque vous créez votre pipeline à l'aide du AWS CLI.

Résoudre les problèmes liés aux builds ayant échoué avec l'IA - Aperçu

⚠ Fonctionnalité de prévisualisation

Cette fonctionnalité est en version préliminaire et est sujette à modification.

Lorsque les builds d'EC2 Image Builder échouent, la cause première peut être longue et complexe à identifier. La fonction de résolution des problèmes d'Image Builder, optimisée par AWS DevOps Agent, automatise ce processus grâce AI-powered à l'analyse. Vous pouvez rapidement identifier la cause d'échecs de construction complexes ou détecter des modèles sur plusieurs pipelines défaillants.

Lorsqu'une compilation échoue, vous pouvez utiliser Troubleshoot pour analyser les journaux de génération, les CloudWatch journaux Amazon, les exécutions des étapes du flux de travail et les AWS ressources associées. Vous recevez une analyse de la cause première et les étapes à suivre pour résoudre le problème.

Vous pouvez résoudre les problèmes liés à l'échec de la construction d'un seul pipeline ou analyser plusieurs pipelines défaillants pour identifier des modèles communs et des causes profondes communes.

Table des matières

- Conditions préalables
- Configurer la résolution des problèmes
- Résoudre les problèmes liés à une compilation qui a échoué
- Résoudre les problèmes liés à plusieurs pipelines défaillants
- Ce que Troubleshoot étudie
- Considérations

Conditions préalables

Avant d'utiliser la fonctionnalité de résolution des problèmes, assurez-vous que l'utilisateur ou le rôle IAM que vous utilisez dispose des autorisations suivantes :

- `aidevops:ListAgentSpaces`— Nécessaire pour découvrir les places d'agent disponibles.
- `aidevops:CreateChat`— Nécessaire pour démarrer les sessions de dépannage.
- `aidevops:SendChatMessage`— Obligatoire pour envoyer des messages lors du dépannage.
- `aidevops:CreateAgentSpace`— Obligatoire uniquement si vous souhaitez qu'Image Builder crée un espace d'agent pour vous. Vous pouvez omettre cette autorisation si vous apportez votre propre espace d'agent.

Pour plus d'informations sur les politiques IAM, consultez la section Politiques et autorisations dans IAM.

Configurer la résolution des problèmes

La fonctionnalité de résolution des problèmes nécessite un espace AWS DevOps agent. Vous pouvez utiliser un espace d'agent existant ou demander à Image Builder d'en créer un pour vous :

- Apportez votre propre espace d'agent — Si vous disposez déjà d'un espace d'agent, Image Builder l'utilise pour résoudre les problèmes.
- Demandez à Image Builder d'en créer un — Si vous ne disposez pas d'un espace d'agent, Image Builder en crée un et le configure lorsque vous commencez votre première enquête. Cette option nécessite l'`aidevops:CreateAgentSpace`autorisation.

Pour commencer à résoudre les problèmes, cliquez sur l'icône Résolution des problèmes dans l' Console de gestion AWS en-tête.

> **Note**
>
> Pour plus d'informations sur AWS DevOps l'agent, consultez le guide de l'utilisateur de l'AWS DevOpsagent.

Résoudre les problèmes liés à une compilation qui a échoué

Lorsqu'une création d'image échoue, vous pouvez lancer une AI-powered enquête à partir de plusieurs points d'entrée dans l'Image Builder Console de gestion AWS.

Lancer une enquête de dépannage

Utilisez l'une des méthodes suivantes pour commencer à résoudre les problèmes liés à l'échec de la compilation :

- Tableau Pipelines : cliquez sur l'icône de dépannage à côté de l'état d'un pipeline défaillant dans la liste Image des pipelines.
- Tableau des images de sortie — Choisissez l'icône de résolution des problèmes à côté d'une image ayant échoué dans la liste des images de sortie.

- Tiroir de résolution des problèmes : dans l' Console de gestion AWS en-tête, cliquez sur l'icône de résolution des problèmes, puis sélectionnez le pipeline que vous souhaitez examiner.

Au cours de l'enquête

Une fois que vous avez commencé à résoudre les problèmes, Troubleshoot étudie la panne. Vous pouvez suivre les progrès en temps réel dans le tiroir au fur et à mesure que les journaux et les ressources sont analysés. L'enquête se termine généralement en quelques minutes. Vous pouvez continuer à travailler dans d'autres Console de gestion AWS domaines pendant que l'analyse est en cours. À la fin de l'enquête, vous recevez :

- Analyse des causes premières qui identifie les raisons de l'échec de la compilation
- Étapes de correction pour résoudre le problème
- Commandes ou modifications de configuration pour résoudre le problème

Après avoir examiné les résultats, vous pouvez fournir des commentaires sur la qualité de la réponse. Pour réexécuter l'analyse sur la même version qui a échoué, choisissez à nouveau Résoudre les problèmes.

Résoudre les problèmes liés à plusieurs pipelines défaillants

Si vous avez plusieurs pipelines défaillants, vous pouvez les analyser ensemble pour identifier des modèles communs et des causes profondes communes.

Lancer une enquête globale

Utilisez l'une des méthodes suivantes pour dépanner plusieurs pipelines :

- Dans le tableau Pipelines d'image, sélectionnez plusieurs pipelines défaillants et choisissez Résoudre les problèmes sélectionnés.
- Ouvrez le tiroir de résolution des problèmes et choisissez Résoudre tous les problèmes pour analyser tous les pipelines défaillants.

Résultats d'analyse agrégés

Vous recevez une analyse groupée organisée par cause première. Pour chaque groupe, vous obtenez :

- Description de la cause première partagée
- Oléoducs concernés par cette cause fondamentale
- Étapes de remédiation pour le groupe

Après l'analyse globale, vous pouvez choisir une invite d'exploration pour étudier plus en détail un pipeline spécifique.

Ce que Troubleshoot étudie

Lorsque vous dépannez une compilation qui a échoué, l'analyse couvre les domaines suivants :

- Créez des journaux et des CloudWatch journaux Amazon — Vérifie les erreurs, les codes de sortie différents de zéro, les délais d'attente et les traces de pile.
- Exécutions des étapes du flux de travail : identifie l'étape qui a échoué et examine ses détails.
- Réseau et autorisations : examine les autorisations des profils d'instance IAM, les groupes de sécurité, le routage des sous-réseaux, les passerelles NAT, les passerelles Internet et les points de terminaison VPC.
- Configuration des composants et des recettes : vérifie les erreurs de script, les dépendances manquantes, les URL cassées et la validité de base de l'AMI.
- Connectivité réseau : vérifie l'accessibilité de l'agent AWS Systems Manager (Systems Manager), la résolution DNS et la configuration du proxy.
- Historique des versions : passe en revue les versions récentes pour distinguer les défaillances persistantes des problèmes intermittents.

Considérations

Lorsque vous utilisez la fonction de résolution des problèmes, tenez compte des points suivants :

- Troubleshoot effectue une analyse en lecture seule. Vos ressources ne sont pas modifiées.
- Le dépannage est limité aux défaillances de génération d'Image Builder et à l'infrastructure associée, telle qu'IAM, VPC et Systems Manager Agent.

> **Note**
>
> L'IA génère ces réponses. Vérifiez l'analyse et les recommandations avant de modifier vos ressources.

Pour plus d'informations sur la résolution des problèmes liés à la génération de pipelines d'Image Builder, consultezRésoudre les problèmes liés aux builds de pipelines.

Résoudre les problèmes liés aux builds à l'aide d'un agent de codage IA

L'Agent Toolkit for AWS fournit aux agents de codage IA les outils, les connaissances et les garanties dont ils ont besoin pour travailler avec AWS les services. Il fonctionne avec des agents tels que Kiro, Claude Code, Cursor et Codex. La boîte à outils comprend les `amazon-ec2-image-builder` compétences, un ensemble d'instructions et des documents de référence qui guident votre agent dans les tâches d'Image Builder.

Grâce à ces compétences, votre agent peut diagnostiquer un échec de compilation à partir de votre terminal ou de votre IDE. La compétence dirige l'agent vers l'étape du flux de travail qui a échoué et vers les journaux des composants dans Amazon CloudWatch Logs. Il fournit des étapes de remédiation pour les scénarios de défaillance courants tels que les délais de connectivité des instances, les autorisations de profil d'instance manquantes et les erreurs de composants.

La compétence couvre le reste du flux de travail d'Image Builder : le rôle IAM de création, les composants gérés et personnalisés par Amazon, les recettes d'images et la configuration de l'infrastructure et de la distribution. Il gère les versions ponctuelles et les pipelines récurrents pour une automatisation optimale des AMI et l'application de correctifs au système d'exploitation. Il prend en charge les AMI Linux, Windows et macOS, ainsi que les images de conteneurs.

> **Note**
>
> La compétence s'exécute dans votre propre agent de codage IA. Pour une analyse des causes AI-powered profondes intégrée à la console Image Builder, consultezRésoudre les problèmes liés aux builds ayant échoué avec l'IA - Aperçu.

Table des matières

- Acquérez la compétence
- Considérations

Acquérez la compétence

Votre agent peut acquérir cette compétence des manières suivantes :

- Découverte au moment de l'exécution via le serveur AWS MCP — Les agents connectés au serveur AWS MCP peuvent rechercher et charger la compétence à la demande, sans installation locale. Pour plus d'informations, consultez la section Compétences du guide de l' AWS utilisateur de l'Agent Toolkit.
- Installation à l'aide de AWS CLI — La commande suivante installe la compétence pour les agents de codage IA détectés sur votre système. La commande spécifie la `us-east-1` région car le catalogue de compétences n'est disponible que dans cette région. Cela ne limite pas les domaines dans lesquels vous pouvez utiliser la compétence. Pour les exigences de configuration et de version, consultez la AWS CLI dans le Guide de l' AWS utilisateur de l'Agent Toolkit.

```
aws agent-toolkit add-skill \
    --skill-name amazon-ec2-image-builder \
    --region us-east-1
```

- Télécharger depuis GitHub — La source de compétences est disponible dans l'Agent Toolkit pour le AWS référentiel sur GitHub. Ajoutez-le au répertoire des compétences de votre agent.

Considérations

Notez les points suivants lorsque vous utilisez cette compétence :

- Votre agent utilise vos AWS informations d'identification pour appeler Image Builder et les services associés. Nous vous recommandons de limiter les autorisations IAM de l'agent au minimum requis par la tâche.
- L'IA génère les réponses. Vérifiez l'analyse et les recommandations avant de modifier vos ressources.

Historique de la documentation des modifications apportées au guide de l'utilisateur d'Image Builder

Le tableau suivant décrit les modifications importantes apportées à la documentation par date. Pour recevoir les notifications de mise à jour de cette documentation, abonnez-vous à un flux RSS.

- Version de l'API : 30/04/2025

Modification	Description	Date
Consignes de sécurité pour l'importation d'images disque ISO	Il a été démontré que les exécutables résolus à partir de l'image ISO fournie et de l'image Windows appliquée s'exécutent avec les privilèges SYSTEM sur l'instance de génération, sous le rôle de profil d'instance défini dans la configuration de l'infrastructure. Il est de votre responsabilité de vérifier l'authenticité de l'ISO. Pour plus d'informations, consultez les rubriques Considérations relatives à la sécurité lors de l'importation d'images disque ISO et Votre responsabilité en matière d'images disque ISO importées.	4 septembre 2026
Exemples de projets sur GitHub	Ajout d'une page pour le référentiel d'exemples d'Image Builder sur GitHub. Le référentiel contient des exemples de projets complets	1er septembre 2026

	dans AWS CloudFormation AWS CDK, et Terraform, ainsi que des documents de AWSTOE composants. Des liens vers des échantill ons individuels provenant de pages connexes ont également été ajoutés. Pour plus d'informations, consultez les exemples de projets Explore Image Builder sur GitHub.	
Trousse à outils pour les agents en AWS matière	Cette `amazon-ec2-image-builder` compétence a été documentée dans l'Agent Toolkit for AWS, qui aide les agents de codage IA à diagnostiquer les versions ayant échoué et à créer des images avec Image Builder. Pour plus d'informations, consultez Résoudre les problèmes liés aux builds à l'aide d'un agent de codage IA.	1er septembre 2026

Contenu AWS Marketplace d'intégration étendu	Documentation expliquant comment spécifier un identifiant de AWS Marketplace produit comme image parent dans la create-image-recipe commande et comment inspecter les codes de produit sur une AMI de base. Pour plus d'informations, consultez la section AWS Marketplace Intégration dans Image Builder.	11 août 2026
Opt-in Prérequis régional pour les paramètres de sortie SSM	Ajout de la région opt-in (région désactivée par défaut) préalable pour les distributions AMI qui incluent des paramètres de sortie SSM. Pour plus d'informations, consultez la section Conditions requises pour les paramètres de sortie SSM.	31 juillet 2026

Post-build mises à jour des scripts de nettoyage pour Linux	Mise à jour du script de nettoyage post-compilation de Linux Required pour afficher les étapes de désinstallation de l'agent Systems Manager. Le script supprime également le `cronie` package lorsqu'Im age Builder a installé un crontab pendant la générati on. Cela s'applique à Amazon Linux 1, Amazon Linux 2 et Amazon Linux 2023. A expliqué comment conserver l'image finale `cronie` en supprimant le fichier de `crontab_installed` marqueurs avant les opération s de nettoyage. Mise à jour également du script de nettoyage Override the Linux pour répertorier les `/var/lib/cloud/` fichiers supprimés par la `CLOUD_INI T_FILES` section. Ajout d'une note décrivant les étapes de nettoyage qui s'exécutent toujours et ne peuvent pas être ignorées.	27 juillet 2026

Contenu d' EventBridge intégration Amazon étendu	Réorganisation de la page du EventBridge pipeline pour couvrir à la fois le démarrage des builds à partir de règles et la réaction aux événement s envoyés par Image Builder. Ajout d'exemples et de AWS CLI procédures de règles relatives aux modèles d'événements, documentation des champs et des condition s préalables pour chaque événement, et clarification de la différence entre un calendrie r de pipeline et une EventBrid ge règle de planification. Pour plus d'informations, voir Utilisation EventBridge avec les pipelines Image Builder.	21 juillet 2026
Politiques gérées mises à jour pour les filigranes AMI	Image Builder a ajouté l'`ec2:AttachImageWat ermark` autorisation aux politiques AWSServic eRoleForImageBuilder et les EC2ImageBuilderExecutionPol icy a gérées afin qu'Image Builder puisse associer des filigranes AMI aux AMI de sortie qu'il crée. Pour plus d'informations, consultez les politiques AWS gérées pour Image Builder.	22 juin 2026

Mise à jour de la fonctionnalité : filigranes AMI	Image Builder prend en charge les filigranes AMI pour le suivi du lignage des images. Vous pouvez spécifier des noms de filigranes dans votre recette d'image, et Image Builder les associe automatiquement aux AMI de sortie pendant le processus de génération. Les filigranes se propagent automatiquement lorsque vous copiez ou distribuez des AMI entre les régions ou les comptes. Pour plus d'informations, voir Suivre le lignage des AMI à l'aide de filigranes.	22 juin 2026
Correction du script de nettoyage après la construction de macOS	Correction du script de nettoyage post-construction de macOS Required, qui contenait auparavant de Linux-specific la logique. Le script s'exécute désormais`ec2-macos-init clean --all`, supprime les clés SSH macOS de `/Users` et `/var/root` et utilise les chemins de fichiers macOS. Ajout de la référence au script de nettoyage Override the macOS.	12 juin 2026

Publication de la politique IAM : politique d'exécution	Image Builder a ajouté la nouvelle politique `EC2ImageBuilderExecutionPolicy` AWS gérée. Vous pouvez associer cette politique à un rôle IAM personnalisé que vous transmettez à Image Builder en tant que rôle d'exécution, au lieu de transmettre le rôle lié à un service. Pour plus d'informations, consultez la EC2ImageBuilderExecutionPolicy politique.	4 juin 2026
Ajout du support pour Windows Server 2025	Ajout de la prise en charge des versions STIG de Windows Server 2025 et application des STIG pour la version du premier trimestre 2026 pour tous les niveaux de conformité (low/medium/high).	10 mars 2026
Mise à jour de la politique IAM : politique sur les rôles de service	Supprimé `ssm:StartAutomationExecution` de la politique relative aux rôles liés aux services, le service ayant terminé sa migration hors de SSM Automation en 2023. Pour plus d'informations, consultez la AWSServiceRoleForImageBuilder politique.	26 février 2026

Version de fonctionnalité : améliorations de la politique de cycle de vie	Les améliorations apportées aux politiques de cycle de vie incluent une gestion rationalisée des rôles avec la création de rôles sur console à l'aide des paramètres par défaut des services, la prise en charge des versions sémantiques génériques pour les politiques basées sur des recettes (1.0.x, 1.x.x, x.x.x) afin de cibler plusieurs versions de recettes avec une seule politique, la collecte de ressources basée sur des balises pour les politiques de cycle de vie et la prise en charge par console des règles d'exclusion sur les politiques basées sur les AMI et les politiques basées sur des balises.	26 février 2026
Améliorations de distribution d'images	Les améliorations incluent la possibilité d'effectuer directement des activités de distribution sans qu'il soit nécessaire de réexécuter une création d'image complète.	20 novembre 2025

Auto-versioning et améliorations apportées à IaC	L'incrémentation automatique des versions et le référencement des versions par des caractères génériques éliminent désormais la gestion manuelle des versions pour les recettes, les composants et les flux de travail. Expérience de test de composants avec exécution à sec et fonctionnalité de création améliorée dans la console.	20 novembre 2025
Le flux de travail d'images prend en charge les fonctions Lambda et Step	Le flux de travail d'image prend désormais en charge l'appel de fonctions Lambda et l'exécution de la machine à états Step Functions, permettant ainsi des flux de travail complexes en plusieurs étapes et une logique de validation personnalisée lors de la création d'images.	13 novembre 2025

Améliorations du pipeline d'images	Les améliorations incluent la prise en charge des journaux d'exécution des pipelines, CloudWatch des groupes de journaux configurables et la configuration permettant de désactiver automatiquement les pipelines planifiés qui échouent à plusieurs reprises. Les mises à jour supplémentaires incluent des informations détaillées sur le calendrier du pipeline et des exemples de messages d'événements CloudWatch Logs mis à jour.	29 septembre 2025
Nouveau script pour Amazon Linux 2023	Ajout du support pour le système d'exploitation SUSE Linux Enterprise Server (SLES). Versions STIG et STIGS appliquées pour la version du troisième trimestre 2025 pour tous les niveaux de conformité (low/medium/élevé).	4 septembre 2025
Nouveau script pour Amazon Linux 2023	Ajout d'un nouveau script spécialement adapté à Amazon Linux 2023.	26 août 2025

Mise à jour de la politique IAM : politique sur les rôles de service	Mise à jour de la politique relative aux rôles liés aux services afin de prendre en charge l'utilisation des paramètres SSM dans les recettes et lors de la distribut ion d'images. Pour plus d'informations, consultez la AWSServiceRoleForI mageBuilder politique.	23 juillet 2025
Mises à jour du STIG Q2 2025	Versions STIG de Windows mises à jour et STIGS appliqué pour la version du deuxième trimestre 2025.	26 juin 2025
Mise à jour de la politique IAM : politique de profil d'instance	Ajoutez `.iso` et utilisez `.Iso` des extensions de fichier valides pour télécharger des images de disque ISO depuis Amazon S3. Pour plus d'informations, consultez la EC2InstanceProfileForImageB uilder politique.	19 mai 2025
Mises à jour du STIG Q1 2025	Versions STIG de Windows mises à jour et versions des composants STIGS et SCAP appliquées pour la version du premier trimestre 2025.	5 mai 2025

Version de fonctionnalité : prise en charge des paramètres SSM	Image Builder prend désormais en charge l'utilisation des paramètres de stockage des paramètres AWS Systems Manager(SSM) dans les recettes et lors de la distribution d'images.	30 avril 2025
Mises à jour du STIG Q4	Versions STIG de Windows mises à jour et STIGS appliqué pour la version du quatrième trimestre 2024.	4 février 2025
Mise à jour de la politique IAM : politique sur les rôles de service	Mise à jour de la politique relative aux rôles liés aux services pour prendre en charge la création d'images à partir de fichiers ISO officiels importés du système d'exploitation client Microsoft. Pour plus d'informations, consultez la AWSServiceRoleForImageBuilder politique.	30 décembre 2024
Mise à jour de la politique IAM : politique de profil d'instance	Mise à jour de la politique de rôle du profil d'instance pour prendre en charge la création d'images à partir de fichiers d'image disque. Pour plus d'informations, consultez la EC2InstanceProfileForImageBuilder politique.	30 décembre 2024

Version de fonctionnalité : ISO vers AMI	Image Builder peut désormais créer une image à partir de fichiers de disque ISO officiels pour les systèmes d'exploitation clients Microsoft Windows 11 et versions ultérieures.	30 décembre 2024
Mises à jour du STIG Q4	Versions Linux STIG mises à jour et STIGS appliqué pour la version du quatrième trimestre 2024. Ajout d'informations sur deux nouveaux paramètres d'entrée pour les composants Linux.	10 décembre 2024
Mise à jour de la politique IAM : politique de profil d'instance	Mise à jour de la politique de profil d'instance pour autoriser l'accès aux AWS Marketplace composants GET. Pour plus d'informations, consultez la EC2InstanceProfileForImageBuilder politique.	2 décembre 2024
Version de fonctionnalité : AWS Marketplace composants	Ajout du support pour les composants AWS Marketplace logiciels.	2 décembre 2024
Mise à jour des fonctionnalités : prise en charge de macOS	Ajout du support pour les images macOS.	22 octobre 2024
Support documentaire pour les opérateurs logiques dans AWS Task Orchestrator and Executor	Utilisez des opérateurs logiques pour ajouter ou modifier des expressions conditionnelles dans votre document de composants.	16 août 2024

Support documentaire pour les constructions conditionnelles dans AWS Task Orchestrator and Executor	Utilisez des constructions conditionnelles telles que l'instruction « if » pour diriger le flux de contrôle des actions des composants dans votre document de composants.	16 août 2024
Support documentaire pour les opérateurs de comparaison dans AWS Task Orchestrator and Executor	Utilisez des opérateurs de comparaison pour comparer les valeurs de votre document de composants.	16 août 2024
Ajout du module d'action Assert	Ajout de la documentation pour le module `Assert` d'action ci-dessous`General execution` .	14 août 2024
Support du système d'exploitation	Ajout du support pour RHEL 9 et Ubuntu 24.04 LTS.	2 août 2024
Mise à jour du document : réorganiser le contenu	Réorganisation de la documentation pour améliorer la présentation et la navigation.	21 juin 2024
Mises à jour du STIG Q2	Versions Linux STIG mises à jour et STIGS appliqué pour la version du deuxième trimestre 2024. Ajout du support pour RHEL 9, CentOS Stream 9 et Ubuntu 22.04. Aucune modification n'a été apportée aux versions de Windows.	10 mai 2024

Mises à jour du STIG Q1	Versions Linux STIG mises à jour et STIGS appliqué pour la version du premier trimestre 2024. Aucune modification n'a été apportée aux versions de Windows.	23 février 2024
Mises à jour du STIG Q1	Versions Linux STIG mises à jour et STIGS appliqué pour la version du premier trimestre 2024. Aucune modification n'a été apportée aux versions de Windows.	23 février 2024
Version de fonctionnalité : gestion du flux d'images	Grâce aux flux de travail d'images, vous bénéficiez d'une flexibilité, d'une visibilité et d'un contrôle accrus sur le processus de création d'images. Vous pouvez personnaliser les étapes de création et de test pour vos flux de travail, ou vous pouvez utiliser le flux de travail par défaut d'Image Builder.	12 décembre 2023
Mises à jour du STIG Q4	Versions Linux STIG mises à jour et STIGS appliqué pour la version du quatrième trimestre 2023. Aucune modification n'a été apportée aux versions de Windows. Linux et Windows SCAP ont également été mis à jour pour les nouveaux composants, les nouveaux logiciels et les numéros de référence.	7 décembre 2023

Version de fonctionnalité : gestion du cycle de vie des images	Grâce aux règles et politiques de gestion du cycle de vie des images, vous pouvez définir votre stratégie de gestion des ressources afin de garantir que les images obsolètes et les ressources associées soient soumises à un processus de balisage et de suppression.	17 novembre 2023
Mise à jour de la politique IAM : rôle de service	Mise à jour de la politique relative aux rôles liés aux services, par exemple pour l'aide au placement. Pour plus d'informations, consultez la AWSServiceRoleForImageBuilder politique.	19 octobre 2023
Mises à jour du STIG Q3	Versions STIG mises à jour et STIGS appliqué pour la version du troisième trimestre 2023. La messagerie a également été mise à jour pour préciser que les packages tiers ne sont pas automatiquement installés, à de très rares exceptions près. Tous les STIG ignorés sont enregistrés.	5 octobre 2023
Nouvelles versions de STIG	Versions STIG mises à jour et STIGS appliqué pour la version du deuxième trimestre 2023.	3 mai 2023

Nouvelles versions de STIG	Versions STIG mises à jour et STIGS appliqué pour la version du premier trimestre 2023. Ajout du support pour AL2023.	14 avril 2023
Mettre à jour les régions prises en charge pour AWSTOE	Ajout du AWSTOE support pour les pays suivants Régions AWS : Asie-Pacifique (Hyderabad), Asie-Pacifique (Jakarta), Europe (Zurich), Europe (Espagne) et Moyen-Orient (Émirats arabes unis).	13 avril 2023
AWSTOE mises à jour des téléchargements d'applications	Mise à jour de la signature pour le téléchargement de AWSTOE l'installation sous Windows. Également mis à jour, notez que les télécharg ements d'applications à partir de compartiments S3 nécessitent désormais la version 1.2 ou ultérieure de TLS.	31 mars 2023
Version de fonctionnalité : flux de travail de création améliorés	Ajout de détails d'exécution pour les créations d'images dans le nouvel onglet de flux de travail dans les détails de la version de création d'image. Informations améliorées pour résoudre les problèmes liés aux builds.	30 mars 2023

Mise à jour de la fonctionnalité : détection et signalement des CVE	Pour les comptes qui ont activé les scans Amazon Inspector, Image Builder peut capturer les résultats courants relatifs aux vulnérabilités et aux expositions (CVE) d'Amazon Inspector lors de la phase de test du processus de création de nouvelles images, y compris les images de conteneur stockées dans Amazon ECR. Image Builder crée un instantané des résultats pour faciliter l'analyse détaillée. Image Builder fournit également des rapports sur le nombre de résultats qui peuvent être filtrés par compte, par pipeline ou par image, avec la possibilité d'approfondir les détails.	30 mars 2023
Historique des versions ajouté	Ajout de l'historique des versions aux sections Windows et Linux.	17 février 2023
Nouvelles versions de STIG	Versions STIG mises à jour et STIGS appliqué pour la version du quatrième trimestre 2022.	1er février 2023

Mise à jour des fonctionnalités : AWS Marketplace intégration et renforcement du CIS	Ajout AWS Marketplace d'une intégration permettant de trouver et d'utiliser facilement une image souscrite comme référence pour une nouvelle image personnalisée, y compris les images renforcées CIS et un nouveau composant CIS Hardening du Center for Internet Security.	13 janvier 2023
composants de durcissement CIS	Ajout de composants de durcissement CIS détenus et entretenus par CIS.	13 janvier 2023
Nouvelles versions de STIG	Introduction de la prise en charge d'Ubuntu, mise à jour des versions STIG et application de STIGS pour la version du deuxième trimestre 2022.	20 juillet 2022
Mise à jour du document : navigation pour la page de création d'un document de composant YAML	Le contenu du document du composant Create YAML a été déplacé vers sa propre page et mis à jour les autres pages pour y faire référence.	7 juin 2022
Nouvelles versions de STIG	Versions STIG mises à jour et STIGS appliqué pour la version du premier trimestre 2022.	25 avril 2022
Module ExecuteDocument d'action ajouté	Ajout de la documentation pour le module `ExecuteDocument` d'action ci-dessousGeneral execution .	28 mars 2022

Mise à jour des fonctionnalités : prise en charge d'un lancement plus rapide de Windows AMI	Ajout de paramètres de configuration de distribution pour permettre un lancement plus rapide des AMI Windows.	21 février 2022
Version de maintenance : mise à jour de l'empreinte AWSTOE numérique binaire	Empreinte binaire mise à jour pour le certificat de AWSTOE signataire.	18 février 2022
Version de fonctionnalité : configurer la saisie pour AWSTOE	Ajout de la prise en charge de l'utilisation d'un fichier de configuration JSON comme entrée pour la AWSTOE run commande.	3 février 2022
Nouvelles versions de STIG	Versions STIG mises à jour et STIGS appliqué pour la version du quatrième trimestre 2021. Une section a également été ajoutée pour les nouveaux composants du SCAP Compliance Checker (SCC).	22 décembre 2021
Version de fonctionnalité : intégration de machines virtuelles Import/Export (VMIE)	Ajout de la prise en charge de l'importation de machines virtuelles via tous les canaux (console API/CLI, etc.) et de l'exportation de machines virtuelles via API/CLI. L'exportation de machines virtuelles n'est actuellement pas disponible depuis la console Image Builder.	20 décembre 2021

Version de fonctionnalité : partage d'AMI pour AWS Organizations et UO	Configuration de distribution mise à jour pour ajouter la prise en charge du partage des AMI de sortie avec AWS Organizations et des unités d'organisation.	24 novembre 2021
Mise à jour des documents : mettre à jour les étapes et les phases des composants	Contenu étendu sur les étapes des composants dans Image Builder et sur la manière dont celles-ci interagissent avec les phases des AWSTOE composants.	22 septembre 2021
Mise à jour du document : ajout de contenu CloudTrail d'intégration	Ajout d'un résumé de la surveillance et CloudTrail d'un contenu d'intégration.	17 septembre 2021
Nouvelles versions de STIG	Versions STIG mises à jour et STIGS appliqué pour la version du troisième trimestre 2021.	10 septembre 2021
Publication de la fonctionnalité : EventBridge intégration avec Amazon	Ajout d' EventBridge un support qui vous permet de connecter Image Builder à des événements connexes Services AWS et de lancer des événements en fonction des règles définies dans EventBridge.	18 août 2021
Mise à jour du document : réorganiser les pages AWSTOE	AWSTOE Pages réorganisées pour plus de clarté.	11 août 2021

Version de fonctionnalité : composants paramétrés et configuration d'instance supplémentaire	Ajout de la prise en charge de la spécification de paramètres pour personnaliser les composants des recettes. Configuration étendue des instances EC2 utilisées pour créer et tester des images, y compris la possibilité de spécifier des commandes à exécuter au lancement, et un contrôle accru sur l'installation et la suppression de l'agent Systems Manager.	7 juillet 2021
Nouvelles versions du STIG	Versions STIG mises à jour et STIGS appliqué pour la version du deuxième trimestre 2021.	30 Juin 2021
Amélioration : améliorations du balisage	Amélioration de la messagerie concernant le balisage des ressources.	25 juin 2021
Mise à jour de la fonctionnalité : intégration du modèle de lancement	Ajout de la prise en charge de l'utilisation des modèles de lancement Amazon EC2 pour la distribution des AMI dans les paramètres de distribution.	7 avril 2021
Mise à jour des fonctionnalités : améliorations apportées à la création de conteneurs	Ajout de la prise en charge de la configuration des mappages de périphériques en mode bloc et de la spécification des AMI à utiliser comme image de base pour les constructions de conteneurs.	7 avril 2021

Nouvelles versions du STIG	Versions STIG mises à jour et STIGS appliqués.	5 mars 2021
Mettre à jour les expressions cron	Le traitement cron d'Image Builder est mis à jour pour augmenter la granularité des expressions cron à la minute près et utiliser un moteur de planification cron standard. Les exemples sont mis à jour avec le nouveau format.	8 février 2021
Version de fonctionnalité : support des conteneurs	Ajout de la prise en charge de la création d'images de conteneurs Docker à l'aide d'Image Builder, avec enregistrement et stockage des images résultantes sur Amazon Elastic Container Registry (Amazon ECR). Le contenu a été réorganisé pour refléter les nouvelles fonctionnalités et s'adapter à la croissance future.	17 décembre 2020

Documentation cron restructurée	Cette page présente désormais plus d'informations sur la façon dont cron fonctionne avec les builds de pipeline d'Image Builder, et inclut des détails sur l'heure UTC. Les caractères génériques qui ne sont pas autorisés pour certains champs ont été supprimés. Les exemples incluent désormais des exemples d'expressions pour la console et la CLI.	13 novembre 2020
Console version 2.0 : modification du pipeline mise à jour	Le contenu change dans les didacticiels de démarrage et de création de pipelines, ainsi que dans la page de gestion des pipelines d'images, afin d'intégrer les nouvelles fonctionnalités et le flux de la console.	13 novembre 2020
Nouvelles versions du STIG	Versions STIG mises à jour et STIGS appliqués. Remarque : le format de la liste a été modifié pour afficher les STIG qui sont appliqués par défaut.	15 octobre 2020
Support pour les constructions en boucle dans AWSTOE	Créez des constructions en boucle pour définir une séquence d'instructions répétée dans l' AWSTOE application.	29 juillet 2020

Soutien au développement local de AWSTOE composants	Développez et testez des composants d'image localement avec l' AWSTOE application.	28 juillet 2020
AMI cryptées	EC2 Image Builder ajoute la prise en charge de la distribut ion chiffrée des AMI.	1er juillet 2020
AutoScaling dépréciation	Obsolète de l'utilisation de AutoScaling pour lancer des instances.	15 juin 2020
Prise en charge de la connectivité via AWS PrivateLi nk	Vous pouvez établir une connexion privée entre votre VPC et EC2 Image Builder en créant un point de terminaison VPC d'interface. Les points de terminaison de l'interface sont alimentés par AWS PrivateLi nk une technologie qui vous permet d'accéder en privé aux API d'Image Builder sans passerelle Internet, périphéri que NAT, connexion VPN ou connexion AWS Direct Connect. Les instances de votre VPC n'ont pas besoin d'adresses IP publiques pour communiquer avec les API Image Builder. Le trafic entre votre VPC et Image Builder ne quitte pas le réseau Amazon.	10 juin 2020
Nouvelles versions du STIG	Versions STIG mises à jour et STIGS appliqués.	23 janvier 2020

Résolution des problèmes	Ajout de scénarios de dépannage généraux.	22 janvier 2020
Composants STIG	Vous pouvez créer des STIG-compliant images avec des composants AWSTOE STIG.	22 janvier 2020

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.