



Guide de l'utilisateur

Amazon ECR



Version de l'API 2015-09-21

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

Amazon ECR: 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

Présentation d'Amazon ECR	1
Concepts et composants	1
Cas d'utilisation courants	4
Fonctions d'Amazon ECR	6
Inscrivez-vous pour un Compte AWS	7
Comment démarrer avec Amazon ECR	7
Tarification pour Amazon ECR	7
Déplacer une image tout au long de son cycle de vie	8
Conditions préalables	8
Installez le AWS CLI	8
Installer Docker	8
Étape 1 : Créer une image Docker	10
Étape 2 : Création d'un référentiel	12
Étape 3 : Authentifiez-vous auprès de votre registre par défaut	12
Étape 4 : Transmettre une image à Amazon ECR	13
Étape 5 : Extraire une image d'Amazon ECR	15
Étape 6 : Supprimer une image	15
Étape 7 : Supprimer un référentiel	16
Optimisation des performances	17
Demandes	19
Commencer avec IPv6	19
Test de compatibilité d'adresses IP	20
Envoi de demandes à l'aide de points de terminaison Dual-Stack	21
Utilisation des points de terminaison Amazon ECR depuis la CLI docker	21
Utilisation des IPv6 adresses dans les politiques IAM	22
Registre privé	24
Concepts de registre	24
Authentification de registre	24
Utilisation de l'assistant d'informations d'identification Amazon ECR	25
Utiliser un jeton d'autorisation	25
Utiliser l'authentification API HTTP	26
Paramètres de registre	27
Montage Blob	28
Concepts de montage Blob	29

Configuration de montage du blob	29
Autorisations de registre	30
Exemples de politiques de registre	30
Octroi d'autorisations pour la réplication entre comptes	33
Octroi d'autorisations pour le cache d'extraction	35
Référentiels privés	37
Concepts de référentiel	37
Création d'un référentiel pour stocker des images	38
Étapes suivantes	40
Affichage des détails d'un référentiel	41
Suppression d'un référentiel	42
Politiques de référentiel	43
Politiques de référentiel vs politiques IAM	43
Exemples de politiques de référentiel	45
Définir une instruction de politique de référentiel	51
Balisage d'un référentiel	53
Principes de base des étiquettes	53
Identification de vos ressources pour facturation	53
Ajout de balises	54
Suppression d'étiquettes	55
Images privées	57
Transmission d'une image	58
Autorisations IAM requises	58
Pousser une image Docker	60
Transmission d'une image multi-architecture	62
Pousser les Charts de Helm	64
Supprimer des artefacts	66
Afficher les détails d'image	69
Extraire une image	70
Extraction de l'image du conteneur Amazon Linux	71
Supprimer une image	73
Archivage d'une image	75
Qu'est-ce que la classe de stockage d'archives ECR ?	75
Archivage d'une image	76
Restaurer une image	78
Modifier l'étiquette d'une image	79

Empêcher le remplacement des balises d'image	82
Configuration de la mutabilité des balises d'image ()AWS Management Console	82
Configuration de la mutabilité des balises d'image ()AWS CLI	84
Formats de manifeste d'images de conteneur	85
Conversion du manifeste d'image Amazon ECR	86
Utiliser des images Amazon ECR avec Amazon ECS	87
Autorisations IAM requises	87
Spécifier une image Amazon ECR dans une définition de tâche	89
Utiliser des images Amazon ECR avec Amazon EKS	90
Autorisations IAM requises	90
Installation d'un graphique Helm sur un cluster Amazon EKS	91
Images de signes	93
Choisissez une méthode de signature	93
Considérations	93
Signature gérée	94
Conditions préalables	94
Prise en main	96
Considérations	98
Vérification de signature	98
Vérification gérée avec Amazon EKS	99
Contrôleur d'admission Lambda pour Amazon ECS	99
Vérification manuelle avec Notation CLI	99
Configuration de l'authentification pour le client Notation	99
Signature manuelle	100
Conditions préalables	100
Scannez les images pour détecter les vulnérabilités	101
Filtres pour référentiels	102
Filtrer les caractères génériques	102
Analyse améliorée	103
Considérations relatives à l'analyse améliorée	103
Modification de la durée de l'analyse améliorée	105
Autorisations IAM requises	105
Configuration de la numérisation améliorée	106
EventBridge événements	108
Récupération des résultats	114
Analyse de base	115

Support du système d'exploitation pour la numérisation de base	116
Configuration de la numérisation de base	116
Numérisation manuelle d'une image	117
Récupération des résultats	119
Résolution des problèmes de numérisation d'images	120
Présentation des statuts d'analyse SCAN_ELIGIBILITY_EXPIRED	121
Synchroniser un registre en amont	122
Modèles de création de référentiels	123
Considérations relatives à l'utilisation des règles du cache d'extraction	123
Autorisations IAM requises	126
Utilisation des autorisations de registre	126
Étapes suivantes	128
Configuration des autorisations entre comptes ECR et ECR PTC	129
Politiques IAM requises pour que l'ECR entre comptes puisse extraire le cache de l'ECR	129
Création d'une règle de mise en cache par extraction	131
Conditions préalables	132
À l'aide du AWS Management Console	132
À l'aide du AWS CLI	142
Étapes suivantes	146
Validation de la règle du cache d'extraction	146
Extraction d'une image à l'aide d'une règle de mise en cache par extraction	148
Stockage des informations d'identification de votre référentiel en amont	150
Personnalisation des préfixes de référentiels	160
Résolution des problèmes liés au cache d'extraction	161
Répliquer des images	163
Exigences relatives à la politique de réplication	163
Présentation de la configuration des politiques	163
Exigences relatives à la politique du registre de destination	164
Exigences relatives au compte source	165
Idées fausses courantes	165
Résolution des problèmes de réplication	165
Considérations relatives à la réplication d'images privées	166
Exemples de réplication	168
Exemple : configuration de la réplication inter-régions sur une même région de destination .	168
Exemple : Configuration de la réplication inter-régions à l'aide d'un filtre de référentiel	169

Exemple : Configuration de la réplication inter-régions vers plusieurs régions de destination	169
Exemple : Configuration de la réplication inter-comptes	170
Exemple : Spécification de plusieurs règles dans une configuration	170
Exemple : suppression de tous les paramètres de réplication	171
Configurer une réplication	172
Suppression de la réplication	174
Modèles de création de référentiels	176
Comment ça marche	176
Création d'un modèle de création de référentiel	180
Autorisations IAM pour créer des modèles de création de référentiels	180
Créer une stratégie personnalisée	181
Créer un rôle IAM	182
Création d'un modèle de création de référentiel	184
Mise à jour des modèles de création de référentiel	189
Suppression d'un modèle de création de référentiel	190
Automatisez le nettoyage des images	192
Fonctionnement des politiques de cycle de vie	192
Règles d'évaluation de la politique de cycle de vie	193
Créer un aperçu de politique de cycle de vie	195
Créer une politique de cycle de vie	197
Prérequis	198
Exemples de politiques de cycle de vie	200
Modèle de politique de cycle de vie	200
Filtrer sur l'ancienneté des images	200
Filtrage en fonction de l'heure de la dernière extraction	201
Filtrage sur le temps de transition des archives	202
Filtrer sur le décompte d'images	203
Filtrer sur plusieurs règles	203
Filtrer sur plusieurs étiquettes dans une seule règle	206
Filtrer sur toutes les images	208
Exemples d'Archive	211
Propriétés des politiques de cycle de vie	213
Priorité de la règle	214
Description	214
État de l'étiquetage	214

Liste des modèles de balises	215
Liste des préfixes d'étiquette	215
Classe de stockage	216
Type de décompte	216
Unité de décompte	216
Chiffre du décompte	217
Action	217
Exclusions liées aux mises à jour instantanées	219
Gestion des exclusions liées aux mises à jour effectuées au moment du téléchargement	219
Considérations relatives aux exclusions liées aux mises à jour automatiques	222
Sécurité	223
Gestion des identités et des accès	224
Public ciblé	224
Authentification par des identités	225
Gestion de l'accès à l'aide de politiques	226
Fonctionnement du registre de conteneur Amazon Elastic avec IAM	228
Identity-based exemples de politiques	233
Utilisation du contrôle Tag-Based d'accès	237
AWS politiques gérées pour Amazon ECR	239
Utilisation des rôles liés à un service	247
Résolution des problèmes	256
Protection des données	258
Chiffrement au repos	259
Validation de conformité	268
Sécurité de l'infrastructure	268
Points de terminaison d'un VPC d'interface (AWS PrivateLink)	269
Cross-service prévention confuse des adjoints	279
Contrôle	281
Visualiser vos Service Quotas et définir des alarmes	282
Métriques d'utilisation	283
Rapports d'utilisation	286
Métriques de référentiel	286
CloudWatch Indicateurs habilitants	286
Métriques et dimensions disponibles	287
Afficher les métriques avec CloudWatch	287
Événements et EventBridge	288

Exemples d'événements d'Amazon ECR	288
Journalisation des actions AWS CloudTrail avec	295
Informations Amazon ECR dans CloudTrail	295
Présentation des entrées des fichiers journaux Amazon ECR	297
Utilisation des AWS SDK	315
Exemples de code	317
Principes de base	318
Bonjour Amazon ECR	318
Principes de base	323
Actions	379
Scénarios	436
Commencer à utiliser Amazon ECR	436
Service quotas	445
Gérer vos quotas de service Amazon ECR dans AWS Management Console	451
Création d'une CloudWatch alarme pour surveiller les métriques d'utilisation de l'API	452
Résolution des problèmes	454
Résolution des problèmes liés à Docker	454
Les journaux Docker ne contiennent pas les messages d'erreur attendus	454
Erreur : « Filesystem Verification Failed » ou « 404: Image Not Found » lors de l'extraction d'une image d'un référentiel Amazon ECR	455
Erreur : « Filesystem Layer Verification Failed » lors de l'extraction d'images d'Amazon ECR	456
Erreurs HTTP 403 ou « no basic auth credentials » lors de la transmission au référentiel	456
Dépannage des messages d'erreur Amazon ECR	457
HTTP 429 : trop de requêtes ou ThrottleException	457
HTTP 403 : « User [arn] is not authorized to perform [operation] »	458
HTTP 404 : « Repository Does Not Exist »	459
Erreur : impossible d'effectuer une connexion interactive à partir d'un appareil autre que TTY	459
Utilisation de Podman avec Amazon ECR	460
Utiliser Podman pour s'authentifier auprès d'Amazon ECR	460
Utilisation de l'assistant d'identification Amazon ECR avec Podman	460
Extraire des images depuis Amazon ECR avec Podman	461
Exécution de conteneurs pour Amazon ECR avec Podman	461
Transférer des images vers Amazon ECR avec Podman	461
Historique du document	463

Registre de conteneur Amazon Elastic

Amazon Elastic Container Registry (Amazon ECR) est AWS un service géré de registre d'images de conteneurs sécurisé, évolutif et fiable. Amazon ECR prend en charge les référentiels privés dotés d'autorisations basées sur les ressources à l'aide d'IAM. AWS Cela permet aux utilisateurs spécifiés ou aux instances Amazon EC2 de pouvoir accéder à vos référentiels et à vos images de conteneurs. Vous pouvez utiliser votre CLI préférée pour pousser, extraire et gérer des images Docker, des images OCI (Open Container Initiative) et des artefacts compatibles OCI.

Note

Amazon ECR prend également en charge les référentiels d'images de conteneurs publics. Pour en savoir plus, consultez [Qu'est-ce qu'Amazon ECR public](#) dans le guide de l'utilisateur Amazon ECR Public.

L'équipe des services de AWS conteneurs tient à jour une feuille de route publique sur GitHub. Il contient des informations sur le travail des équipes et permet à tous les AWS clients de donner des commentaires directs. Pour en savoir plus, consultez [Feuille de route des conteneurs AWS](#).

Concepts et composants d'Amazon ECR

Amazon ECR est un service de registre de conteneurs Docker entièrement géré fourni par AWS. Il vous permet de stocker, de gérer et de déployer des images de conteneurs Docker de manière sécurisée et fiable. Ces concepts et composants fonctionnent ensemble pour fournir un service de registre de conteneurs Docker sécurisé, évolutif et fiable au sein du AWS, vous permettant de gérer et de déployer efficacement vos applications conteneurisées.

Voici quelques concepts et composants clés d'Amazon ECR :

Registre

Un registre Amazon ECR est un référentiel privé fourni à chaque AWS compte, dans lequel vous pouvez créer un ou plusieurs référentiels. Ces référentiels vous permettent de stocker et de distribuer des images Docker, des images Open Container Initiative (OCI) et d'autres OCI-compatible artefacts au sein de votre environnement. AWS Pour de plus amples informations, veuillez consulter [Registre privé Amazon ECR](#).

Jeton d'autorisation

Votre client doit s'authentifier auprès d'un registre privé Amazon ECR en tant qu'utilisateur AWS avant de pouvoir transmettre et extraire des images. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).

Référentiel

Un référentiel dans Amazon ECR est une collection logique dans laquelle vous pouvez stocker vos images Docker, vos images Open Container Initiative (OCI) et d'autres artefacts. OCI-compatible Au sein d'un même registre Amazon ECR, vous pouvez disposer de plusieurs référentiels pour organiser les images de vos conteneurs. Pour de plus amples informations, veuillez consulter [Référentiels privés Amazon ECR](#).

Politique de dépôt

Vous pouvez contrôler l'accès à vos référentiels et à leur contenu grâce aux politiques de référentiel. Pour de plus amples informations, veuillez consulter [Politiques relatives aux référentiels privés dans Amazon ECR](#).

Image

Vous pouvez transmettre et extraire des images de conteneur à vos référentiels. Vous pouvez utiliser ces images localement sur votre système de développement ou les utiliser dans des définitions de tâche Amazon ECS et des spécifications de pod d'Amazon EKS. Pour plus d'informations, consultez [Utiliser des images Amazon ECR avec Amazon ECS](#) et [Utiliser des images Amazon ECR avec Amazon EKS](#).

Politique de cycle de vie

Les politiques de cycle de vie d'Amazon ECR vous permettent de gérer le cycle de vie de vos images en définissant des règles d'élagage et d'expiration des images anciennes ou inutilisées. Pour de plus amples informations, veuillez consulter [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).

Numérisation d'images

Amazon ECR fournit une fonctionnalité de numérisation d'image intégrée qui permet d'identifier les vulnérabilités logicielles dans vos images de conteneur. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR](#).

Contrôle d'accès

Amazon ECR utilise IAM pour contrôler l'accès à vos référentiels. Vous pouvez créer des utilisateurs, des groupes et des rôles IAM dotés d'autorisations spécifiques pour envoyer, extraire ou gérer les référentiels Amazon ECR. Pour de plus amples informations, veuillez consulter [Sécurité dans le registre de conteneur Amazon Elastic](#).

Cross-account et Cross-region réplication

Amazon ECR prend en charge la réplication d'images sur plusieurs AWS comptes et régions pour une disponibilité accrue et une latence réduite. Pour de plus amples informations, veuillez consulter [Réplication d'images privées sur Amazon ECR](#).

Chiffrement

Amazon ECR prend en charge le chiffrement côté serveur de vos images Docker au repos à l'aide de AWS KMS. Pour de plus amples informations, veuillez consulter [Protection des données dans Amazon ECR](#).

AWS Command Line Interface Integration

AWS CLI fournit des commandes pour interagir avec les référentiels Amazon ECR, telles que la création, la mise en vente, le transfert et l'extraction d'images.

AWS Management Console

Amazon ECR peut également être géré via le AWS Management Console, fournissant une interface Web conviviale pour travailler avec vos référentiels et vos images.

AWS CloudTrail

Amazon ECR s'intègre à Amazon ECR AWS CloudTrail, ce qui vous permet de consigner et d'auditer les appels d'API passés à Amazon ECR à des fins de sécurité et de conformité. Pour de plus amples informations, veuillez consulter [Journalisation des actions Amazon ECR avec AWS CloudTrail](#).

Amazon CloudWatch

Amazon ECR fournit des métriques et des journaux qui peuvent être surveillés à l'aide de ce Amazon CloudWatch logiciel, ce qui vous permet de suivre les performances et l'utilisation de vos référentiels Amazon ECR. Pour de plus amples informations, veuillez consulter [Métriques de référentiel Amazon ECR](#).

Signature gérée

La signature gérée génère automatiquement des signatures cryptographiques lorsque les images sont transmises à Amazon ECR, ce qui simplifie la signature des images des conteneurs. Pour de plus amples informations, veuillez consulter [Signature gérée](#).

Cas d'utilisation courants dans Amazon ECR

Amazon ECR est un service de registre de conteneurs Docker entièrement géré proposé par AWS. Il fournit un référentiel sécurisé et évolutif pour le stockage et la distribution d'images de conteneurs Docker, ce qui en fait un composant essentiel des déploiements d'applications conteneurisées. Amazon ECR simplifie le processus de création, de distribution et d'exécution d'applications conteneurisées dans différents AWS services et environnements sur site.

Voici quelques exemples d'utilisation clés d'Amazon ECR :

Stockage et distribution d'images de conteneurs

Amazon ECR sert de référentiel centralisé pour le stockage et la distribution d'images de conteneurs Docker au sein d'une organisation ou à des fins de consommation publique. Les développeurs peuvent transférer leurs images de conteneur vers Amazon ECR, puis les extraire de n'importe quel environnement informatique AWS, tel qu'Amazon EC2, AWS Fargate ou Amazon EKS. Pour de plus amples informations, veuillez consulter [Référentiels privés Amazon ECR](#).

Intégration continue et déploiement continu (CI/CD)

Amazon ECR s'intègre parfaitement à AWS CodeBuild, AWS CodePipeline, et à d'autres CI/CD outils, permettant la création, le test et le déploiement automatisés d'applications conteneurisées. Les images de conteneur peuvent être automatiquement transmises à Amazon ECR dans le cadre du CI/CD pipeline, ce qui garantit un déploiement cohérent et fiable dans différents environnements.

Architecture des microservices

Amazon ECR convient parfaitement aux architectures de microservices, dans lesquelles les applications sont décomposées en services plus petits et découplés, conditionnés sous forme de conteneurs. Chaque microservice peut disposer de sa propre image de conteneur stockée dans Amazon ECR, ce qui permet le développement, le déploiement et le dimensionnement indépendants de services individuels.

Hybride et Multi-Cloud déploiements

Amazon ECR permet d'extraire des images de conteneurs depuis d'autres registres de conteneurs, tels que Docker Hub ou des registres tiers. Cela permet aux entreprises de maintenir un modèle de déploiement cohérent dans les environnements hybrides ou multicloud, en utilisant Amazon ECR comme référentiel central pour les images de conteneurs.

Contrôle d'accès et sécurité

Amazon ECR fournit des mécanismes de contrôle d'accès précis, permettant aux entreprises de contrôler qui peut envoyer ou extraire des images de conteneurs du registre. Il s'intègre également à Gestion des identités et des accès AWS l'authentification et à l'autorisation, garantissant ainsi un accès sécurisé aux images des conteneurs. Pour de plus amples informations, veuillez consulter [Sécurité dans le registre de conteneur Amazon Elastic](#).

Analyse des vulnérabilités des images

Amazon ECR propose une analyse automatique des images de conteneurs pour détecter les vulnérabilités logicielles et les éventuelles erreurs de configuration, afin de garantir un environnement de conteneurs sécurisé et conforme. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR](#).

Registre des conteneurs privés

Pour les entreprises ayant des exigences strictes en matière de sécurité ou de conformité, Amazon ECR peut être utilisé comme registre de conteneurs privé, garantissant ainsi que les images de conteneurs sensibles ne sont pas exposées aux registres publics et ne sont accessibles que dans l'environnement de AWS l'entreprise. Pour de plus amples informations, veuillez consulter [Registre privé Amazon ECR](#).

Déploiement d'applications distribuées dans le monde entier avec Amazon ECR Replication

En tirant parti de la fonctionnalité de réplication Amazon ECR, vous pouvez centraliser les images de vos applications Web conteneurisées dans un référentiel principal, permettre une distribution automatisée dans plusieurs AWS régions, garantir des déploiements mondiaux cohérents avec une faible latence dans le monde entier et réduire la charge opérationnelle. Pour de plus amples informations, consultez [Réplication d'images privées sur Amazon ECR](#).

Nettoyage automatique des images de conteneurs périmées

Les politiques de cycle de vie d'Amazon ECR permettent le nettoyage automatique des images de conteneurs périmées en fonction de règles définies telles que l'âge, le nombre ou les balises,

l'optimisation des coûts de stockage, le maintien d'un registre organisé, le renforcement de la sécurité et de la conformité, et la rationalisation des flux de travail de développement grâce à l'automatisation. Pour de plus amples informations, consultez [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).

Fonctions d'Amazon ECR

Amazon ECR offre les fonctions suivantes :

- Les politiques de cycle de vie aident à gérer le cycle de vie des images dans vos référentiels. Vous définissez des règles qui entraînent le nettoyage des images inutilisées. Vous pouvez tester les règles avant de les appliquer à votre référentiel. Pour de plus amples informations, veuillez consulter [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).
- La numérisation des images permet d'identifier les vulnérabilités logicielles dans vos images de conteneur. Chaque référentiel peut être configuré pour numérisation sur poussée. Cela garantit que chaque nouvelle image poussée vers le référentiel est numérisée. Vous pouvez ensuite récupérer les résultats de la numérisation d'image. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR](#).
- Cross-Region et la réplique entre comptes vous permet d'avoir plus facilement vos images là où vous en avez besoin. Elle est configurée en tant que paramètre de registre et en fonction de la région. Pour de plus amples informations, veuillez consulter [Paramètres du registre privé dans Amazon ECR](#).
- Les règles de mise en cache par extraction permettent de mettre en cache les référentiels dans un registre en amont de votre registre privé Amazon ECR. En utilisant une règle de mise en cache par extraction, Amazon ECR contactera périodiquement le registre en amont pour s'assurer que l'image mise en cache dans votre registre privé Amazon ECR est à jour. Pour de plus amples informations, veuillez consulter [Synchroniser un registre en amont avec un registre privé Amazon ECR](#).
- Les modèles de création de référentiels vous permettent de définir les paramètres des référentiels créés par Amazon ECR en votre nom lors d'actions d'extraction du cache, de création en mode push ou de réplique. Vous pouvez spécifier l'immuabilité des balises, la configuration du chiffrement, les politiques de référentiel, les politiques de cycle de vie et les balises de ressources pour les référentiels créés automatiquement. Pour de plus amples informations, veuillez consulter [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplique](#).

- La signature gérée génère automatiquement des signatures cryptographiques lorsque les images sont transmises à Amazon ECR, ce qui simplifie la signature des images des conteneurs. Pour de plus amples informations, veuillez consulter [Signature gérée](#).

Inscrivez-vous pour un Compte AWS

Pour commencer AWS, vous avez besoin d'un Compte AWS. Pour plus d'informations sur la création d'un Compte AWS, voir [Getting started with an Compte AWS](#) dans le Guide de Gestion de compte AWS référence.

Comment démarrer avec Amazon ECR

Si vous utilisez Amazon Elastic Container Service (Amazon ECS) ou Amazon Elastic Kubernetes Service (Amazon EKS), notez que la configuration de ces deux services est similaire à celle d'Amazon ECR, car Amazon ECR est une extension des deux services.

Lorsque vous utilisez le AWS Command Line Interface avec Amazon ECR, utilisez une version du AWS CLI qui prend en charge les dernières fonctionnalités d'Amazon ECR. Si aucune fonctionnalité Amazon ECR n'est prise en charge dans le AWS CLI, passez à la dernière version du AWS CLI. Pour plus d'informations sur l'installation de la dernière version du AWS CLI, voir [Installer ou mettre à jour vers la dernière version du AWS CLI dans le](#) guide de AWS Command Line Interface l'utilisateur.

Pour savoir comment transférer une image de conteneur vers un référentiel Amazon ECR privé à l'aide de Docker AWS CLI et, consultez. [Déplacement d'une image tout au long de son cycle de vie dans Amazon ECR](#)

Tarification pour Amazon ECR

Avec Amazon ECR, vous payez en fonction de la quantité de données que vous stockez dans vos référentiels, du transfert de données provenant de vos captures et extractions d'images, ainsi que des actions que vous choisissez d'effectuer, telles que la signature et la réplication d'images. Pour plus d'informations, consultez [Tarification Amazon ECR](#).

Déplacement d'une image tout au long de son cycle de vie dans Amazon ECR

Si vous utilisez Amazon ECR pour la première fois, suivez les étapes suivantes avec la CLI Docker et AWS CLI pour créer un exemple d'image, vous authentifier auprès du registre par défaut et créer un référentiel privé. Transférez ensuite une image vers le dépôt privé et extrayez-la de celui-ci. Lorsque vous avez terminé avec l'exemple d'image, supprimez-le et le référentiel.

Pour utiliser le AWS Management Console au lieu du AWS CLI, voir [the section called “Création d'un référentiel pour stocker des images”](#).

Pour plus d'informations sur les autres outils disponibles pour gérer vos AWS ressources, notamment les différents AWS SDK, boîtes à outils IDE et outils de ligne de PowerShell commande Windows, consultez. <http://aws.amazon.com/tools/>

Conditions préalables

Si la dernière version AWS CLI de Docker n'est pas installée et prête à être utilisée, procédez comme suit pour installer ces deux outils.

Installez le AWS CLI

Pour l'utiliser AWS CLI avec Amazon ECR, installez la dernière AWS CLI version. Pour plus d'informations, consultez [Installation de la AWS Command Line Interface](#) dans le Guide de l'utilisateur de la AWS Command Line Interface .

Installer Docker

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

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

Si vous avez déjà installé Docker, passez à [Étape 1 : Créer une image Docker](#).

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

1. Lancez une instance avec la dernière AMI Amazon Linux 2023. Pour plus d'informations, consultez la section [Lancement d'une instance](#) dans le guide de l'utilisateur Amazon EC2.
2. Connectez-vous à votre instance. Pour plus d'informations, consultez [Connect to your Linux instance](#) dans le guide de l'utilisateur Amazon EC2.
3. Mettez à jour les packages installés et le cache du package sur votre instance.

```
sudo yum update -y
```

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

```
sudo yum install docker
```

5. Lancez le service Docker.

```
sudo service docker start
```

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

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

7. Déconnectez-vous et reconnectez-vous pour récupérer les nouvelles autorisations de groupe `docker`. Vous pouvez effectuer ces opérations en fermant votre fenêtre de terminal SSH actuelle et en vous reconnectant à votre instance dans une nouvelle fenêtre. Votre nouvelle session SSH disposera des autorisations de groupe `docker` appropriées.
8. Vérifiez que `ec2-user` peut exécuter les commandes Docker sans `sudo`.

```
docker info
```

Note

Dans certains cas, vous devrez peut-être redémarrer votre instance pour autoriser `ec2-user` à accéder au démon Docker. Essayez de redémarrer l'instance si vous voyez l'erreur suivante :

Cannot connect to the Docker daemon. Is the docker daemon running on this host?

Étape 1 : Créer une image Docker

Au cours de cette étape, vous créez une image Docker pour une application web simple et vous la testez sur votre système local ou l'instance Amazon EC2.

Créer une image Docker d'une application web simple

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

```
touch Dockerfile
```

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

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

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

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html


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

EXPOSE 80

CMD /root/run_apache.sh
```

Ce fichier Dockerfile utilise l'image publique Amazon Linux 2 hébergée sur Amazon ECR Public. Les instructions RUN mettent à jour les caches du package, installent certains packages logiciels pour le serveur web et écrivent ensuite le message « Hello World! » contenu à la racine du document des serveurs Web. L'instruction EXPOSE expose le port 80 sur le conteneur et l'instruction CMD démarre le serveur Web.

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

 Note

Certaines versions de Docker exigent le chemin d'accès complet à votre Dockerfile dans la commande suivante au lieu du chemin d'accès relatif indiqué ci-après.

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

4. Répertoriez l'image de votre conteneur.


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

Sortie :

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

5. Exécutez la nouvelle image. L'option `-p 80:80` mappe le port exposé 80 du conteneur au port 80 du système hôte. Pour en savoir plus sur docker run, accédez à [Docker run reference](#).

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

 Note

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

- Ouvrez un navigateur et pointez vers le serveur qui exécute Docker et qui héberge votre conteneur.
 - Si vous utilisez une instance EC2, il s'agit de la valeur de DNS public du serveur, c'est-à-dire la même adresse que celle que vous utilisez pour vous connecter à l'instance avec le protocole SSH. Assurez-vous que le groupe de sécurité de votre instance autorise le trafic entrant sur le port 80.
 - Si vous utilisez Docker localement, pointez votre navigateur sur <http://localhost/>.
 - Si vous l'utilisez docker-machine sur un ordinateur Windows ou Mac, recherchez l'adresse IP de la VirtualBox machine virtuelle hébergeant Docker avec la docker-machine ip commande, en la remplaçant par *machine-name* le nom de la machine docker que vous utilisez.

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

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

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

Étape 2 : Création d'un référentiel

Maintenant que vous disposez d'une image à transmettre à Amazon ECR, vous devez créer un référentiel afin de la contenir. Dans cet exemple, vous créez un référentiel nommé `hello-repository` dans lequel vous pourrez transmettre l'image `hello-world:latest`. Pour créer un référentiel, exécutez la commande suivante :

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

Étape 3 : Authentifiez-vous auprès de votre registre par défaut

Après avoir installé et configuré le AWS CLI, authentifiez la CLI Docker dans votre registre par défaut. Ainsi, la commande docker peut pousser et extraire des images avec Amazon ECR. AWS CLI Fournit une `get-login-password` commande pour simplifier le processus d'authentification.

Note

Votre principal IAM doit être `ecr:GetAuthorizationToken` autorisé à s'authentifier auprès d'un registre. Pour de plus amples informations, veuillez consulter [AWS politiques gérées pour Amazon Elastic Container Registry](#).

Pour authentifier Docker dans un registre Amazon ECR avec `get-login-password`, exécutez la commande `aws ecr get-login-password`. Lorsque vous passez le jeton d'authentification à la commande `docker login`, utilisez la valeur AWS pour le nom d'utilisateur et spécifiez l'URI de registre Amazon ECR auquel vous voulez vous authentifier. Si vous vous authentifier sur plusieurs registres, vous devrez répéter la commande pour chacun d'eux.

Important

Si vous recevez une erreur, installez la dernière version de la CLI ou effectuez une mise à niveau vers cette version AWS CLI. Pour en savoir plus, consultez [Installer la AWS Command Line Interface](#) dans le guide de l'utilisateur AWS Command Line Interface .

- [get-login-password](#) (AWS CLI)

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

- [Get-ECRLoginCommand](#) (AWS Tools for Windows PowerShell)

```
(Get-ECRLoginCommand).Password | docker login --username AWS --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Étape 4 : Transmettre une image à Amazon ECR

Vous pouvez maintenant transmettre l'image au référentiel Amazon ECR que vous avez créé à la section précédente. Utilisez la docker CLI pour envoyer des images une fois que les conditions préalables suivantes sont remplies :

- La version minimale de docker est installée : 1.7.

- Le jeton d'autorisation Amazon ECR a été configuré avec `docker login`.
- Le référentiel Amazon ECR existe et l'utilisateur dispose d'un accès lui permettant de transmettre des images dans le référentiel.

Une fois ces conditions remplies, vous pouvez transmettre l'image au référentiel qui vient d'être créé dans le registre par défaut de votre compte.

Étiqueter et transmettre une image à Amazon ECR

1. Répertoriez les images que vous avez stockées localement afin d'identifier l'image à étiqueter et à transmettre.

```
docker images
```

Sortie :

REPOSITORY	TAG	IMAGE ID	CREATED
hello-world	latest	e9ffedc8c286	4 minutes ago
241MB			

2. Étiquetez l'image à transmettre à votre référentiel.

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

3. Transmettez l'image.

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

Sortie :

```
The push refers to a repository [aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository] (len: 1)
e9ae3c220b23: Pushed
a6785352b25c: Pushed
0998bf8fb9e9: Pushed
0a85502c06c9: Pushed
```

```
latest: digest: sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636EXAMPLE
size: 6774
```

Étape 5 : Extraire une image d'Amazon ECR

Une fois votre image envoyée dans votre référentiel Amazon ECR, vous pouvez l'extraire d'autres emplacements. Utilisez la docker CLI pour extraire des images une fois que les conditions préalables suivantes sont remplies :

- La version minimale de docker est installée : 1.7.
- Le jeton d'autorisation Amazon ECR a été configuré avec `docker login`.
- Le référentiel Amazon ECR existe et l'utilisateur dispose d'un accès lui permettant d'extraire des images du référentiel.

Une fois ces conditions remplies, vous pouvez extraire l'image. Pour extraire votre exemple d'image d'Amazon ECR, exécutez la commande suivante :

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository:latest
```

Sortie :

```
latest: Pulling from hello-repository
0a85502c06c9: Pull complete
0998bf8fb9e9: Pull complete
a6785352b25c: Pull complete
e9ae3c220b23: Pull complete
Digest: sha256:215d7e4121b30157d8839e81c4e0912606fca105775bb0636EXAMPLE
Status: Downloaded newer image for aws_account_id.dkr.region.amazonaws.com/hello-
repository:latest
```

Étape 6 : Supprimer une image

Si vous n'avez plus besoin d'une image dans l'un de vos référentiels, vous pouvez la supprimer. Pour supprimer une image, spécifiez le référentiel dans lequel elle se trouve et une `imageDigest` valeur `imageTag` ou pour l'image. L'exemple suivant supprime une image du `hello-repository` référentiel avec la balise `latest` image. Pour supprimer votre exemple d'image du référentiel, exécutez la commande suivante :

```
aws ecr batch-delete-image \  
  --repository-name hello-repository \  
  --image-ids imageTag=latest \  
  --region region
```

Étape 7 : Supprimer un référentiel

Si vous n'avez plus besoin d'un référentiel complet d'images, vous pouvez le supprimer. L'exemple suivant utilise l'option `--force` pour supprimer un référentiel contenant des images. Pour supprimer un référentiel et toutes les images qu'il contient, exécutez la commande suivante :

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

Optimisation des performances pour Amazon ECR

Vous pouvez utiliser les recommandations suivantes concernant les paramètres et les stratégies pour optimiser les performances lorsque vous utilisez Amazon ECR.

Utiliser Docker 1.10 et des versions ultérieures afin de bénéficier des chargements de couches simultanés

Les images Docker sont composées de couches qui constituent des étapes de création intermédiaires de l'image. Chaque ligne d'un fichier Docker crée une nouvelle couche. Lorsque vous utilisez Docker 1.10 et les versions ultérieures, Docker transmet par défaut autant de couches qu'il y a eu de chargements dans Amazon ECR, ce qui permet d'accélérer les temps de chargement.

Utiliser une image de base plus petite

Les images par défaut disponibles via Docker Hub peuvent contenir de nombreuses dépendances dont votre application n'a pas besoin. Nous vous conseillons d'utiliser une image plus petite créée et gérée par d'autres personnes de la communauté Docker, ou de créer votre propre image de base à l'aide d'une image de Docker réduite au minimum. Pour en savoir plus, consultez [Créer une image de base](#) dans la documentation Docker.

Placement des dépendances qui changent le moins plus tôt dans votre fichier Docker

Docker met en cache des couches, ce qui accélère les temps de création. S'il n'y a eu aucun changement sur une couche depuis la dernière création, Docker utilisera la version mise en cache au lieu de recréer la couche. Toutefois, chaque couche est dépendante des couches précédentes. Si une couche change, Docker recompilera non seulement cette couche, mais également toutes les couches ultérieures.

Afin de réduire le temps nécessaire à la recréation d'un fichier Docker et au rechargement des couches, pensez à placer les dépendances qui changent le moins souvent plus tôt dans votre fichier Docker. Placez les dépendances qui changent rapidement (par exemple, le code source de votre application) plus tard dans la pile.

Chaîner des commandes pour éviter le stockage de fichier inutile

Les fichiers intermédiaires créés sur une couche continuent à faire partie de cette couche, même s'ils sont supprimés dans une couche ultérieure. Prenez l'exemple suivant :

```
WORKDIR /tmp
```

```
RUN wget http://example.com/software.tar.gz
RUN wget tar -xvf software.tar.gz
RUN mv software/binary /opt/bin/myapp
RUN rm software.tar.gz
```

Dans cet exemple, les couches créées par la première et la deuxième commande RUN contiennent le fichier .tar.gz d'origine et l'ensemble de son contenu non compressé. Et ce, même si le fichier .tar.gz est supprimé par la quatrième commande RUN. Ces commandes peuvent être regroupées dans une seule instruction RUN afin d'éviter que ces fichiers inutiles fassent partie de l'image Docker finale :

```
WORKDIR /tmp
RUN wget http://example.com/software.tar.gz &&\
    wget tar -xvf software.tar.gz &&\
    mv software/binary /opt/bin/myapp &&\
    rm software.tar.gz
```

Utiliser le point de terminaison régional le plus proche

Vous pouvez réduire la latence d'extraction des images d'Amazon ECR en veillant à utiliser le point de terminaison régional le plus proche de l'emplacement d'exécution de votre application. Si votre application s'exécute sur une EC2 instance Amazon, vous pouvez utiliser le code shell suivant pour obtenir la région à partir de la zone de disponibilité de l'instance :

```
REGION=$(curl -s http://169.254.169.254/latest/meta-data/placement/availability-zone
|\
    sed -n 's/\(\d*\)[a-zA-Z]*$/\1/p')
```

La région peut être transmise aux AWS CLI commandes à l'aide du --region paramètre ou définie comme région par défaut pour un profil à l'aide de la aws configure commande. Vous pouvez également définir la région lorsque vous passez des appels à l'aide du AWS SDK. Pour en savoir plus , consultez la documentation du kit SDK correspondant au langage de programmation utilisé.

Envoyer des demandes aux registres Amazon ECR

Vous pouvez envoyer, extraire, supprimer, afficher et gérer des images OCI, des images Docker et des artefacts compatibles OCI dans les registres privés Amazon ECR en utilisant soit des points de terminaison IPv4 uniquement, soit des points de terminaison à double pile (et). IPv4 IPv6 Pour effectuer des demandes depuis des IPv4 réseaux, vous pouvez utiliser des endpoints ou des points de IPv4 terminaison. Pour effectuer des demandes depuis un IPv6 réseau, utilisez un point de terminaison à double pile. Pour plus d'informations sur l'envoi de demandes aux registres publics Amazon ECR à l'aide IPv4 de points de terminaison à double pile, consultez la section [Faire des demandes aux registres publics Amazon ECR](#). L'accès à Amazon ECR via IPv6 Amazon est gratuit. Pour plus d'informations sur les tarifs, consultez les [tarifs d'Amazon Elastic Container Registry](#).

Les points de terminaison Amazon ECR sont désignés par des attributs autres que la prise en charge des points de IPv4 terminaison uniquement ou des points de terminaison à double pile. Ces attributs peuvent inclure :

- Région — Chaque point de terminaison est spécifique à une région.
- Type — La sélection du point de terminaison varie selon que vous utilisez le AWS SDK ou des interfaces de ligne de commande Docker compatibles avec l'OCI-Compatible.
- Sécurité — Dans certaines régions, Amazon ECR propose des points de terminaison conformes à la norme FIPS. Pour plus d'informations sur la liste des points de terminaison Amazon ECR conformes à la norme FIPS, [consultez le Federal Information Processing Standard \(FIPS\) 140-3](#).

[Pour plus d'informations sur les points de terminaison de service pris en charge par IPv4 le client à double pile, Docker et OCI, qui gère les appels d'API Amazon ECR depuis la AWS CLI, consultez la section Points de terminaison de service. AWS SDKs](#)

Commencer à faire des demandes IPv6

Pour envoyer une demande à un registre Amazon ECR via IPv6, vous devez utiliser un point de terminaison à double pile. Avant d'accéder à un registre Amazon ECR IPv6, vérifiez les conditions suivantes :

- Votre client et votre réseau doivent vous aider IPv6.
- Amazon ECR prend en charge les types de demandes suivants : IPv6

- Demandes des clients OCI et Docker :

```
<registry-id>.dkr-ecr.<aws-region>.on.aws
```

- AWS Demandes d'API :

```
ecr.<aws-region>.api.aws
```

- Vous devez mettre à jour toutes les politiques Gestion des identités et des accès AWS (IAM) ou de registre qui utilisent le filtrage des adresses IP source pour inclure des plages d' IPv6 adresses. Pour de plus amples informations, veuillez consulter [Utilisation des IPv6 adresses dans les politiques IAM](#).
- Lorsque vous l'utilisez IPv6, les journaux d'accès au serveur affichent Remote IP les adresses au IPv6 format. Mettez à jour vos outils, scripts et logiciels existants pour analyser ces adresses IP IPv6 formatées.

Note

Si vous rencontrez des problèmes liés à la présence d' IPv6 adresses dans les fichiers journaux, contactez [AWS Support](#).

Test de compatibilité d'adresses IP

Si vous utilisez use Linux/Unix ou Mac OS X, vous pouvez vérifier si vous pouvez accéder à un point de terminaison à double pile en IPv6 utilisant la `curl` commande comme indiqué dans l'exemple suivant :

Exemple

```
curl --verbose https://ecr.us-west-2.api.aws
```

Les informations que vous obtenez doivent ressembler à celles de l'exemple ci-dessous. Si vous êtes connecté via IPv6 l'adresse IP connectée sera une IPv6 adresse.

```
* About to connect() to ecr.us-west-2.api.aws port 443 (#0)
* Trying IPv6 address... connected
* Connected to ecr.us-west-2.api.aws (IPv6 address) port 443 (#0)
> Host: ecr.us-west-2.api.aws
* Request completely sent off
```

Si vous utilisez Microsoft Windows 7 ou Windows 10, vous pouvez vérifier si vous pouvez accéder à un point de terminaison à double pile via IPv4 ou en IPv6 utilisant la ping commande comme indiqué dans l'exemple suivant.

```
ping ecr.us-west-2.api.aws
```

Effectuer des demandes à l'aide IPv6 de points de terminaison à double pile

Vous pouvez effectuer des appels d'API Amazon ECR à l' IPv6 aide de points de terminaison à double pile. Les fonctionnalités et les performances des opérations de l'API Amazon ECR restent cohérentes, que vous utilisiez IPv4 ou IPv6.

Lorsque vous utilisez le AWS Command Line Interface (AWS CLI) et AWS SDKs, vous pouvez l'activer IPv6 soit en utilisant un paramètre ou un indicateur pour passer à un point de terminaison à double pile, soit en spécifiant directement le point de terminaison à double pile dans votre fichier de configuration pour remplacer le point de terminaison Amazon ECR par défaut. Vous pouvez également apporter des modifications de configuration à l'aide d'une commande définie `use_dualstack_endpoint` sur `true` dans le profil par défaut. Pour plus d'informations `use_dualstack_endpoint`, voir Points de terminaison [à double pile et FIPS](#).

Exemple Modification de la configuration à l'aide d'une commande

```
aws configure set default.ecr.use_dualstack_endpoint true
```

Exemple Faire des demandes plutôt que IPv6 d'utiliser AWS CLI

```
aws ecr describe-repositories --region us-west-2 --endpoint-url https://  
ecr.us-west-2.api.aws
```

Utilisation des points de terminaison Amazon ECR depuis la CLI docker

Une fois connecté à votre référentiel Amazon ECR et étiqueté votre image, vous pouvez transférer et extraire des images OCI et des images Docker vers et depuis les registres Amazon ECR. Les exemples suivants illustrent les commandes docker push et docker pull avec les deux points de terminaison à double pile.

Exemple Transmission d'images docker à l'aide d'un point de terminaison IPv4

```
docker push <registry-id>.dkr.ecr.us-west-1.amazonaws.com/my-repository:tag
```

Exemple Transmission d'images docker à l'aide d'un point de terminaison à double pile

```
docker push <registry-id>.dkr-ecr.us-west-1.on.aws/my-repository:tag
```

Exemple Extraction d'images docker à l'aide d'un point de terminaison IPv4

```
docker pull <registry-id>.dkr.ecr.us-west-1.amazonaws.com/my-repository:tag
```

Exemple Extraction d'images docker à l'aide d'un point de terminaison à double pile

```
docker pull <registry-id>.dkr-ecr.us-west-1.on.aws/my-repository:tag
```

Utilisation des IPv6 adresses dans les politiques IAM

Avant d'accéder à un registre en utilisant IPv6, assurez-vous que votre utilisateur IAM et les politiques de registre Amazon ECR qui utilisent le filtrage des adresses IP incluent des plages d'IPv6 adresses. Si les politiques de filtrage des adresses IP ne sont pas mises à jour pour gérer IPv6 les adresses, les clients risquent de perdre ou d'accéder au registre par erreur lorsqu'ils commencent à l'utiliser IPv6. Pour de plus amples informations sur la gestion des autorisations d'accès avec IAM, veuillez consulter [Gestion des identités et des accès au registre de conteneur Amazon Elastic](#).

Les stratégies IAM qui permettent de filtrer les adresses IP utilisent des [opérateurs de condition d'adresse IP](#). L'exemple de politique de registre suivant montre comment identifier la 54.240.143.* plage d'IPv4 adresses autorisées à l'aide des opérateurs de condition d'adresse IP. Toutes les adresses IP situées en dehors de cette plage se voient refuser l'accès au registre (exempleregistry). Comme toutes les IPv6 adresses se situent en dehors de la plage autorisée, cette politique empêche les IPv6 adresses d'y accéder exempleregistry.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
```

```
"Principal": "*",
"Action": "ecr:*",
"Resource": "arn:aws:ecr:*:*:repository/exampleregistry/*",
"Condition": {
  "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
}
]
}
```

Pour autoriser à la fois les plages d'adresses IPv4 IPv6 (54.240.143.0/24 2001:DB8:1234:5678::/64) et (), modifiez l'élément Condition de la politique de registre comme indiqué dans l'exemple suivant. Vous pouvez utiliser ce format de Condition bloc pour mettre à jour vos politiques d'utilisateur et de registre IAM.

```
"Condition": {
  "IpAddress": {
    "aws:SourceIp": [
      "54.240.143.0/24",
      "2001:DB8:1234:5678::/64"
    ]
  }
}
```

Important

Avant de l'utiliser, IPv6 vous devez mettre à jour toutes les politiques d'utilisateur et de registre IAM pertinentes qui utilisent le filtrage des adresses IP. Nous ne recommandons pas d'utiliser le filtrage des adresses IP dans les politiques de registre.

Vous pouvez consulter vos politiques utilisateur IAM à l'aide de la console IAM à l'adresse. <https://console.aws.amazon.com/iam/> Pour de plus amples informations sur IAM, consultez le [Guide de l'utilisateur IAM](#).

Registre privé Amazon ECR

Un registre privé Amazon ECR héberge vos images de conteneur dans une architecture hautement disponible et évolutive. Vous pouvez utiliser votre registre pour gérer les référentiels d'images privés composés d'images et d'artefacts Docker et Open Container Initiative (OCI). Chaque compte AWS est fourni avec un registre privé Amazon ECR par défaut. Pour en savoir plus sur les registres publics Amazon ECR, consultez [Registres publics](#) dans le guide de l'utilisateur du registre de conteneur Amazon Elastic public.

Concepts de registre privé

- L'URL de votre registre privé par défaut est `https://aws_account_id.dkr.ecr.region.amazonaws.com`.
- Par défaut, votre compte dispose d'un accès en lecture et en écriture aux référentiels de votre registre privé. Cependant, les utilisateurs ont besoin d'autorisations pour appeler les API Amazon ECR et pour envoyer ou extraire des images vers et depuis vos référentiels privés. Amazon ECR fournit plusieurs politiques gérées pour contrôler l'accès des utilisateurs à différents niveaux. Pour de plus amples informations, veuillez consulter [Exemples de Identity-based politiques Amazon Elastic Container Registry](#).
- Vous devez authentifier votre client Docker auprès de votre registre privé afin de pouvoir utiliser les commandes docker push et docker pull pour transmettre et extraire les images vers et depuis les référentiels de ce registre. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).
- Les référentiels peuvent être contrôlés à l'aide de politiques d'accès utilisateur et de politiques de référentiel. Pour en savoir plus sur les politiques de référentiel, consultez [Politiques relatives aux référentiels privés dans Amazon ECR](#).
- Les référentiels de votre registre privé peuvent être répliqués entre AWS les régions de votre propre registre privé et sur des comptes distincts en configurant la réplication pour votre registre privé. Pour de plus amples informations, veuillez consulter [Réplication d'images privées sur Amazon ECR](#).

Authentification du registre privé dans Amazon ECR

Vous pouvez utiliser le AWS Management Console AWS CLI, le ou les AWS SDK pour créer et gérer des référentiels privés. Vous pouvez également utiliser ces méthodes pour exécuter certaines actions

sur les images, par exemple les répertorier sur une liste ou les supprimer. Ces clients utilisent des méthodes AWS d'authentification standard. Bien que vous puissiez utiliser l'API Amazon ECR pour transmettre et extraire des images, vous utiliserez probablement plus souvent la CLI Docker ou une bibliothèque Docker propre à un langage.

La CLI Docker ne prend pas en charge les méthodes d'authentification IAM natives. Des étapes supplémentaires sont nécessaires pour qu'Amazon ECR puisse authentifier et autoriser les demandes push et pull de Docker.

Vous trouverez dans les sections suivantes les méthodes d'authentification de registre détaillées.

Utilisation de l'assistant d'informations d'identification Amazon ECR

Amazon ECR propose un assistant d'informations d'identification Docker qui facilite le stockage et l'utilisation des informations d'identification Docker lors de la transmission ou l'extraction d'images vers Amazon ECR. Pour connaître les étapes d'installation et de configuration, consultez [Assistant d'informations d'identification Amazon ECR Docker](#).

Note

L'assistant d'informations d'identification Amazon ECR Docker ne prend pas en charge l'authentification multi-facteur (MFA) actuellement.

Utiliser un jeton d'autorisation

L'étendue d'autorisation d'un jeton d'autorisation correspond à celle du principal IAM utilisé pour récupérer le jeton d'authentification. Un jeton d'authentification est utilisé pour accéder à tout registre Amazon ECR auquel votre principal IAM a accès. Il est valide pendant 12 heures. Pour obtenir un jeton d'autorisation, vous devez utiliser l'opération [GetAuthorizationToken](#) API pour récupérer un jeton d'autorisation codé en base64 contenant le nom d'utilisateur AWS et un mot de passe codé. La AWS CLI `get-login-password` commande simplifie cela en récupérant et en décodant le jeton d'autorisation que vous pouvez ensuite rediriger vers une docker login commande d'authentification.

Pour authentifier Docker dans un registre privé Amazon ECR avec `get-login-password`

- Pour authentifier Docker dans un registre Amazon ECR avec `get-login-password`, exécutez la commande `aws ecr get-login-password`. Lorsque vous passez le jeton d'authentification à la commande `docker login`, utilisez la valeur AWS pour le nom d'utilisateur et spécifiez l'URI

de registre Amazon ECR auquel vous voulez vous authentifier. Si vous vous authentifiez sur plusieurs registres, vous devrez répéter la commande pour chacun d'eux.

Important

Si vous recevez une erreur, installez la dernière version de la CLI ou effectuez une mise à niveau vers cette version AWS CLI. Pour en savoir plus, consultez [Installer la AWS Command Line Interface](#) dans le guide de l'utilisateur AWS Command Line Interface .

- [get-login-password](#) (AWS CLI)

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

- [Get-ECRLoginCommand](#) (AWS Tools for Windows PowerShell)

```
(Get-ECRLoginCommand).Password | docker login --username AWS --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Utiliser l'authentification API HTTP

Amazon ECR prend en charge [l'API HTTP du registre Docker](#). Cependant, étant donné qu'Amazon ECR est un registre privé, vous devez fournir un jeton d'autorisation avec chaque demande HTTP. Vous pouvez ajouter un en-tête d'autorisation HTTP à l'aide de l'option `-H` pour curl et transmettre le jeton d'autorisation fourni par la `get-authorization-token` AWS CLI commande.

S'authentifier auprès de l'API HTTP Amazon ECR

1. Récupérez un jeton d'autorisation avec le AWS CLI et définissez-le sur une variable d'environnement.

```
TOKEN=$(aws ecr get-authorization-token --output text --query 'authorizationData[].authorizationToken')
```

2. Pour vous authentifier auprès de l'API, transmettez la variable `$TOKEN` à l'option `-H` de curl. Par exemple, la commande suivante répertorie les étiquettes d'image dans un référentiel Amazon

ECR. Pour en savoir plus, consultez la documentation de la référence [API HTTP du registre Docker](#).

```
curl -i -H "Authorization: Basic $TOKEN"  
https://aws_account_id.dkr.ecr.region.amazonaws.com/v2/amazonlinux/tags/list
```

La sortie est la suivante :

```
HTTP/1.1 200 OK  
Content-Type: text/plain; charset=utf-8  
Date: Thu, 04 Jan 2018 16:06:59 GMT  
Docker-Distribution-Api-Version: registry/2.0  
Content-Length: 50  
Connection: keep-alive  
  
{"name":"amazonlinux","tags":["2017.09","latest"]}
```

Paramètres du registre privé dans Amazon ECR

Amazon ECR utilise des paramètres de registre privés pour configurer les fonctions au niveau du registre. Les paramètres du registre privé sont configurés séparément pour chaque région. Vous pouvez utiliser des paramètres de registre privés pour configurer les fonctions suivantes.

- **Autorisations de registre** : une politique d'autorisations de registre permet de contrôler la réplication et d'extraire les autorisations de cache. Pour de plus amples informations, veuillez consulter [Autorisations de registre privé dans Amazon ECR](#).
- **Règles de cache d'extraction** : une règle de cache d'extraction est utilisée pour mettre en cache les images d'un registre en amont dans votre registre privé Amazon ECR. Pour de plus amples informations, veuillez consulter [Synchroniser un registre en amont avec un registre privé Amazon ECR](#).
- **Configuration de réplication** — La configuration de réplication est utilisée pour contrôler si vos référentiels sont copiés entre Régions AWS ou Comptes AWS. Pour de plus amples informations, veuillez consulter [Réplication d'images privées sur Amazon ECR](#).
- **Modèles de création de référentiels** — Un modèle de création de référentiel est utilisé pour définir les paramètres standard à appliquer lorsque de nouveaux référentiels sont créés par Amazon ECR en votre nom. Par exemple, les référentiels créés par une action de mise en cache par extraction, création en mode push ou réplication. Pour de plus amples informations, veuillez

consulter [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).

- Configuration de numérisation — Par défaut, votre registre est activé pour l'analyse de base. Vous pouvez activer l'analyse améliorée qui fournit un mode d'analyse automatisé et continu qui recherche les vulnérabilités des packages du système d'exploitation et du langage de programmation. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR](#).
- Pull-time exclusion de mise à jour : vous pouvez configurer des exclusions de mise à jour au moment de l'extraction afin d'empêcher que l'heure de la dernière extraction ne soit mise à jour pour des images spécifiques lors de leur extraction. Cela est utile pour les images utilisées à des CI/CD fins de test ou pour lesquelles vous ne souhaitez pas que le temps d'extraction ait une incidence sur les décisions relatives au cycle de vie. Pour de plus amples informations, veuillez consulter [Exclusions liées aux mises à jour instantanées](#).
- Configuration de montage de blob — La configuration de montage de blob est utilisée pour contrôler si les référentiels de votre registre partagent des couches communes plutôt que de stocker des couches dupliquées. Pour de plus amples informations, veuillez consulter [Montage de blob dans Amazon ECR](#).

Montage de blob dans Amazon ECR

Amazon ECR prend en charge une fonctionnalité appelée montage par blob pour partager des couches d'images communes entre les référentiels d'un registre. Lorsque cette option est activée, les référentiels d'un même registre peuvent référencer des couches provenant d'autres référentiels du même registre au lieu de stocker des copies dupliquées.

Lorsque le montage des blobs de registre est activé, Amazon ECR vérifie les couches existantes dans votre registre lors des opérations push lorsque les paramètres de montage sont inclus. Si une couche existe déjà dans un autre référentiel du même registre, Amazon ECR montera la couche existante au lieu d'en télécharger une copie.

Note

Les clients OCI incluent automatiquement les paramètres de montage s'ils détectent qu'un blob existe peut-être déjà dans un autre référentiel. Amazon ECR tente de procéder au montage uniquement lorsque ces paramètres sont présents dans la requête POST du client.

Concepts de montage Blob

- Le montage de blob ne fonctionne que dans le même registre (même compte et même région).
- Les référentiels doivent utiliser le même type de chiffrement et les mêmes clés.
- Le montage par blob n'est pas pris en charge pour les images créées via le cache d'extraction.
- Si vous décidez de désactiver le montage par blob, les images existantes qui ont été transférées avec le montage par blob configuré continueront de fonctionner et les couches resteront montées.

Configuration de montage du blob

Vous pouvez utiliser le AWS Management Console ou AWS CLI pour configurer le montage des blobs pour votre registre.

Note

Les utilisateurs doivent disposer de l'autorisation `ecr:GetDownloadUrlForLayer` IAM sur un référentiel pour monter des couches à partir de celui-ci.

AWS Management Console

Procédez comme suit pour mettre à jour la configuration de montage des blobs de votre registre à l'aide du AWS Management Console.

Activez la configuration de montage du blob pour votre registre privé

1. Ouvrez la console Amazon ECR à l'adresse <https://console.aws.amazon.com/ecr/private-registry/repositories>
2. Dans la barre de navigation, choisissez la région.
3. Dans le volet de navigation, choisissez Private registry, Feature & Settings, puis choisissez Blob mounting.
4. Sur la page de montage du blob, choisissez Activer.

Une bannière s'affiche indiquant que la configuration de montage du blob a été mise à jour pour être activée.

AWS CLI

Utilisez la commande suivante pour mettre à jour la configuration de montage des blobs de votre registre à l'aide du AWS CLI.

- ```
aws ecr put-account-setting --name BLOB_MOUNTING --value ENABLED
```

## Autorisations de registre privé dans Amazon ECR

Amazon ECR utilise une politique de registre pour accorder des autorisations à un principal AWS au niveau du registre privé.

Amazon ECR autorise toutes les actions ECR dans la politique et applique la politique de registre dans toutes les demandes ECR. Vous pouvez utiliser les politiques de registre pour accorder des autorisations pour des actions telles que la configuration de la réplication, la création de règles de cache pull-through et la création de référentiels. Pour obtenir la liste complète des actions d'API, consultez le [guide des API Amazon ECR](#). Pour plus d'informations sur les paramètres généraux de votre registre privé Amazon ECR, consultez [Paramètres du registre privé dans Amazon ECR](#).

### Note

Bien qu'il soit possible d'ajouter l'`ecr : *action` à une politique de registre privé, il est recommandé de n'ajouter que les actions spécifiques requises en fonction de la fonctionnalité que vous utilisez plutôt que d'utiliser un caractère générique.

## Rubriques

- [Exemples de politiques de registre privé pour Amazon ECR](#)
- [Octroi d'autorisations de registre pour la réplication entre comptes dans Amazon ECR](#)
- [Octroi d'autorisations de registre pour le cache d'extraction dans Amazon ECR](#)

## Exemples de politiques de registre privé pour Amazon ECR

Les exemples suivants illustrent des déclarations de politique que vous pouvez utiliser pour contrôler les autorisations octroyées aux utilisateurs sur votre registre Amazon ECR.

**Note**

Dans chaque exemple, si l'`ecr:CreateRepository` action est supprimée de votre politique de registre, la réplication peut toujours avoir lieu. Toutefois, pour une réplication réussie, vous devez créer des référentiels portant le même nom dans votre compte.

## Exemple : autoriser tous les principaux IAM d'un compte source à répliquer tous les référentiels

La politique d'autorisation de registre suivante permet à tous les principaux IAM (utilisateurs et rôles) d'un compte source de répliquer tous les référentiels.

Notez ce qui suit :

- **Important** : Lorsque vous spécifiez un Compte AWS ID en tant que principal dans une politique, vous accordez l'accès à tous les utilisateurs et rôles IAM au sein de ce compte, et pas uniquement à l'utilisateur root. Cela permet un accès étendu à l'ensemble du compte.
- **Considérations relatives à la sécurité** : les Account-level autorisations accordent l'accès à toutes les entités IAM du compte spécifié. Pour un accès plus restrictif, spécifiez des utilisateurs et des rôles IAM individuels ou utilisez des instructions de condition pour limiter davantage l'accès.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ReplicationAccessCrossAccount",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:root"
 },
 "Action": [
 "ecr:CreateRepository",
 "ecr:ReplicateImage"
],
 "Resource": [
 "arn:aws:ecr:us-west-2:444455556666:repository/*"
]
 }
]
}
```

```

]
 }
]
}

```

## Exemple : autoriser les principaux IAM à accéder à plusieurs comptes

La politique d'autorisation de registre suivante comporte deux déclarations. Chaque instruction permet à tous les principaux IAM (utilisateurs et rôles) d'un compte source de répliquer tous les référentiels.

JSON

```

{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ReplicationAccessCrossAccount1",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:root"
 },
 "Action": [
 "ecr:CreateRepository",
 "ecr:ReplicateImage"
],
 "Resource": [
 "arn:aws:ecr:us-west-2:123456789012:repository/*"
]
 },
 {
 "Sid": "ReplicationAccessCrossAccount2",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::444455556666:root"
 },
 "Action": [
 "ecr:CreateRepository",
 "ecr:ReplicateImage"
],
 "Resource": [

```

```

 "arn:aws:ecr:us-west-2:123456789012:repository/*"
]
}

```

Exemple : autorisez tous les principaux IAM d'un compte source à répliquer tous les référentiels avec un préfixe. **prod-**

La politique d'autorisation de registre suivante permet à tous les principaux IAM (utilisateurs et rôles) d'un compte source de répliquer tous les référentiels commençant par. **prod-**

JSON

```

{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ReplicationAccessCrossAccount",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:root"
 },
 "Action": [
 "ecr:CreateRepository",
 "ecr:ReplicateImage"
],
 "Resource": [
 "arn:aws:ecr:us-west-2:444455556666:repository/prod-*"
]
 }
]
}

```

## Octroi d'autorisations de registre pour la réplication entre comptes dans Amazon ECR

Le type de politique entre comptes est utilisé pour accorder des autorisations à un AWS principal, permettant ainsi la réplication des référentiels d'un registre source vers votre registre. Par défaut,

vous avez l'autorisation de configurer la réplication inter-régions dans votre propre registre. Vous devez uniquement configurer la politique de registre si vous accordez à un autre compte l'autorisation de répliquer du contenu dans votre registre.

Une politique de registre doit octroyer l'autorisation pour l'action d'API `ecr:ReplicateImage`. Cette API est une API Amazon ECR interne qui peut répliquer des images entre régions ou comptes. Vous pouvez également octroyer l'autorisation pour l'action `ecr:CreateRepository`, qui permet à Amazon ECR de créer des référentiels dans votre registre s'ils n'existent pas déjà. Si l'autorisation `ecr:CreateRepository` n'est pas octroyée, un référentiel portant le même nom que le référentiel source doit être créé manuellement dans votre registre. Si aucun des deux n'est fait, la réplication échouera. Tout échec `CreateRepository` ou toute action d'`ReplicateImageAPI` apparaît dans `CloudTrail`.

Pour configurer une politique d'autorisations pour la réplication (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région dans laquelle configurer votre politique de registre.
3. Dans le volet de navigation, choisissez Registre privé, Fonctionnalités et paramètres, puis Autorisations.
4. Dans la page Registry permissions (Autorisations de registre), choisissez Generate statement (Générer une instruction).
5. Effectuez les étapes suivantes pour définir votre déclaration de politique à l'aide du générateur de politique.
  - a. Pour Type de politique, choisissez Réplication - comptes croisés.
  - b. Pour Numéro de relevé, entrez un identifiant de relevé unique. Ce champ est utilisé comme Sid dans la politique de registre.
  - c. Pour Comptes, saisissez les ID de compte pour chaque compte auquel vous souhaitez octroyer des autorisations. Lorsque vous précisez plusieurs ID de compte, séparez-les par une virgule.
6. Choisissez Enregistrer.

Pour configurer une politique d'autorisations pour la réplication (AWS CLI)

1. Créez un fichier nommé `registry_policy.json` et remplissez-le avec une politique de registre.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ReplicationAccessCrossAccount",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:root"
 },
 "Action": [
 "ecr:CreateRepository",
 "ecr:ReplicateImage"
],
 "Resource": [
 "arn:aws:ecr:us-west-2:444455556666:repository/*"
]
 }
]
}
```

2. Créez la politique de registre à l'aide du fichier de politique.

```
aws ecr put-registry-policy \
 --policy-text file://registry_policy.json \
 --region us-west-2
```

3. Récupérez la politique de votre registre pour confirmer.

```
aws ecr get-registry-policy \
 --region us-west-2
```

## Octroi d'autorisations de registre pour le cache d'extraction dans Amazon ECR

Les autorisations du registre privé Amazon ECR peuvent être utilisées pour étendre les autorisations des entités IAM individuelles à utiliser la mise en cache par extraction. Si une entité IAM dispose

de plus d'autorisations accordées par une politique IAM que celles accordées par la politique d'autorisations de registre, la politique IAM a la priorité.

Pour créer une politique d'autorisations de registre privée (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/>l'adresse.
2. Dans la barre de navigation, choisissez la région dans laquelle vous souhaitez configurer votre instruction d'autorisations de registre privé.
3. Dans le volet de navigation, choisissez Registre privé, Fonctionnalités et paramètres, puis Autorisations.
4. Dans la page Registry permissions (Autorisations de registre), choisissez Generate statement (Générer une instruction).
5. Pour chaque instruction de politique d'autorisations de mise en cache par extraction que vous souhaitez créer, procédez comme suit.
  - a. Pour Policy type (Type de politique), choisissez Pull through cache policy (Politique de mise en cache par extraction).
  - b. Pour Statement id (ID d'instruction), fournissez un nom pour l'instruction de politique de mise en cache par extraction.
  - c. Pour IAM entities (Entités IAM), indiquez les utilisateurs, groupes ou rôles à inclure dans la politique.
  - d. Pour l'espace de noms du cache, sélectionnez la règle de cache d'extraction à laquelle associer la politique.
  - e. Pour Repository names (Noms de référentiel), spécifiez le nom de base du référentiel pour lequel appliquer la règle. Par exemple, si vous voulez spécifier le référentiel Amazon Linux sur Amazon ECR Public, le nom du référentiel sera `amazonlinux`.

# Référentiels privés Amazon ECR

Un référentiel privé Amazon ECR contient vos images Docker, vos images Open Container Initiative (OCI) et vos artefacts compatibles avec OCI. Vous pouvez créer, surveiller et supprimer des référentiels d'images et définir des autorisations qui contrôlent qui peut y accéder à l'aide des opérations de l'API Amazon ECR ou de la section Référentiels de la console Amazon ECR. Amazon ECR s'intègre également à la CLI Docker, afin que vous puissiez transférer et extraire des images de vos environnements de développement vers vos référentiels.

## Rubriques

- [Concepts de référentiel privé](#)
- [Création d'un référentiel privé Amazon ECR pour stocker des images](#)
- [Afficher le contenu et les détails d'un référentiel privé dans Amazon ECR](#)
- [Supprimer un dépôt privé dans Amazon ECR](#)
- [Politiques relatives aux référentiels privés dans Amazon ECR](#)
- [Marquage d'un référentiel privé dans Amazon ECR](#)

## Concepts de référentiel privé

- Par défaut, votre compte dispose d'un accès en lecture et en écriture aux référentiels de votre registre par défaut (`aws_account_id.dkr.ecr.region.amazonaws.com`). Toutefois, les utilisateurs ont besoin d'autorisations pour passer des appels vers Amazon ECR APIs et pour transférer ou extraire des images vers et depuis vos référentiels. Amazon ECR fournit plusieurs politiques gérées pour contrôler l'accès des utilisateurs à différents niveaux. Pour de plus amples informations, veuillez consulter [Exemples de Identity-based politiques Amazon Elastic Container Registry](#).
- Les référentiels peuvent être contrôlés à l'aide de politiques d'accès utilisateur et de politiques de référentiel. Pour de plus amples informations, veuillez consulter [Politiques relatives aux référentiels privés dans Amazon ECR](#).
- Les noms des référentiels peuvent comporter des espaces de noms, que vous pouvez utiliser pour regrouper des référentiels similaires. Par exemple, si plusieurs équipes utilisent le même registre, l'équipe A pourra utiliser l'espace de noms `team-a`, tandis que l'équipe B ajoutera l'espace de noms `team-b`. En faisant cela, chaque équipe a sa propre image appelée `web-app` avec chaque image préfacée avec l'espace de noms de l'équipe. Cette configuration permet que ces images de

chaque équipe puissent être utilisées simultanément sans interférence. L'image de l'équipe A est `team-a/web-app`, et l'image de l'équipe B est `team-b/web-app`.

- Vos images peuvent être répliquées vers d'autres référentiels dans les régions de votre propre registre et entre les comptes. Pour ce faire, indiquez une configuration de réplication dans les paramètres de votre registre. Pour de plus amples informations, veuillez consulter [Paramètres du registre privé dans Amazon ECR](#).
- Lorsque le montage blob est activé au niveau du registre, les référentiels peuvent partager des couches d'images communes.

## Création d'un référentiel privé Amazon ECR pour stocker des images

### Important

Le chiffrement double couche côté serveur avec AWS KMS (DSSE-KMS) n'est disponible que dans les régions. AWS GovCloud (US)

Créez un référentiel privé Amazon ECR, puis utilisez-le pour stocker les images de vos conteneurs. Suivez les étapes suivantes pour créer un référentiel privé à l'aide de la AWS Management Console.

### Créer un référentiel (AWS Management Console)


1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région dans laquelle vous souhaitez créer votre référentiel.
3. Choisissez Dépôts privés, puis sélectionnez Créer un référentiel.
4. Pour Nom du référentiel, saisissez un nom unique pour votre référentiel. Le nom du référentiel peut être spécifié seul (par exemple `nginx-web-app`). Alternativement, il peut être précédé par un espace de noms pour regrouper le référentiel dans une catégorie (par exemple `project-a/nginx-web-app`).

### Note

Le nom du référentiel peut contenir un maximum de 256 caractères. Le nom doit commencer par une lettre et peut uniquement contenir des lettres minuscules, des

chiffres, des traits d'union, des traits de soulignement, des points et des barres obliques. L'utilisation d'une double barre oblique n'est pas prise en charge.

5. Pour l'immuabilité des balises Image, choisissez l'un des paramètres de mutabilité des balises suivants pour le référentiel.
  - **Mutable** : choisissez cette option si vous souhaitez que les balises d'image soient remplacées. Recommandé pour les référentiels utilisant des actions de cache d'extraction afin de garantir qu'Amazon ECR puisse mettre à jour les images mises en cache. En outre, pour désactiver les mises à jour de balises pour quelques balises modifiables, entrez le nom des balises ou utilisez des caractères génériques (\*) pour faire correspondre plusieurs balises similaires dans la zone de texte d'exclusion des balises modifiables.
  - **Immuable** : choisissez cette option si vous souhaitez empêcher le remplacement des balises d'image. Elle s'applique à toutes les balises et exclusions du référentiel lors du transfert d'une image avec une balise existante. Amazon ECR renvoie un `ImageTagAlreadyExistsException` si vous tentez de publier une image avec une balise existante. En outre, pour activer les mises à jour des balises pour quelques balises immuables, entrez les noms des balises ou utilisez des caractères génériques (\*) pour faire correspondre plusieurs balises similaires dans la zone de texte Exclusion de balises immuable.

 Note

Les paramètres de mutabilité des balises individuelles ne sont pas pris en charge.

6. Pour la configuration du chiffrement, choisissez entre AES-256 ou. AWS KMS Pour de plus amples informations, veuillez consulter [Chiffrement au repos](#).
  - a. Si cette option AWS KMS est sélectionnée, choisissez entre le chiffrement monocouche et le chiffrement double couche. L'utilisation du chiffrement à double couche AWS KMS entraîne des frais supplémentaires. Pour plus d'informations, consultez la section [Amazon ECR Service Pricing](#).
  - b. Par défaut, la clé AWS gérée avec l'alias `aws/ecr` est choisie. Cette clé est créée dans votre compte la première fois que vous créez un référentiel avec AWS KMS le chiffrement activé. Sélectionnez Clé gérée par le client (avancée) pour choisir votre propre AWS KMS clé. La AWS KMS clé doit se trouver dans la même région que le cluster. Sélectionnez Créer une AWS KMS clé pour accéder à la AWS KMS console afin de créer votre propre clé.

7. En ce qui concerne les paramètres de numérisation d'images, vous pouvez définir les paramètres de numérisation au niveau du référentiel pour la numérisation de base, mais il est recommandé de spécifier la configuration de numérisation au niveau du registre privé. La configuration des paramètres de numérisation au niveau du registre privé vous permet de choisir entre une analyse améliorée ou une analyse de base, et vous permet également de définir des filtres pour spécifier les référentiels à analyser.
8. Choisissez Créer.

### Créer un référentiel (AWS CLI)

1. Vous pouvez créer un dépôt à l'aide de la `aws ecr create-repository` commande AWS CLI with.

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

2. Si vous avez défini un modèle de création de référentiel, vous pouvez créer un référentiel en diffusant votre image à l'aide des commandes push Amazon ECR habituelles avec le nom de référentiel souhaité. Amazon ECR créera automatiquement le référentiel pour vous en utilisant les paramètres prédéfinis de votre modèle de création de référentiel. Si aucun modèle de création de référentiel n'est encore défini, votre demande à votre référentiel d'images inexistant échouera.

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/prefix/my-new-repository:tag
```

## Étapes suivantes

Pour afficher les étapes à suivre pour transférer une image vers votre référentiel, sélectionnez le référentiel et choisissez Afficher les commandes push. Pour plus d'informations sur le transfert d'une image vers un référentiel, consultez [Transférer une image vers un référentiel privé Amazon ECR](#).

# Afficher le contenu et les détails d'un référentiel privé dans Amazon ECR

Après avoir créé un dépôt privé, vous pouvez consulter les informations le concernant dans AWS Management Console :

- Images stockées dans un référentiel
- Les informations sur chaque image stockée dans le référentiel, y compris la taille et le résumé SHA de chaque image
- La fréquence d'analyse spécifiée pour le contenu du référentiel
- Indique si le référentiel est associé à une règle de cache par extraction active
- Les paramètres de chiffrement pour le référentiel

## Note

Depuis la version 1.9 de Docker, le client Docker compresse les couches d'images avant de les transmettre à un registre Docker V2. La sortie de la commande `docker images` affiche la taille de l'image non compressée. Par conséquent, n'oubliez pas que Docker peut renvoyer une image plus grande que l'image montrée dans la AWS Management Console.

## Afficher les informations du référentiel (AWS Management Console)

1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel à afficher.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Dans la page Repositories (Référentiels), choisissez l'onglet Private (Privé), puis choisissez le référentiel à afficher.
5. Sur la page des informations sur le référentiel, la console affiche par défaut la vue Images. Utilisez le menu de navigation pour afficher d'autres informations sur le référentiel.
  - Choisissez Summary (Récapitulatif) pour afficher les détails du référentiel et les données de comptage d'extraction du référentiel.
  - Choisissez Images pour afficher les informations relatives aux balises des images du référentiel. Pour afficher des informations supplémentaires sur l'image, sélectionnez la

balise de l'image. Pour de plus amples informations, veuillez consulter [Afficher les détails d'une image dans Amazon ECR](#).

Si vous souhaitez supprimer des images non étiquetées, vous pouvez cocher la case à gauche des référentiels à supprimer, puis choisir Supprimer. Pour de plus amples informations, veuillez consulter [Supprimer une image dans Amazon ECR](#).

- Choisissez Autorisations pour afficher les politiques de référentiel qui sont appliquées au référentiel. Pour de plus amples informations, veuillez consulter [Politiques relatives aux référentiels privés dans Amazon ECR](#).
- Choisissez Politique de cycle de vie pour afficher les règles de politique de cycle de vie qui sont appliquées au référentiel, ainsi que l'historique des événements du cycle de vie. Pour de plus amples informations, veuillez consulter [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).
- Choisissez Étiquettes pour afficher les étiquettes de métadonnées appliquées au référentiel.

## Supprimer un dépôt privé dans Amazon ECR

Si vous n'avez plus besoin d'un référentiel, vous pouvez le supprimer. Lorsque vous supprimez un référentiel dans le AWS Management Console, toutes les images qu'il contient sont également supprimées ; cela ne peut pas être annulé.

### Important

Les images des référentiels supprimés sont également supprimées. Vous ne pouvez pas annuler cette opération.

### Supprimer un référentiel (AWS Management Console)

1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel à supprimer.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Dans la page Repositories (Référentiels), choisissez la page Private (Privé), puis sélectionnez le référentiel à supprimer et choisissez Delete (Supprimer).
5. Dans la *repository\_name* fenêtre Supprimer, vérifiez que les référentiels sélectionnés doivent être supprimés et choisissez Supprimer.

# Politiques relatives aux référentiels privés dans Amazon ECR

Amazon ECR utilise les autorisations basées sur les ressources pour contrôler l'accès aux référentiels. Les autorisations basées sur les ressources vous permettent de spécifier quels utilisateurs ou rôles ont accès à un référentiel et quelles actions ils peuvent effectuer sur le référentiel. Par défaut, seul le AWS compte qui a créé le référentiel a accès au référentiel. Vous pouvez appliquer une politique de dépôt qui autorise un accès supplémentaire à votre dépôt.

## Rubriques

- [Politiques de référentiel vs politiques IAM](#)
- [Exemples de politiques relatives aux référentiels privés dans Amazon ECR](#)
- [Définition d'une déclaration de politique de dépôt privé dans Amazon ECR](#)

## Politiques de référentiel vs politiques IAM

Les politiques de référentiel Amazon ECR constituent un sous-ensemble de politiques IAM spécifiquement limitées et utilisées pour contrôler l'accès aux référentiels Amazon ECR individuels. Les politiques IAM sont généralement utilisées pour appliquer des autorisations pour l'ensemble du service Amazon ECR, mais elles peuvent également être utilisées pour contrôler l'accès à des ressources spécifiques.

Les politiques de référentiel Amazon ECR et les politiques IAM sont utilisées pour déterminer les actions qu'un utilisateur ou un rôle particulier peut effectuer sur un référentiel. Si un utilisateur ou un rôle est autorisé à effectuer une action via une politique de référentiel, mais se voit refuser l'autorisation via une politique IAM (ou inversement), l'action sera refusée. Un utilisateur ou un rôle doit être autorisé à effectuer une action par une politique de référentiel ou par une stratégie IAM, mais pas par les deux, pour que l'action soit autorisée.

### Important

Amazon ECR exige que les utilisateurs aient l'autorisation d'effectuer des appels d'API `ecr:GetAuthorizationToken` via une politique IAM avant qu'ils puissent s'authentifier auprès d'un référentiel et transmettre ou extraire des images à partir d'un référentiel Amazon ECR. Amazon ECR fournit plusieurs politiques IAM gérées pour contrôler l'accès des utilisateurs à différents niveaux. Pour de plus amples informations, veuillez consulter [Exemples de Identity-based politiques Amazon Elastic Container Registry](#).

Vous pouvez utiliser l'un ou l'autre de ces types de politique pour contrôler l'accès à vos référentiels, comme illustré dans les exemples suivants.

Cet exemple présente une politique de référentiel Amazon ECR qui autorise un utilisateur spécifique à décrire le référentiel et les images de ce référentiel.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ECRRepositoryPolicy",
 "Effect": "Allow",
 "Principal": {"AWS": "arn:aws:iam::111122223333:user/username"},
 "Action": [
 "ecr:DescribeImages",
 "ecr:DescribeRepositories"
],
 "Resource": "*"
 }
]
}
```

Cet exemple présente une politique IAM qui permet d'atteindre le même objectif que ci-dessus en délimitant la politique à un référentiel (spécifié par l'ARN complet du référentiel) à l'aide du paramètre de ressource. Pour en savoir plus sur le format Amazon Resource Name (ARN), consultez [Ressources](#).

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "AllowDescribeRepoImage",
 "Effect": "Allow",
 "Action": [
 "ecr:DescribeImages",
 "ecr:DescribeRepositories"
]
 }
]
}
```

```
],
 "Resource": ["arn:aws:ecr:us-
east-1:111122223333:repository/repository-name"]
 }
]
}
```

## Exemples de politiques relatives aux référentiels privés dans Amazon ECR

### Important

Les exemples de politique de référentiel présentés sur cette page sont destinés à être appliqués aux référentiels privés Amazon ECR. Ils ne fonctionneront pas correctement s'ils sont utilisés directement avec un principal IAM, sauf s'ils sont modifiés pour spécifier le référentiel Amazon ECR comme ressource. Pour plus d'informations sur les paramètres des politiques de référentiel, consultez [Définition d'une déclaration de politique de dépôt privé dans Amazon ECR](#).

Les politiques de référentiel Amazon ECR constituent un sous-ensemble de politiques IAM spécifiquement limitées et utilisées pour contrôler l'accès aux référentiels Amazon ECR individuels. Les politiques IAM sont généralement utilisées pour appliquer des autorisations pour l'ensemble du service Amazon ECR, mais elles peuvent également être utilisées pour contrôler l'accès à des ressources spécifiques. Pour de plus amples informations, veuillez consulter [Politiques de référentiel vs politiques IAM](#).

Les exemples de politique de référentiel suivants illustrent des déclarations d'autorisation que vous pouvez utiliser pour contrôler l'accès à vos référentiels privés Amazon ECR.

### Important

Amazon ECR exige que les utilisateurs aient l'autorisation d'effectuer des appels d'API `ecr:GetAuthorizationToken` via une politique IAM avant qu'ils puissent s'authentifier auprès d'un référentiel et transmettre ou extraire des images à partir d'un référentiel Amazon ECR. Amazon ECR fournit plusieurs politiques IAM gérées pour contrôler l'accès des utilisateurs à différents niveaux. Pour de plus amples informations, veuillez consulter [Exemples de Identity-based politiques Amazon Elastic Container Registry](#).

## Exemple : Autoriser un ou plusieurs utilisateurs

La politique de référentiel suivante autorise un ou plusieurs utilisateurs à transmettre et extraire des images vers et depuis un référentiel.

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "AllowPushPull",
 "Effect": "Allow",
 "Principal": {
 "AWS": [
 "arn:aws:iam::111122223333:user/push-pull-user-1",
 "arn:aws:iam::111122223333:user/push-pull-user-2"
]
 },
 "Action": [
 "ecr:BatchGetImage",
 "ecr:BatchCheckLayerAvailability",
 "ecr:CompleteLayerUpload",
 "ecr:GetDownloadUrlForLayer",
 "ecr:InitiateLayerUpload",
 "ecr:PutImage",
 "ecr:UploadLayerPart"
],
 "Resource": "*"
 }
]
}
```

## Exemple : autoriser un autre compte

La politique de référentiel suivante autorise un compte spécifique à transmettre des images.

**⚠ Important**

Le compte auquel vous octroyez des autorisations doit avoir activé la région dans laquelle vous créez la politique de référentiel, sinon une erreur se produira.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "AllowCrossAccountPush",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:root"
 },
 "Action": [
 "ecr:BatchCheckLayerAvailability",
 "ecr:CompleteLayerUpload",
 "ecr:InitiateLayerUpload",
 "ecr:PutImage",
 "ecr:UploadLayerPart"
],
 "Resource": "*"
 }
]
}
```

La politique de dépôt suivante permet à certains utilisateurs d'extraire des images (*pull-user-1* et *pull-user-2*) tout en fournissant un accès complet à une autre (*admin-user*).

**i Note**

Pour les politiques de référentiel plus complexes qui ne sont actuellement pas prises en charge dans le AWS Management Console, vous pouvez appliquer la politique à l'aide de la [set-repository-policy](#) AWS CLI commande.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "AllowPull",
 "Effect": "Allow",
 "Principal": {
 "AWS": [
 "arn:aws:iam::111122223333:user/pull-user-1",
 "arn:aws:iam::111122223333:user/pull-user-2"
]
 },
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer"
],
 "Resource": "*"
 },
 {
 "Sid": "AllowAll",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:user/admin-user"
 },
 "Action": [
 "ecr:*"
],
 "Resource": "*"
 }
]
}
```

## Exemple : Refuser tout

La politique de référentiel suivante permet à tous les utilisateurs de tous les comptes d'extraire des images.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "DenyPull",
 "Effect": "Deny",
 "Principal": "*",
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer"
],
 "Resource": "*"
 }
]
}
```

## Exemple : Restriction de l'accès à des adresses IP spécifiques

L'exemple suivant refuse des autorisations à tout utilisateur souhaitant effectuer des opérations Amazon ECR appliquées à un référentiel à partir d'une plage d'adresses spécifique.

La condition contenue dans cette déclaration identifie la 54.240.143.\* plage d'adresses IP autorisées du protocole Internet version 4 (IPv4).

Le Condition bloc utilise les NotIpAddress conditions et la clé de aws:SourceIp condition, qui est une clé AWS de condition étendue. Pour obtenir plus d'informations sur les clés de condition, consultez la section [Clés de contexte de condition globale AWS](#). 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": "ECRPolicyId1",
 "Statement": [
 {
 "Sid": "IPAllow",
```

```

 "Effect": "Deny",
 "Principal": "*",
 "Action": "ecr:*",
 "Resource": "*",
 "Condition": {
 "NotIpAddress": {
 "aws:SourceIp": "54.240.143.0/24"
 }
 }
 }
]
}

```

## Exemple : autoriser un AWS service

La politique de référentiel suivante autorise l' AWS CodeBuild accès aux actions de l'API Amazon ECR nécessaires à l'intégration avec ce service. Lorsque vous utilisez l'exemple suivant, vous devez utiliser les clés de condition `aws:SourceArn` et `aws:SourceAccount` pour définir quelles ressources peuvent assumer ces autorisations. Pour plus d'informations, consultez l'[exemple Amazon ECR CodeBuild](#) dans le guide de l'AWS CodeBuild utilisateur.

## JSON

```

{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "CodeBuildAccess",
 "Effect": "Allow",
 "Principal": {
 "Service": "codebuild.amazonaws.com"
 },
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer"
],
 "Resource": "*",
 "Condition": {
 "ArnLike": {
 "aws:SourceArn": "arn:aws:codebuild:us-east-1:123456789012:project/project-name"
 }
 }
 }
]
}

```

```
 },
 "StringEquals":{
 "aws:SourceAccount":"123456789012"
 }
 }
]
}
```

## Définition d'une déclaration de politique de dépôt privé dans Amazon ECR


Vous pouvez ajouter une déclaration de politique d'accès à un référentiel dans le AWS Management Console en suivant les étapes ci-dessous. Vous pouvez ajouter plusieurs instructions de politique par référentiel. Pour obtenir des exemples de politiques, consultez [Exemples de politiques relatives aux référentiels privés dans Amazon ECR](#).

### Important

Amazon ECR exige que les utilisateurs aient l'autorisation d'effectuer des appels d'API `ecr:GetAuthorizationToken` via une politique IAM avant qu'ils puissent s'authentifier auprès d'un référentiel et transmettre ou extraire des images à partir d'un référentiel Amazon ECR. Amazon ECR fournit plusieurs politiques IAM gérées pour contrôler l'accès des utilisateurs à différents niveaux. Pour de plus amples informations, veuillez consulter [Exemples de Identity-based politiques Amazon Elastic Container Registry](#).


### Définir une instruction de politique de référentiel

1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel pour lequel vous souhaitez définir une instruction de politique.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Dans la page Référentiels, choisissez le référentiel pour lequel vous souhaitez définir une instruction de politique pour afficher le contenu du référentiel.
5. Dans la vue de liste des images du référentiel, dans le panneau de navigation, sélectionnez Autorisations, Modifier.

 Note


Si vous ne voyez pas l'option Autorisations dans le panneau de navigation, vérifiez que vous êtes dans la vue de la liste des images du référentiel.

6. Dans la page Modifier des autorisations, choisissez Ajouter une instruction.
7. Pour Nom d'instruction, saisissez un nom pour l'instruction.
8. Pour Effet, choisissez si l'instruction de politique entraînera une autorisation ou un refus explicite.
9. Pour principal, choisissez l'étendue à laquelle s'applique l'instruction de politique. Pour en savoir plus, consultez [AWS Éléments de politique JSON : principal](#) dans le guide de l'utilisateur IAM.
  - Vous pouvez appliquer la déclaration à tous les AWS utilisateurs authentifiés en cochant la case Tout le monde (\*).
  - Pour principal du service, indiquez le nom du principal du service (par exemple, ecs.amazonaws.com) pour appliquer l'instruction à un service particulier.
  - Dans AWS Compte IDs, spécifiez un numéro de AWS compte (par exemple, 111122223333) pour appliquer le relevé à tous les utilisateurs d'un AWS compte spécifique. Il est possible de préciser plusieurs comptes à l'aide d'une liste séparée par des virgules.

 Important

Le compte auquel vous octroyez des autorisations doit avoir activé la région dans laquelle vous créez la politique de référentiel, sinon une erreur se produira.

- Pour les entités IAM, sélectionnez les rôles ou les utilisateurs de votre AWS compte auxquels appliquer la déclaration.

 Note

Pour les politiques de référentiel plus complexes qui ne sont actuellement pas prises en charge dans le AWS Management Console, vous pouvez appliquer la politique à l'aide de la [set-repository-policy](#) AWS CLI commande.

10. Pour Actions, choisissez l'étendue des opérations d'API Amazon ECR auxquelles l'instruction de politique doit s'appliquer dans la liste des opérations d'API individuelles.
11. Lorsque vous aurez terminé, choisissez Enregistrer pour définir la politique.

12. Répétez l'étape précédente pour chaque politique de référentiel à ajouter.

## Marquage d'un référentiel privé dans Amazon ECR

Pour vous aider à gérer vos référentiels Amazon ECR, vous pouvez attribuer vos propres métadonnées à des référentiels Amazon ECR nouveaux ou existants à l'aide de balises de ressource. AWS Par exemple, vous pouvez définir un ensemble d'identifications pour les référentiels Amazon ECR de votre compte qui vous aide à suivre le propriétaire de chaque référentiel.

### Principes de base des étiquettes

Les balises n'ont pas de signification sémantique pour Amazon ECR et sont interprétées strictement comme des chaînes de caractères. Les balises ne sont pas automatiquement affectées à vos ressources. Vous pouvez modifier les clés et valeurs de balise, et vous pouvez retirer des balises d'une ressource à tout moment. Vous pouvez définir la valeur d'une balise sur une chaîne vide, mais vous ne pouvez pas définir la valeur d'une balise sur null. Si vous ajoutez une balise ayant la même clé qu'une balise existante sur cette ressource, la nouvelle valeur remplace l'ancienne valeur. Si vous supprimez une ressource, ses balises sont également supprimées.

Vous pouvez travailler avec des balises à l'aide de la console Amazon ECR, du AWS CLI, et de l'API Amazon ECR.

À l'aide de Gestion des identités et des accès AWS (IAM), vous pouvez contrôler quels utilisateurs de votre AWS compte sont autorisés à créer, modifier ou supprimer des tags. Pour plus d'informations sur les balises dans les politiques IAM, consultez [the section called "Utilisation du contrôle Tag-Based d'accès"](#).

### Identification de vos ressources pour facturation

Les balises que vous ajoutez à vos référentiels Amazon ECR sont utiles lorsque vous examinez l'allocation des coûts après les avoir activés dans votre rapport Coût et utilisation. Pour de plus amples informations, veuillez consulter [Rapports d'utilisation d'Amazon ECR](#).

Pour voir le coût de vos ressources combinées, vous pouvez organiser vos informations de facturation en fonction des ressources possédant les mêmes valeurs de clé d'étiquette. Par exemple, vous pouvez étiqueter plusieurs ressources avec un nom d'application spécifique, puis organiser vos informations de facturation pour afficher le coût total de cette application dans plusieurs services. Pour en savoir plus sur la configuration d'un rapport de répartition des coûts avec des étiquettes, consultez [Rapport d'allocation des coûts mensuel](#) dans le guide de l'utilisateur AWS Billing .

**Note**

Si vous venez d'activer la création de rapports, les données du mois en cours peuvent être consultées après 24 heures.

## Ajouter des balises à un référentiel privé dans Amazon ECR

Vous pouvez ajouter des balises à un dépôt privé.

Pour plus d'informations sur les noms et les meilleures pratiques relatives aux balises, consultez les sections [Limites et exigences en matière de dénomination](#) des balises et [Bonnes pratiques](#) du Guide de l'utilisateur AWS des ressources de balisage.

### Ajouter des balises à un référentiel (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Sur la page Référentiels, cochez la case en regard du référentiel que vous voulez baliser.
5. Dans le menu Action, sélectionnez Balises du référentiel.
6. Sur la page Balises du référentiel, sélectionnez Ajouter des balises, Ajouter une balise.
7. Sur la page Modifier les balises du référentiel, spécifiez la clé et la valeur de chaque balise, puis choisissez Enregistrer.

### Ajouter des balises à un référentiel (AWS CLI ou à une API)

Vous pouvez ajouter ou remplacer une ou plusieurs balises à l'aide de l'API AWS CLI ou d'une API.

- AWS CLI - [tag-ressource](#)
- Action de l'API - [TagResource](#)

Les exemples suivants montrent comment ajouter des balises à l'aide du AWS CLI.

#### Exemple 1 : étiqueter un dépôt

La commande suivante balise un référentiel.

```
aws ecr tag-resource \
 --resource-arn arn:aws:ecr:region:account_id:repository/repository_name \
 --tags Key=stack,Value=dev
```

Exemple 2 : étiqueter un dépôt avec plusieurs balises

La commande suivante ajoute trois balises à un référentiel.

```
aws ecr tag-resource \
 --resource-arn arn:aws:ecr:region:account_id:repository/repository_name \
 --tags Key=key1,Value=value1 Key=key2,Value=value2 Key=key3,Value=value3
```

Exemple 3 : Afficher la liste des balises d'un référentiel

La commande suivante répertorie les balises associées à un référentiel.

```
aws ecr list-tags-for-resource \
 --resource-arn arn:aws:ecr:region:account_id:repository/repository_name
```

Exemple 4 : Création d'un référentiel et ajout d'une balise

La commande suivante permet de créer un référentiel nommé `test-repo` et d'ajouter une balise avec la clé `team` et la valeur `devs`.

```
aws ecr create-repository \
 --repository-name test-repo \
 --tags Key=team,Value=devs
```

## Supprimer des balises d'un référentiel privé dans Amazon ECR

Vous pouvez supprimer des balises d'un dépôt privé.

Pour supprimer un tag d'un dépôt privé (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Sur la page Référentiels, cochez la case en regard du référentiel dont vous voulez supprimer une balise.
4. Dans le menu Action, sélectionnez Balises du référentiel.

5. Sur la page Balises du référentiel, sélectionnez Modifier.
6. Sur la page Modifier les balises du référentiel, sélectionnez Supprimer pour chaque balise que vous voulez supprimer, puis choisissez Enregistrer.

Pour supprimer un tag d'un dépôt privé (AWS CLI)

Vous pouvez supprimer une ou plusieurs balises à l'aide de l'API AWS CLI ou d'une API.

- AWS CLI - [untag-resource](#)
- Action de l'API - [UntagResource](#)

L'exemple suivant montre comment supprimer une balise d'un référentiel à l'aide du AWS CLI.

```
aws ecr untag-resource \
 --resource-arn arn:aws:ecr:region:account_id:repository/repository_name \
 --tag-keys tag_key
```

# Images privées dans Amazon ECR

Amazon ECR stocke les images Docker, les images Open Container Initiative (OCI) et les artefacts compatibles OCI dans des référentiels privés. Vous pouvez utiliser la CLI Docker pour transmettre et extraire les images des référentiels.

[Grâce à la prise en charge d'OCI v1.1 par Amazon ECR, vous pouvez stocker et gérer les artefacts de référence définis par l'API OCI Referrers.](#) Les artefacts incluent les signatures, la nomenclature logicielle (SBoMs), les diagrammes Helm, les résultats de numérisation et les attestations. Un ensemble d'artefacts pour une image de conteneur est transféré avec ce conteneur et stocké sous forme d'image séparée qui est considérée comme une image consommée pour votre référentiel.

Les [Suppression de signatures et d'autres artefacts d'un référentiel privé Amazon ECR](#) pages [Signer des images dans Amazon ECR](#) et fournissent des exemples d'utilisation des artefacts liés aux signatures. Pour plus d'informations sur la signature d'images de conteneurs, consultez [la section Signature d'images de conteneur](#) dans le guide du AWS Signer développeur.

## Rubriques

- [Transférer une image vers un référentiel privé Amazon ECR](#)
- [Suppression de signatures et d'autres artefacts d'un référentiel privé Amazon ECR](#)
- [Afficher les détails d'une image dans Amazon ECR](#)
- [Extraction d'une image vers votre environnement local à partir d'un référentiel privé Amazon ECR](#)
- [Extraction de l'image du conteneur Amazon Linux](#)
- [Supprimer une image dans Amazon ECR](#)
- [Archivage d'une image dans Amazon ECR](#)
- [Modifier le balisage d'une image dans Amazon ECR](#)
- [Empêcher le remplacement des balises d'image dans Amazon ECR](#)
- [Prise en charge du format de manifeste d'image de conteneur dans Amazon ECR](#)
- [Utiliser des images Amazon ECR avec Amazon ECS](#)
- [Utiliser des images Amazon ECR avec Amazon EKS](#)

# Transférer une image vers un référentiel privé Amazon ECR

Vous pouvez transférer vos images Docker, listes manifestes, images OCI (Open Container Initiative) et artefacts compatibles vers vos référentiels privés.

Amazon ECR fournit un moyen de répliquer vos images dans d'autres référentiels. En spécifiant une configuration de réplication dans les paramètres de votre registre privé, vous pouvez effectuer une réplication entre les régions de votre propre registre et sur différents comptes. Pour de plus amples informations, veuillez consulter [Paramètres du registre privé dans Amazon ECR](#).

## Note

Si vous publiez une image actuellement archivée, cette image sera automatiquement restaurée et supprimée de l'archive. Pour plus d'informations sur l'archivage et la restauration d'images, consultez [Archivage d'une image dans Amazon ECR](#).

Lorsque le montage des blobs de registre est activé et que les paramètres de montage sont inclus, Amazon ECR vérifie automatiquement les couches existantes dans votre registre lors des opérations push. Si une couche existe déjà dans un autre référentiel du même registre, Amazon ECR montera la couche existante au lieu d'en télécharger une copie. Pour de plus amples informations, veuillez consulter [Montage de blob dans Amazon ECR](#).

## Rubriques

- [Autorisations IAM pour transférer une image vers un référentiel privé Amazon ECR](#)
- [Transférer une image Docker vers un référentiel privé Amazon ECR](#)
- [Transmission d'une image multi-architecture vers un référentiel privé Amazon ECR](#)
- [Transférer un graphique de Helm vers un référentiel privé Amazon ECR](#)

## Autorisations IAM pour transférer une image vers un référentiel privé Amazon ECR

Les utilisateurs ont besoin d'autorisations IAM pour transférer des images vers les référentiels privés Amazon ECR. Conformément à la meilleure pratique consistant à accorder le moindre privilège, vous pouvez accorder l'accès à un référentiel spécifique. Vous pouvez également accorder l'accès à tous les référentiels.

Un utilisateur doit s'authentifier auprès de chaque registre Amazon ECR auquel il souhaite envoyer des images en demandant un jeton d'autorisation. Amazon ECR fournit plusieurs politiques AWS gérées pour contrôler l'accès des utilisateurs à différents niveaux. Pour de plus amples informations, veuillez consulter [AWS politiques gérées pour Amazon Elastic Container Registry](#).

Vous pouvez également créer vos propres politiques IAM. La politique IAM suivante accorde les autorisations requises pour transférer une image vers un référentiel spécifique. Pour limiter les autorisations pour un référentiel spécifique, utilisez le nom Amazon Resource Name (ARN) complet du référentiel.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "ecr:CompleteLayerUpload",
 "ecr:UploadLayerPart",
 "ecr:InitiateLayerUpload",
 "ecr:BatchCheckLayerAvailability",
 "ecr:PutImage",
 "ecr:BatchGetImage"
],
 "Resource": "arn:aws:ecr:us-
east-1:111122223333:repository/repository-name"
 },
 {
 "Effect": "Allow",
 "Action": "ecr:GetAuthorizationToken",
 "Resource": "*"
 }
]
}
```

La politique IAM suivante accorde les autorisations requises pour transférer une image vers tous les référentiels.

```
{
```

```
"Version": "2012-10-17",
"Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "ecr:CompleteLayerUpload",
 "ecr:GetAuthorizationToken",
 "ecr:UploadLayerPart",
 "ecr:InitiateLayerUpload",
 "ecr:BatchCheckLayerAvailability",
 "ecr:PutImage",
 "ecr:BatchGetImage"
],
 "Resource": "arn:aws:ecr:us-west-2:111122223333:repository/*"
 }
]
```

## Transférer une image Docker vers un référentiel privé Amazon ECR

Vous pouvez envoyer vos images de conteneur vers un référentiel Amazon ECR à l'aide de la commande docker push.

Amazon ECR prend également en charge la création et le transfert de listes de manifestes Docker utilisées pour les images multi-architectures. Pour plus d'informations, consultez [Transmission d'une image multi-architecture vers un référentiel privé Amazon ECR](#).

Pour transmettre une image Docker à un référentiel Amazon ECR

Le référentiel Amazon ECR doit exister avant que vous ne publiiez l'image, ou vous devez avoir défini un modèle de création de référentiel. Pour plus d'informations, consultez [Création d'un référentiel privé Amazon ECR pour stocker des images](#) et [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).

1. Authentifiez le client Docker auprès du registre Amazon ECR dans lequel vous prévoyez de transmettre votre image. Vous devez obtenir des jetons d'authentification pour chaque registre utilisé ; les jetons sont valides pendant 12 heures. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).

Pour authentifier Docker dans un registre Amazon ECR, exécutez la commande `aws ecr get-login-password`. Lorsque vous passez le jeton d'authentification à la commande `docker login`,

utilisez la valeur AWS pour le nom d'utilisateur et spécifiez l'URI de registre Amazon ECR auquel vous souhaitez vous authentifier. Si vous vous authentifiez sur plusieurs registres, vous devrez répéter la commande pour chacun d'eux.

### Important

Si vous recevez une erreur, installez la dernière version de la CLI ou effectuez une mise à niveau vers cette version AWS CLI. Pour plus d'informations, consultez [Installation d' AWS Command Line Interface](#) dans le Guide de l'utilisateur AWS Command Line Interface .

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

2. Si votre référentiel d'images n'existe pas encore dans le registre vers lequel vous souhaitez envoyer le transfert, et si un modèle de création de référentiel est défini, vous pouvez transférer votre image en utilisant le préfixe du modèle de création de votre référentiel et le nom de référentiel de votre choix. ECR créera automatiquement le référentiel pour vous en utilisant les paramètres prédéfinis de votre modèle de création de référentiel.

Si aucun modèle de création de dépôt correspondant n'est défini, vous devez créer un référentiel. Pour plus d'informations, consultez [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#) ou [Création d'un référentiel privé Amazon ECR pour stocker des images](#).

3. Identifiez l'image à transmettre. Exécutez la commande `docker images` afin d'afficher la liste des images du conteneur du système.

```
docker images
```

Vous pouvez identifier une image à l'aide de la `repository:tag` valeur ou de l'identifiant de l'image dans la sortie de commande qui en résulte.

4. Balisez l'image avec la combinaison registre Amazon ECR, référentiel et nom de balise d'image facultatif à utiliser. Le format de registre est `aws_account_id.dkr.ecr.region.amazonaws.com`. Le nom du référentiel doit correspondre au référentiel que vous avez créé pour l'image. Si vous omettez la balise de l'image, nous supposons que c'est `latest`.

L'exemple suivant balise une image locale avec `e9ae3c220b23`  
l'`aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:tag`.

```
docker tag e9ae3c220b23 aws_account_id.dkr.ecr.region.amazonaws.com/my-
repository:tag
```

5. Transmettez l'image à l'aide de la commande `docker push` :

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:tag
```

6. (Facultatif) Appliquez toute balise supplémentaire à l'image et transmettez ces balises à Amazon ECR en répétant [Step 4](#) et [Step 5](#).

## Transmission d'une image multi-architecture vers un référentiel privé Amazon ECR

Vous pouvez transférer des images multi-architectures vers un référentiel Amazon ECR en créant et en diffusant des listes de manifestes Docker. Une liste manifeste est une liste d'images créée en spécifiant un ou plusieurs noms d'image. Dans la plupart des cas, la liste des manifestes est créée à partir d'images ayant la même fonction mais correspondant à des systèmes d'exploitation ou à des architectures différents. La liste manifeste n'est pas obligatoire. Pour en savoir plus, consultez [manifeste docker](#).

Une liste manifeste peut être extraite ou référencée dans une définition de tâche Amazon ECS ou dans une spécification de pod Amazon EKS, comme d'autres images Amazon ECR.

### Conditions préalables

- Dans votre CLI Docker, activez les fonctionnalités expérimentales. Pour plus d'informations sur les fonctionnalités expérimentales, consultez la section [Fonctionnalités expérimentales](#) dans la documentation Docker.
- Le référentiel Amazon ECR doit exister avant que vous poussiez l'image. Pour de plus amples informations, veuillez consulter [the section called "Création d'un référentiel pour stocker des images"](#).
- Les images doivent être transférées vers votre dépôt avant de créer le manifeste Docker. Pour en savoir plus sur la création d'une image, consultez [Transférer une image Docker vers un référentiel privé Amazon ECR](#).

## Transmettre une image Docker multi-architecture vers un référentiel Amazon ECR

1. Authentifiez le client Docker auprès du registre Amazon ECR dans lequel vous prévoyez de transmettre votre image. Vous devez obtenir des jetons d'authentification pour chaque registre utilisé ; les jetons sont valides pendant 12 heures. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).

Pour authentifier Docker dans un registre Amazon ECR, exécutez la commande `aws ecr get-login-password`. Lorsque vous passez le jeton d'authentification à la commande `docker login`, utilisez la valeur AWS pour le nom d'utilisateur et spécifiez l'URI de registre Amazon ECR auquel vous souhaitez vous authentifier. Si vous vous authentifier sur plusieurs registres, vous devrez répéter la commande pour chacun d'eux.

### Important

Si vous recevez une erreur, installez la dernière version de la CLI ou effectuez une mise à niveau vers cette version AWS CLI. Pour en savoir plus, consultez [Installer la AWS Command Line Interface](#) dans le guide de l'utilisateur AWS Command Line Interface .

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

2. Répertoriez les images de votre référentiel, en confirmant les balises d'image.

```
aws ecr describe-images --repository-name my-repository
```

3. Créez la liste manifeste Docker. La commande `manifest create` vérifie que les images référencées se trouvent déjà dans votre référentiel et crée le manifeste localement.

```
docker manifest create aws_account_id.dkr.ecr.region.amazonaws.com/my-repository aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:image_one_tag aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:image_two
```

4. (Facultatif) Vérifiez la liste manifeste Docker. Cela vous permet de confirmer la taille et le résumé de chaque manifeste d'image référencé dans la liste manifeste.

```
docker manifest inspect aws_account_id.dkr.ecr.region.amazonaws.com/my-repository
```

5. Transmettez la liste manifeste Docker à votre référentiel Amazon ECR.

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

## Transférer un graphique de Helm vers un référentiel privé Amazon ECR

Vous pouvez transférer des artefacts de l'Open Container Initiative (OCI) vers un référentiel Amazon ECR. Pour voir un exemple de cette fonctionnalité, suivez les étapes ci-dessous pour transférer un graphique Helm vers Amazon ECR.

Pour plus d'informations sur l'utilisation de vos cartes Helm hébergées par Amazon ECR avec Amazon EKS, consultez [Installation d'un graphique Helm sur un cluster Amazon EKS](#).

### Envoyer les Charts de Helm à un référentiel Amazon ECR

1. Installez la dernière version du Helm client. Ces étapes ont été écrites à l'aide de la version Helm 3.18.6. Pour assurer la compatibilité avec les versions de Kubernetes prises en charge par Amazon EKS, utilisez Helm version 3.9 ou ultérieure. Pour en savoir plus, consultez [Installation Helm](#).
2. Pour créer les Charts de Helm de test, effectuez les étapes suivantes. Pour en savoir plus, consultez [Documents Helm – Prise en main](#).
  - a. Créer les Charts de Helm nommés `helm-test-chart`, puis effacez le contenu du répertoire `templates`.

```
helm create helm-test-chart
rm -rf helm-test-chart/templates/*
```

- b. Créez un ConfigMap dans le `templates` dossier.

```
cd helm-test-chart/templates
cat <<EOF > configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
 name: helm-test-chart-configmap
data:
 myvalue: "Hello World"
EOF
```

3. Empaquetez le graphique. La sortie contiendra le nom de fichier du graphique empaqueté que vous utilisez lorsque vous appuyez sur les Charts de Helm.

```
cd ../../
helm package helm-test-chart
```

#### Output

```
Successfully packaged chart and saved it to: /Users/username/helm-test-chart-0.1.0.tgz
```

4. Créez un référentiel pour stocker les Charts de Helm. Le nom de votre référentiel doit correspondre au nom que vous avez utilisé lors de la création des Charts de Helm à l'étape 2. Pour de plus amples informations, veuillez consulter [Création d'un référentiel privé Amazon ECR pour stocker des images](#).

```
aws ecr create-repository \
 --repository-name helm-test-chart \
 --region us-west-2
```

5. Authentifiez votre Helm client auprès du registre Amazon ECR dans lequel vous prévoyez de transmettre l'image. Vous devez obtenir des jetons d'authentification pour chaque registre utilisé ; les jetons sont valides pendant 12 heures. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).

```
aws ecr get-login-password \
 --region us-west-2 | helm registry login \
 --username AWS \
 --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

6. Poussez les Charts de Helm à l'aide de la commande `helm push`. La sortie doit inclure l'URI du référentiel Amazon ECR et le résumé SHA.

```
helm push helm-test-chart-0.1.0.tgz \
 oci://aws_account_id.dkr.ecr.region.amazonaws.com/
```

7. Décrivez les Charts de Helm.

```
aws ecr describe-images \
 --repository-name helm-test-chart \
 --region us-west-2
```

```
--region us-west-2
```

Dans la sortie, vérifiez que le paramètre `artifactMediaType` indique le type d'artefact approprié.

```
{
 "imageDetails": [
 {
 "registryId": "aws_account_id",
 "repositoryName": "helm-test-chart",
 "imageDigest":
"sha256:dd8aebdda7df991a0ffe0b3d6c0cf315fd582cd26f9755a347a52adEXAMPLE",
 "imageTags": [
 "0.1.0"
],
 "imageSizeInBytes": 1620,
 "imagePushedAt": "2021-09-23T11:39:30-05:00",
 "imageManifestMediaType": "application/vnd.oci.image.manifest.v1+json",
 "artifactMediaType": "application/vnd.cncf.helm.config.v1+json"
 }
]
}
```

8. (Facultatif) Pour des étapes supplémentaires, installez le Helm ConfigMap et commencez à utiliser Amazon EKS. Pour de plus amples informations, veuillez consulter [Installation d'un graphique Helm sur un cluster Amazon EKS](#).

## Suppression de signatures et d'autres artefacts d'un référentiel privé Amazon ECR

Vous pouvez utiliser le client ORAS pour répertorier et supprimer des signatures et d'autres artefacts de type référence d'un référentiel privé Amazon ECR. La suppression de signatures et d'autres artefacts de référence est similaire à la suppression d'une image (voir [Supprimer une image dans Amazon ECR](#)). Voici comment répertorier les artefacts et supprimer des signatures :

Pour gérer les artefacts d'image à l'aide de la CLI ORAS

1. Installez et configurez le client ORAS.

Pour plus d'informations sur l'installation et la configuration du client ORAS, consultez la section [Installation](#) dans la documentation ORAS.

2. Pour répertorier les artefacts disponibles pour une image Amazon ECR, utilisez `oras discover`, suivi du nom de l'image :

```
oras discover 111222333444.dkr.ecr.us-east-1.amazonaws.com/oci:helloWorld
```

La sortie doit être similaire à ceci : .

```
111222333444.dkr.ecr.us-east-1.amazonaws.com/
oci@sha256:88c0c54329bfdc1d94d6f58cd3fcb1226d46f58670f44a8c689cb3c9b37b6925
application/vnd.cnf.notary.signature
sha256:387c10c1598ee18aae81dcfc86d0d06d116e46461d1c3cda8927e69c48108c42
sha256:6527bcec87adf1d55460666183b9d0968b3cd4e4bc34602d485206a219851171
```

3. Pour supprimer une signature à l'aide de la CLI ORAS, dans l'exemple précédent, exécutez la commande suivante :

```
oras manifest delete 111222333444.dkr.ecr.us-east-1.amazonaws.com/
oci@sha256:387c10c1598ee18aae81dcfc86d0d06d116e46461d1c3cda8927e69c48108c42
```

La sortie doit être similaire à ceci : .

```
Are you sure you want to delete the manifest "111222333444.dkr.ecr.us-
east-1.amazonaws.com/
oci@sha256:387c10c1598ee18aae81dcfc86d0d06d116e46461d1c3cda8927e69c48108c42" and
all tags associated with it? [y/N] y
```

4. Appuyez sur `y`. L'artefact doit être supprimé.

Pour résoudre les problèmes liés à la suppression d'artefacts

Si la suppression d'une signature, telle que celle qui vient d'être affichée, échoue, une sortie similaire à la suivante apparaît.

```
Error response from registry: failed to delete 111222333444.dkr.ecr.us-
east-1.amazonaws.com/
oci@sha256:387c10c1598ee18aae81dcfc86d0d06d116e46461d1c3cda8927e69c48108c42:
unsupported: Requested image referenced by manifest list:
[sha256:005e2c97a6373e483799fa4ff29ac64a42dd10f08efcc166d6775f9b74943b5b]
```

Cet échec peut se produire lors de la suppression d'une image envoyée avant le lancement d'OCI 1.1. Comme indiqué dans l'erreur, vous devez supprimer le manifeste faisant référence à l'image avant de pouvoir supprimer l'image comme suit :

1. Pour supprimer le manifeste associé à la signature que vous souhaitez supprimer, tapez :

```
oras manifest delete 111222333444.dkr.ecr.us-east-1.amazonaws.com/
oci@sha256:005e2c97a6373e483799fa4ff29ac64a42dd10f08efcc166d6775f9b74943b5b
```

La sortie doit être similaire à ceci :

```
Are you sure you want to delete the manifest
"sha256:005e2c97a6373e483799fa4ff29ac64a42dd10f08efcc166d6775f9b74943b5b" and all
tags associated with it? [y/N] y
```

2. Appuyez sur y. Le manifeste doit être supprimé.
3. Une fois le manifeste disparu, vous pouvez supprimer la signature :

```
oras manifest delete 111222333444.dkr.ecr.us-east-1.amazonaws.com/
oci@sha256:387c10c1598ee18aae81dcfc86d0d06d116e46461d1c3cda8927e69c48108c42
```

La sortie doit ressembler à ceci. Appuyez sur y.

```
Are you sure you want to delete the manifest
"sha256:387c10c1598ee18aae81dcfc86d0d06d116e46461d1c3cda8927e69c48108c42" and all
tags associated with it? [y/N] y
Deleted [registry] 111222333444.dkr.ecr.us-east-1.amazonaws.com/
oci@sha256:387c10c1598ee18aae81dcfc86d0d06d116e46461d1c3cda8927e69c48108c42
```

4. Pour vérifier que la signature a été supprimée, tapez :

```
oras discover 111222333444.dkr.ecr.us-east-1.amazonaws.com/oci:helloworld
```

La sortie doit être similaire à ceci :

```
111222333444.dkr.ecr.us-east-1.amazonaws.com/
oci@sha256:88c0c54329bfdc1d94d6f58cd3fcb1226d46f58670f44a8c689cb3c9b37b6925
application/vnd.cncf.notary.signature
sha256:6527bcec87adf1d55460666183b9d0968b3cd4e4bc34602d485206a219851171
```

## Afficher les détails d'une image dans Amazon ECR

Une fois que vous avez transféré une image dans votre dépôt, vous pouvez consulter les informations la concernant. Les détails inclus sont les suivants :

- URI de l'image
- Étiquettes de l'image
- Type de média de l'artefact
- Type de manifeste de l'image
- État de l'analyse
- Taille des images en Mo
- Date de transmission de l'image dans le référentiel
- État de la réplication

### Afficher les détails de l'image (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/repositories/> l'adresse.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel contenant votre image.
3. Dans le volet de navigation, sous Registre privé, choisissez Référentiels.
4. Sur la page Référentiels privés, choisissez le référentiel à afficher.
5. Sur la *repository\_name* page Référentiels :, choisissez l'image dont vous souhaitez afficher les détails.

# Extraction d'une image vers votre environnement local à partir d'un référentiel privé Amazon ECR

Si vous souhaitez exécuter une image Docker disponible dans Amazon ECR, vous pouvez l'extraire et la transmettre à votre environnement local à l'aide de la commande `docker pull`. Vous pouvez le faire depuis votre registre par défaut ou depuis un registre associé à un autre AWS compte.

Pour utiliser une image Amazon ECR dans une définition de tâche Amazon ECS, consultez [Utiliser des images Amazon ECR avec Amazon ECS](#).

## Important

Vous ne pouvez pas extraire une image archivée. Les images archivées doivent être restaurées avant de pouvoir être extraites. Pour plus d'informations sur l'archivage et la restauration d'images, consultez [Archivage d'une image dans Amazon ECR](#).

## Important

Amazon ECR exige que les utilisateurs aient l'autorisation d'effectuer des appels d'API `ecr:GetAuthorizationToken` via une politique IAM avant qu'ils puissent s'authentifier auprès d'un référentiel et transmettre ou extraire des images à partir d'un référentiel Amazon ECR. Amazon ECR fournit plusieurs politiques AWS gérées pour contrôler l'accès des utilisateurs à différents niveaux. Pour plus d'informations sur les politiques AWS gérées pour Amazon ECR, consultez [AWS politiques gérées pour Amazon Elastic Container Registry](#).

Pour extraire une image Docker d'un référentiel Amazon ECR

1. Authentifiez votre client Docker auprès du registre Amazon ECR à partir duquel l'image doit être extraite. Vous devez obtenir des jetons d'authentification pour chaque registre utilisé ; les jetons sont valides pendant 12 heures. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).
2. (Facultatif) Identifiez l'image à extraire.
  - Vous pouvez consulter une liste des référentiels dans un registre avec la commande `aws ecr describe-repositories` :

```
aws ecr describe-repositories
```

L'exemple de registre ci-dessus possède un référentiel appelé `amazonlinux`.

- Vous pouvez décrire les images d'un référentiel à l'aide de la commande `aws ecr describe-images` :

```
aws ecr describe-images --repository-name amazonlinux
```

L'exemple de référentiel ci-dessus comporte une image balisée

en tant que `latest` et `2016.09`, avec le hachage d'image

```
sha256:f1d4ae3f7261a72e98c6ebefe9985cf10a0ea5bd762585a43e0700ed99863807.
```

3. Procédez à l'extraction de l'image à l'aide de la commande `docker pull`. Le format du nom de l'image doit être `registry/repository[:tag]` pour une extraction par balise ou `registry/repository[@digest]` pour une extraction par hachage.

```
docker pull aws_account_id.dkr.ecr.us-west-2.amazonaws.com/amazonlinux:latest
```

#### Important

Si vous recevez un `repository-url not found: does not exist or no pull access` d'erreur, il se peut que vous deviez authentifier votre client Docker auprès d'Amazon ECR. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).

## Extraction de l'image du conteneur Amazon Linux


L'image de conteneur Amazon Linux est créée à partir des mêmes composants logiciels que ceux inclus dans l'AMI Amazon Linux. L'image du conteneur Amazon Linux peut être utilisée dans n'importe quel environnement en tant qu'image de base pour les charges de travail Docker. Si vous utilisez l'AMI Amazon Linux pour des applications dans Amazon EC2, vous pouvez conteneuriser vos applications avec l'image du conteneur Amazon Linux.

Vous pouvez utiliser l'image du conteneur Amazon Linux dans votre environnement de développement local, puis pousser votre application à AWS utiliser Amazon ECS. Pour de plus amples informations, veuillez consulter [Utiliser des images Amazon ECR avec Amazon ECS](#).

L'image de conteneur Amazon Linux est disponible sur Amazon ECR Public et sur [Docker Hub](#). Pour obtenir de l'aide concernant l'image du conteneur Amazon Linux, rendez-vous sur les [forums des AWS développeurs](#).

Pour extraire l'image de conteneur Amazon Linux depuis Amazon ECR Public

1. Authentifiez votre client Docker dans votre registre Amazon Linux Public . Les jetons d'authentification sont valides pendant 12 heures. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).

 Note

Les commandes `ecr-public` sont disponibles dans la AWS CLI à partir de la version 1.18.1.187, mais nous vous recommandons d'utiliser la dernière version de l' AWS CLI. Pour en savoir plus, consultez [Installer la AWS Command Line Interface](#) dans le guide de l'utilisateur AWS Command Line Interface .

```
aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws
```

La sortie est la suivante :

```
Login succeeded
```

2. Procédez à l'extraction de l'image de conteneur Amazon Linux à l'aide de la commande `docker pull`. Pour afficher l'image du conteneur Amazon Linux dans la galerie publique Amazon ECR, consultez [Galerie publique Amazon ECR – amazonlinux](#).

```
docker pull public.ecr.aws/amazonlinux/amazonlinux:latest
```

3. (Facultatif) Exécutez le conteneur localement.

```
docker run -it public.ecr.aws/amazonlinux/amazonlinux /bin/bash
```

Pour extraire l'image de conteneur Amazon Linux de Docker Hub

1. Procédez à l'extraction de l'image de conteneur Amazon Linux à l'aide de la commande docker pull.

```
docker pull amazonlinux
```

2. (Facultatif) Exécutez le conteneur localement.

```
docker run -it amazonlinux:latest /bin/bash
```

## Supprimer une image dans Amazon ECR

Si vous n'avez plus besoin d'une image, vous pouvez la supprimer du référentiel. Si vous n'avez plus besoin d'un référentiel, vous pouvez le supprimer totalement, ainsi que les images qu'il contient. Pour de plus amples informations, veuillez consulter [Supprimer un dépôt privé dans Amazon ECR](#).

Comme alternative à la suppression manuelle des images, vous pouvez créer des politiques de cycle de vie des référentiels qui permettent un contrôle accru sur la gestion du cycle de vie des images dans vos référentiels. Les politiques de cycle de vie automatisent ce processus pour vous. Pour de plus amples informations, veuillez consulter [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).

### Note

Si votre référentiel contient un mélange d'images, dont certaines ont été publiées avant qu'Amazon ECR ne prenne en charge l'OCI v1.1, certaines signatures seront associées à des index d'images ou à des listes de manifestes pointant vers elles. Par conséquent, lorsque vous supprimez une image antérieure à OCI v1.1, vous devrez peut-être supprimer manuellement la liste des manifestes qui fait référence à l'image afin de supprimer l'artefact.

Supprimer une image (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/repositories> l'adresse.
2. Dans la barre de navigation, choisissez la région qui contient l'image à supprimer.
3. Dans le panneau de navigation, choisissez Référentiels.

4. Dans la page Référentiels, choisissez le référentiel qui contient l'image à supprimer.
5. Sur la *repository\_name* page Référentiels : cochez la case située à gauche de l'image à supprimer et choisissez Supprimer.
6. Dans la boîte de dialogue Supprimer image(s), vérifiez que les images sélectionnées sont bien celles qui doivent être supprimées, puis choisissez Supprimer.

### Supprimer une image (AWS CLI)

1. Répertoriez les images dans votre référentiel. Les images étiquetées auront à la fois un résumé d'image et une liste d'étiquettes associées. Les images non étiquetées n'auront qu'un résumé d'image.

```
aws ecr list-images \
 --repository-name my-repo
```

2. (Facultatif) Supprimez toutes les étiquettes non souhaitées pour l'image en spécifiant l'étiquette de l'image à supprimer. Lorsque vous aurez supprimé la dernière étiquette de l'image, cette dernière sera supprimée.

```
aws ecr batch-delete-image \
 --repository-name my-repo \
 --image-ids imageTag=tag1 imageTag=tag2
```

3. Supprimez une image étiquetée ou non étiquetée en spécifiant le résumé de l'image. Lorsque vous supprimez une image en référençant son hachage, l'image et toutes ses étiquettes sont supprimées.

```
aws ecr batch-delete-image \
 --repository-name my-repo \
 --image-ids imageDigest=sha256:4f70ef7a4d29e8c0c302b13e25962d8f7a0bd304EXAMPLE
```

Pour supprimer plusieurs images, vous pouvez spécifier plusieurs étiquettes d'image ou des résumés d'image dans la demande.

```
aws ecr batch-delete-image \
 --repository-name my-repo \
 --image-ids imageDigest=sha256:4f70ef7a4d29e8c0c302b13e25962d8f7a0bd304EXAMPLE
 imageDigest=sha256:f5t0e245ssffc302b13e25962d8f7a0bd304EXAMPLE
```

# Archivage d'une image dans Amazon ECR

## Qu'est-ce que la classe de stockage d'archives ECR ?

La classe de stockage d'archivage Amazon ECR est une nouvelle classe de stockage qui fournit un stockage à long terme et à faible coût pour les images de conteneurs. Amazon ECR propose deux classes de stockage :

- Classe de stockage standard ECR : classe de stockage par défaut pour les images actives régulièrement consultées.
- Classe de stockage d'archivage ECR : classe de stockage peu coûteuse pour les images rarement consultées mais qui doivent être conservées pour des raisons de conformité ou de référence à long terme. La classe de stockage d'archivage permet de réaliser des économies pour de grandes quantités d'images par rapport à la classe de stockage standard pour la conservation des images à long terme. Pour obtenir des informations détaillées sur les prix, consultez les [tarifs Amazon ECR](#).

Pour archiver des images, deux options s'offrent à vous. Tout d'abord, vous pouvez configurer des règles de cycle de vie pour archiver automatiquement les images en fonction des éléments suivants :

- Durée écoulée depuis le transfert de l'image
- Durée écoulée depuis la dernière extraction de l'image
- Nombre d'images dans le référentiel

Vous pouvez également configurer les paramètres pour supprimer définitivement les images une fois qu'elles ont été archivées pendant une période spécifiée. Pour plus d'informations, consultez la section [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).

Vous pouvez également archiver des images à l'aide de la console Amazon ECR ou AWS CLI. Pour plus d'informations, consultez la section [Archivage d'une image](#).

Lorsque vous devez réutiliser une image archivée, vous pouvez la restaurer dans la classe de stockage ECR Standard. Vous pouvez vous attendre à ce que l'ECR restaure l'image dans les 20 minutes. Les images restaurées se comportent comme des images récemment envoyées et peuvent être utilisées immédiatement une fois la restauration terminée. Les images restaurées sont soumises aux politiques de numérisation, de réplication et de cycle de vie du référentiel. Pour plus d'informations, consultez la section [Restaurer une image](#).

## Archivage d'une image

Vous pouvez archiver les images manuellement à l'aide de la console Amazon ECR ou AWS CLI automatiquement à l'aide des politiques de cycle de vie. Lorsqu'une image est archivée :

- L'image est déplacée vers la classe de stockage d'archives.
- Les images archivées ne peuvent pas être extraites. Les demandes d'extraction de l'image archivée échoueront avec une erreur 404.
- Bien que l'image ne puisse pas être extraite, elle peut toujours être décrite à l'aide de la `describe-images` commande ou répertoriée à l'aide de la `list-images` commande. L'état de l'image sera affiché sous la forme `ARCHIVED`.
- Les images archivées ont une durée de stockage minimale de 90 jours. Vous ne pouvez pas configurer de politiques de cycle de vie qui suppriment les images archivées depuis moins de 90 jours. Si vous devez supprimer des images archivées depuis moins de 90 jours, vous devez utiliser l'`batch-delete-image` API, mais la durée de stockage minimale de 90 jours vous sera facturée.
- L'image apparaît dans un onglet Images archivées dans la vue du référentiel (cet onglet n'apparaît que si au moins une image est archivée dans le référentiel).
- L'image peut être restaurée en tant qu'image active en la sélectionnant manuellement à restaurer ou en repoussant l'image vers le référentiel.
- L'image sera supprimée si le référentiel dispose de politiques de cycle de vie qui suppriment l'image en fonction de critères tels que la durée d'archivage.

### AWS Management Console

#### Pour archiver une image

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/repositories/> l'adresse.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel contenant l'image que vous souhaitez archiver.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Sur la page Référentiels, choisissez le référentiel contenant l'image que vous souhaitez archiver.
5. Sélectionnez l'image que vous souhaitez archiver. Vous verrez les détails de l'image.
6. Pour archiver l'image, cliquez sur le bouton Archiver et sélectionnez Confirmer lorsque vous y êtes invité.

7. S'il s'agit de la première image archivée du référentiel, un nouvel onglet Images archivées apparaît avec la nouvelle image archivée. S'il existe d'autres images archivées, cette image sera ajoutée à cet onglet.

## AWS CLI

Pour archiver une image

- Utilisez la `update-image-storage-class` commande pour archiver une image en mettant à jour sa classe de stockage pour ARCHIVE :

```
aws ecr update-image-storage-class \
 --repository-name my-repository \
 --image-id
 imageDigest=sha256:4f70ef7a4d29e8c0c302b13e25962d8f7a0bd304EXAMPLE \
 --target-storage-class ARCHIVE
```

Pour archiver une image à l'aide de politiques de cycle de vie

- Vous pouvez configurer des règles d'archivage pour vos référentiels à l'aide de politiques de cycle de vie pour archiver automatiquement les images. Les politiques de cycle de vie vous permettent d'archiver automatiquement les images en fonction de critères tels que :
  - Durée écoulée depuis le transfert de l'image
  - Durée écoulée depuis la dernière extraction de l'image
  - Nombre maximum d'images à maintenir actives

Vous pouvez également configurer des politiques de cycle de vie pour supprimer définitivement les images une fois qu'elles ont été archivées pendant une période spécifiée. Pour plus d'informations et des exemples de politiques de cycle de vie comportant des actions d'archivage, consultez [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).

### Note

Les images archivées ont une durée de stockage minimale de 90 jours. Vous ne pouvez pas configurer de politiques de cycle de vie qui suppriment les images archivées depuis

moins de 90 jours. Si vous devez supprimer des images archivées depuis moins de 90 jours, vous devez utiliser l'`batch-delete-image` API, mais la durée de stockage minimale de 90 jours vous sera facturée.

Lorsque vous décrivez des images à l'aide de la `describe-images` commande, les images archivées ont le `image-status` caractère de `ARCHIVED`. Vous pouvez filtrer les images `image-status` pour afficher uniquement les images archivées ou uniquement les images actives.

## Restaurer une image

Lorsque vous restaurez une image archivée, elle est déplacée de la classe de stockage ECR Archive vers la classe de stockage ECR Standard. Les images restaurées sont facturées aux tarifs de stockage standard. Le processus de restauration exécute des actions similaires à celles qui se produisent lors de la création d'une nouvelle image :

- L'image peut être extraite une fois la restauration terminée. La restauration prend généralement jusqu'à 20 minutes, mais elle peut être plus rapide.
- Si le scan en mode push est activé pour le référentiel, l'image restaurée sera numérisée. Notez que les résultats de numérisation antérieurs à l'archivage de l'image ne seront pas disponibles.
- Si la réplication est configurée pour le référentiel, l'image restaurée sera répliquée si la réplication était activée au moment de la restauration.
- L'image restaurée apparaît dans la liste des images actives.

La restauration d'une image prend généralement jusqu'à 20 minutes, mais elle peut être plus rapide. Pendant le processus de restauration, l'image reste dans l'état archivé et ne peut pas être extraite tant que la restauration n'est pas terminée.

### AWS Management Console

Pour restaurer une image archivée

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/repositories/> l'adresse.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel contenant l'image archivée que vous souhaitez restaurer.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Sur la page Référentiels, choisissez le référentiel contenant l'image archivée.

5. Choisissez l'onglet Images archivées.
6. Sélectionnez l'image archivée que vous souhaitez restaurer.
7. Choisissez Restaurer et confirmez l'action de restauration.
8. Attendez que la restauration soit terminée. L'image apparaîtra dans la liste des images actives une fois la restauration terminée.

## AWS CLI

Pour restaurer une image archivée

- Utilisez la `update-image-storage-class` commande pour restaurer une image archivée en mettant à jour sa classe de stockage pour STANDARD :

```
aws ecr update-image-storage-class \
 --repository-name my-repository \
 --image-id
 imageDigest=sha256:4f70ef7a4d29e8c0c302b13e25962d8f7a0bd304EXAMPLE \
 --target-storage-class STANDARD
```

Lorsque vous décrivez des images à l'aide de la `describe-images` commande, les images en cours de restauration ont le `image-status` caractère `deACTIVATING`. Vous pouvez filtrer les images `image-status` en fonction de la valeur `ACTIVATING` pour afficher les images en cours de restauration.

Une autre méthode pour restaurer une image archivée consiste à retransférer l'image vers le référentiel. Lorsque vous publiez une image actuellement archivée, cette image est immédiatement restaurée et supprimée de l'archive.

## Modifier le balisage d'une image dans Amazon ECR

Avec les images Docker Image Manifest V2 Schéma 2, vous pouvez utiliser l'option `--image-tag` de la commande `put-image` pour réétiqueter une image existante. Vous pouvez réétiqueter une image sans la transmettre ni l'extraire avec Docker. Pour les images plus grandes, ce processus permet d'économiser une grande quantité de bande passante réseau et de temps nécessaires au réétiquetage d'une image.

## Réétiqueter une image (AWS CLI)

Pour réétiqueter une image à l'aide du AWS CLI

1. Utilisez la commande `batch-get-image` pour obtenir le manifeste d'image pour l'image à réétiqueter et l'écrire dans un fichier. Dans cet exemple, le manifeste d'une image avec la balise `latest`, dans le référentiel `amazonlinux`, est écrit dans une variable d'environnement nommée `MANIFEST`.

```
MANIFEST=$(aws ecr batch-get-image --repository-name amazonlinux --image-ids
imageTag=latest --output text --query 'images[].imageManifest')
```

2. Utilisez l'option `--image-tag` de la commande `put-image` afin de placer le manifeste de l'image dans Amazon ECR avec une nouvelle étiquette. Dans cet exemple, l'image est étiquetée comme `2017.03`.

### Note

Si l'option `--image-tag` n'est pas disponible dans votre version du AWS CLI, passez à la dernière version. Pour en savoir plus, consultez [Installer la AWS Command Line Interface](#) dans le guide de l'utilisateur AWS Command Line Interface .

```
aws ecr put-image --repository-name amazonlinux --image-tag 2017.03 --image-
manifest "$MANIFEST"
```

3. Vérifiez que la nouvelle étiquette de l'image est attachée à l'image. Dans la sortie ci-dessous, l'image porte les étiquettes `latest` et `2017.03`.

```
aws ecr describe-images --repository-name amazonlinux
```

La sortie est la suivante :

```
{
 "imageDetails": [
 {
 "imageSizeInBytes": 98755613,
 "imageDigest":
"sha256:8d00af8f076eb15a33019c2a3e7f1f655375681c4e5be157a26EXAMPLE",
```

```
 "imageTags": [
 "latest",
 "2017.03"
],
 "registryId": "aws_account_id",
 "repositoryName": "amazonlinux",
 "imagePushedAt": 1499287667.0
 }
]
}
```

## Réétiqueter une image (AWS Tools for Windows PowerShell)

Pour réétiqueter une image à l'aide du AWS Tools for Windows PowerShell

1. Utilisez le `Get-ECRImageBatch` cmdlet pour obtenir la description de l'image à rebaliser et l'écrire dans une variable d'environnement. Dans cet exemple, une image avec la balise `latest`, dans le référentiel `amazonlinux`, est écrite dans la variable d'environnement, `$Image`.

### Note

Si ce n'est pas le cas `Get-ECRImageBatch` cmdlet sur votre système, reportez-vous à la section [Configuration du AWS Tools for Windows PowerShell dans le](#) guide de Outils AWS pour PowerShell l'utilisateur.

```
$Image = Get-ECRImageBatch -ImageId @{ imageTag="latest" } -
RepositoryName amazonlinux
```

2. Écrivez le manifeste de l'image dans la variable d'`$Manifest` environnement.

```
$Manifest = $Image.Images[0].ImageManifest
```

3. Utilisez l'`-ImageTagoption Write-ECRImage` cmdlet pour placer le manifeste de l'image sur Amazon ECR avec une nouvelle balise. Dans cet exemple, l'image est étiquetée comme `2017.09`.

```
Write-ECRImage -RepositoryName amazonlinux -ImageManifest $Manifest -
ImageTag 2017.09
```

4. Vérifiez que la nouvelle étiquette de l'image est attachée à l'image. Dans la sortie ci-dessous, l'image porte les étiquettes `latest` et `2017.09`.

```
Get-ECRImage -RepositoryName amazonlinux
```

La sortie est la suivante :

| ImageDigest                                                             | ImageTag |
|-------------------------------------------------------------------------|----------|
| -----                                                                   | -----    |
| sha256:359b948ea8866817e94765822787cd482279eed0c17bc674a7707f4256d5d497 | latest   |
| sha256:359b948ea8866817e94765822787cd482279eed0c17bc674a7707f4256d5d497 | 2017.09  |

## Empêcher le remplacement des balises d'image dans Amazon ECR

Vous pouvez empêcher le remplacement des balises d'image en activant l'immutabilité des balises dans un référentiel. Une fois l'immutabilité des balises activée, l'`ImageTagAlreadyExistsException` est renvoyée si vous envoyez une image avec une balise déjà présente dans le référentiel. L'immutabilité des balises affecte toutes les balises. Vous ne pouvez pas rendre certaines balises immuables alors que d'autres ne le sont pas.

Vous pouvez utiliser les AWS CLI outils AWS Management Console et pour définir la mutabilité des balises d'image pour un nouveau référentiel ou pour un référentiel existant. Pour créer un référentiel à l'aide des étapes de la console, voir [Création d'un référentiel privé Amazon ECR pour stocker des images](#).

### Configuration de la mutabilité des balises d'image (AWS Management Console)

Pour définir la mutabilité des balises d'image

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/repositories/> l'adresse.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel à modifier.
3. Dans le volet de navigation, choisissez Repositories sous Registre privé.

Si vous ne voyez pas Repositories, choisissez Private registry pour développer le menu, puis choisissez Repositories.

4. Sur la page Dépôts privés, cliquez sur le bouton radio situé devant le nom du référentiel pour lequel vous souhaitez définir les paramètres de mutabilité des balises d'image.
5. Choisissez Actions, puis sélectionnez Référentiel sous Modifier.
6. Pour l'immuabilité des balises Image, choisissez l'un des paramètres de mutabilité des balises suivants pour le référentiel.
  - **Mutable** : choisissez cette option si vous souhaitez que les balises d'image soient remplacées. Recommandé pour les référentiels utilisant des actions de mise en cache par extraction afin de garantir qu'Amazon ECR peut mettre à jour les images mises en cache. En outre, pour désactiver les mises à jour de balises pour quelques balises modifiables, entrez le nom des balises ou utilisez des caractères génériques (\*) pour faire correspondre plusieurs balises similaires dans la zone de texte d'exclusion des balises modifiables.
  - **Immuable** : choisissez cette option si vous souhaitez empêcher le remplacement des balises d'image. Elle s'applique à toutes les balises et exclusions du référentiel lors du transfert d'une image avec une balise existante. Amazon ECR renvoie une exception `ImageTagAlreadyExistsException` si vous tentez de publier une image avec une balise existante. En outre, pour activer les mises à jour des balises pour quelques balises immuables, entrez les noms des balises ou utilisez des caractères génériques (\*) pour faire correspondre plusieurs balises similaires dans la zone de texte Exclusion de balises immuable.
7. Pour Image scan settings (Paramètres d'analyse de l'image), bien que vous puissiez spécifier les paramètres d'analyse au niveau du référentiel pour l'analyse de base, il est recommandé de spécifier la configuration de l'analyse au niveau du registre privé. Spécifiez les paramètres d'analyse dans le registre privé qui vous permettent d'activer l'analyse améliorée ou l'analyse de base, ainsi que de définir des filtres pour spécifier quels référentiels seront analysés. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR](#).
8. Pour Encryption settings (Paramètres de chiffrement), il s'agit d'un champ de vue uniquement car les paramètres de chiffrement d'un référentiel ne peuvent pas être modifiés une fois le référentiel créé.
9. Choisissez Enregistrer pour mettre à jour les paramètres du référentiel.

## Configuration de la mutabilité des balises d'image (AWS CLI)

Créer un référentiel avec des étiquettes immuables configurées

Utilisez l'une des commandes suivantes pour créer un référentiel d'images avec des étiquettes immuables configurées.

- [create-repository](#) (AWS CLI) avec mutabilité des balises d'image

```
aws ecr create-repository --repository-name name --image-tag-mutability IMMUTABLE --region us-east-2
```

- [create-repository](#) (AWS CLI) avec filtres d'exclusion de mutabilité des balises d'image

```
aws ecr create-repository --repository-name name --image-tag-mutability IMMUTABLE_WITH_EXCLUSION --image-tag-mutability-exclusion-filters filterType=WILDCARD,filter=filter-text --region us-east-2
```

- [New-ECRRepository](#) (AWS Tools for Windows PowerShell) avec mutabilité des balises d'image

```
New-ECRRepository -RepositoryName name -ImageTagMutability IMMUTABLE -Region us-east-2 -Force
```

- [New-ECRRepository](#) (AWS Tools for Windows PowerShell) avec filtres d'exclusion de mutabilité des balises d'image

```
New-ECRRepository -RepositoryName name -ImageTagMutability IMMUTABLE_WITH_EXCLUSION -ImageTagMutabilityExclusionFilter @{FilterType=WILDCARD Filter=filter-text} -Region us-east-2 -Force
```

Pour mettre à jour les paramètres de mutabilité des balises d'image pour un référentiel

Utilisez l'une des commandes suivantes pour mettre à jour les paramètres d'immuabilité des étiquettes d'image pour un référentiel existant.

- [put-image-tag-mutability](#) () avec mutabilité des balises d'image AWS CLI

```
aws ecr put-image-tag-mutability --repository-name name --image-tag-mutability IMMUTABLE --region us-east-2
```

- [put-image-tag-mutability](#) (AWS CLI avec filtres d'exclusion de mutabilité des balises d'image)

```
aws ecr put-image-tag-mutability --repository-name name --image-tag-mutability IMMUTABLE_WITH_EXCLUSION --image-tag-mutability-exclusion-filters filterType=WILDCARD,filter=latest --region us-east-2
```

- [Write-ECRImageTagMutability](#) (AWS Tools for Windows PowerShell) avec mutabilité des balises d'image

```
Write-ECRImageTagMutability -RepositoryName name -ImageTagMutability IMMUTABLE -Region us-east-2 -Force
```

- [Write-ECRImageTagMutability](#) (AWS Tools for Windows PowerShell) avec filtres d'exclusion de mutabilité des balises d'image

```
Write-ECRImageTagMutability -RepositoryName name -ImageTagMutability IMMUTABLE_WITH_EXCLUSION -ImageTagMutabilityExclusionFilter @{FilterType=WILDCARD Filter=latest}
```

## Prise en charge du format de manifeste d'image de conteneur dans Amazon ECR

Amazon ECR prend en charge les formats suivants pour manifestes d'images de conteneur :

- Docker Image Manifest V2, Schéma 1 (utilisé avec la version 1.9 de Docker et les versions antérieures)
- Docker Image Manifest V2, Schéma 2 (utilisé avec la version 1.10 de Docker et les versions les plus récentes)
- Spécifications de l'Open Container Initiative (OCI) (v1.0 et v1.1)

La prise en charge de Docker Image Manifest V2, Schéma 2 offre la fonctionnalité suivante :

- Capacité d'utiliser plusieurs étiquettes par image.
- Prise en charge du stockage des images de conteneur Windows.

## Conversion du manifeste d'image Amazon ECR

Lorsque vous procédez à la transmission ou à l'extraction des images vers et depuis Amazon ECR, le client moteur du conteneur (par exemple, Docker) communique avec le registre pour convenir du format de manifeste compris par le client et le registre à utiliser pour l'image.

Lorsque vous transmettez une image à Amazon ECR avec la version de Docker 1.9 ou version antérieure, le format du manifeste d'image est stocké en tant que Docker Image Manifest V2, Schéma 1. Lorsque vous transmettez une image à Amazon ECR avec la version de Docker 1.10 ou version plus récente, le format du manifeste d'image est stocké en tant que Docker Image Manifest V2, Schéma 2.

Lorsque vous procédez à l'extraction d'une image d'Amazon ECR par étiquette, renvoie le format du manifeste d'image stocké dans le référentiel. Le format est renvoyé uniquement si ce format est compris par le client. Si le format du manifeste d'image stocké n'est pas compris par le client, Amazon ECR le convertit dans un format qui est compris par le client. Par exemple, si un client Docker 1.9 demande un manifeste d'image stocké en tant que Docker Image Manifest V2 Schéma 2, Amazon ECR renvoie le manifeste au format Docker Image Manifest V2 Schéma 1. Le tableau ci-dessous décrit les conversions disponibles prises en charge par Amazon ECR lorsqu'une image est extraite par étiquette :

| Schéma demandé par le client | Transmis à ECR en tant que V2, schéma 1                        | Transmis à ECR en tant que V2, schéma 2 | Transmis à ECR en tant qu'OCI |
|------------------------------|----------------------------------------------------------------|-----------------------------------------|-------------------------------|
| V2, schéma 1                 | Aucune conversion requise                                      | Conversion à V2, schéma 1               | Aucune conversion disponible  |
| V2, schéma 2                 | Aucune conversion disponible, le client revient à V2, Schéma 1 | Aucune conversion requise               | Conversion à V2, schéma 2     |
| OCI                          | Aucune conversion disponible                                   | Conversion à OCI                        | Aucune conversion requise     |

**⚠ Important**

Si vous tirez une image par résumé, il n'y aura pas de conversion disponible. Votre client devra comprendre le format du manifeste d'image stocké dans Amazon ECR. Si vous demandez une image Docker Image Manifest V2, Schéma 2 par hachage sur un client Docker 1.9 ou antérieur, l'extraction de l'image échouera. Pour en savoir plus, consultez [Compatibilité de registre](#) dans la documentation Docker.

Dans cet exemple, si vous demandez la même image par étiquette, Amazon ECR convertira le manifeste d'image dans un format que le client pourra comprendre. L'extraction d'image a réussi.

## Utiliser des images Amazon ECR avec Amazon ECS

Vous pouvez utiliser vos référentiels Amazon ECR privés pour héberger des images de conteneurs et des artefacts que vos tâches Amazon ECS peuvent extraire. Pour que cela fonctionne, l'agent de conteneur Amazon ECS, ou Fargate, doit être autorisé à créer `ecr:BatchGetImage` `lecr:GetDownloadUrlForLayer`, et `ecr:GetAuthorizationToken` APIs

### Autorisations IAM requises

Le tableau suivant indique le rôle IAM à utiliser, pour chaque type de lancement, qui fournit les autorisations requises pour que vos tâches puissent être extraites d'un référentiel Amazon ECR privé. Amazon ECS fournit des politiques IAM gérées qui incluent les autorisations requises.

| Type de lancement                       | Rôle IAM                                                                                                                                                                                                                                                 | AWS politique IAM gérée                                                                                                                                                                          |
|-----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Amazon ECS sur des instances Amazon EC2 | Utilisez le rôle IAM de l'instance de conteneur, qui est associé à l'instance Amazon EC2 enregistrée dans votre cluster Amazon ECS. Pour plus d'informations, consultez la section consacrée au <a href="#">Rôle IAM d'instance de conteneur</a> dans le | AmazonEC2ContainerServiceforEC2Role<br><br>Pour plus d'informations, consultez <a href="#">AmazonEC2ContainerServiceforEC2Role</a> dans le Guide du développeur Amazon Elastic Container Service |

| Type de lancement                     | Rôle IAM                                                                                                                                                                                                                                                                                                                                               | AWS politique IAM gérée                                                                                                                                                                           |
|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                       | Guide du développeur<br>Amazon Elastic Container Service                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                   |
| Amazon ECS sur Fargate                | Utilisez le rôle IAM d'exécution des tâches auquel vous faites référence dans votre définition de tâche Amazon ECS. Pour de plus amples informations, veuillez consulter <a href="#">Rôle IAM d'exécution de tâche</a> dans le Manuel du développeur Amazon Elastic Container Service                                                                  | AmazonECSTaskExecutionRolePolicy<br><br>Pour plus d'informations, consultez <a href="#">AmazonECS TaskExecutionRolePolicy</a> dans le Guide du développeur Amazon Elastic Container Service       |
| Amazon ECS sur les instances externes | Utilisez le rôle IAM de l'instance de conteneur, qui est associé au serveur sur site ou à la machine virtuelle (VM) enregistrée vers votre cluster Amazon ECS. Pour plus d'informations, reportez-vous à la section consacrée au <a href="#">Rôle Amazon ECS d'instance de conteneur</a> dans le Guide du développeur Amazon Elastic Container Service | AmazonEC2ContainerServiceforEC2Role<br><br>Pour plus d'informations, consultez <a href="#">AmazonEC2 ContainerServiceforEC2Role</a> dans le Guide du développeur Amazon Elastic Container Service |

### Important

Les politiques IAM AWS gérées contiennent des autorisations supplémentaires dont vous n'aurez peut-être pas besoin pour votre utilisation. Dans ce cas, il s'agit des autorisations minimales requises pour extraire des données d'un référentiel Amazon ECR privé.

JSON

```
{
```

```
"Version": "2012-10-17",
"Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer",
 "ecr:GetAuthorizationToken"
],
 "Resource": "*"
 }
]
```

## Spécifier une image Amazon ECR dans une définition de tâche Amazon ECS

Lorsque vous créez une définition de tâche Amazon ECS, vous pouvez spécifier une image de conteneur hébergée dans un référentiel Amazon ECR privé. Dans la définition de tâches, assurez-vous d'utiliser la dénomination `registry/repository:tag` complète pour vos images Amazon ECR. Par exemple, `aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:latest`.

L'extrait de définition de tâche suivant montre la syntaxe que vous utiliseriez pour spécifier une image de conteneur hébergée dans Amazon ECR pour votre définition de tâche Amazon ECS.

```
{
 "family": "task-definition-name",
 ...
 "containerDefinitions": [
 {
 "name": "container-name",
 "image": "aws_account_id.dkr.ecr.region.amazonaws.com/my-
repository:latest",
 ...
 }
],
 ...
}
```

# Utiliser des images Amazon ECR avec Amazon EKS

Vous pouvez utiliser vos images Amazon ECR avec Amazon EKS.

Lorsque vous référencez une image à partir d'Amazon ECR, vous devez utiliser le nom complet `registry/repository:tag` de l'image. Par exemple, `aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:latest`.

## Autorisations IAM requises

Si vous avez des charges de travail Amazon EKS hébergées sur des nœuds gérés, des nœuds autogérés AWS Fargate, ou consultez les points suivants :

- Charges de travail Amazon EKS hébergées sur des nœuds gérés ou autogérés : le rôle IAM du nœud de travail Amazon EKS (NodeInstanceRole) est requis. Le rôle IAM de composant master Amazon EKS doit contenir les autorisations de politique IAM suivantes pour Amazon ECR.

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "ecr:BatchCheckLayerAvailability",
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer",
 "ecr:GetAuthorizationToken"
],
 "Resource": "*"
 }
]
}
```

### Note

Si vous avez utilisé `eksctl` les CloudFormation modèles de [Getting Started with Amazon EKS](#) pour créer votre cluster et vos groupes de nœuds de travail, ces autorisations IAM sont appliquées par défaut au rôle IAM de votre nœud de travail.

- Charges de travail Amazon EKS hébergées sur AWS Fargate : utilisez le rôle d'exécution du module Fargate, qui autorise vos pods à extraire des images depuis des référentiels Amazon ECR privés. Pour plus d'informations, consultez [Création d'un rôle d'exécution de pod Fargate](#).

## Installation d'un graphique Helm sur un cluster Amazon EKS

Les cartes Helm hébergées dans Amazon ECR peuvent être installées sur vos clusters Amazon EKS.

### Conditions préalables

- Installez la dernière version du Helm client. Ces étapes ont été écrites à l'aide de la version Helm 3.9.0. Pour en savoir plus, consultez [Installation Helm](#).
- Vous avez au moins une version 1.23.9 ou 2.6.3 du AWS CLI installé sur votre ordinateur. Pour plus d'informations, consultez [Installation ou mise à jour de la version la plus récente de l' AWS CLI](#).
- Vous avez envoyé les Charts de Helm à votre référentiel Amazon ECR. Pour de plus amples informations, veuillez consulter [Transférer un graphique de Helm vers un référentiel privé Amazon ECR](#).
- Vous avez configuré le `kubectl` afin qu'il fonctionne avec Amazon EKS. Pour en savoir plus, consultez [Créer un kubeconfig pour Amazon EKS](#) dans le guide de l'utilisateur Amazon EKS. Si les commandes suivantes aboutissent pour votre cluster, votre configuration est correcte.

```
kubectl get svc
```

Pour installer un graphique Helm sur un cluster Amazon EKS

1. Authentifiez votre client Helm dans le registre Amazon ECR où vos Charts de Helm sont hébergés. Vous devez obtenir des jetons d'authentification pour chaque registre utilisé. Les jetons sont valides pendant 12 heures. Pour de plus amples informations, veuillez consulter [Authentification du registre privé dans Amazon ECR](#).

```
aws ecr get-login-password \
 --region us-west-2 | helm registry login \
 --username AWS \
 --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

2. Installez le chart. `helm-test-chart` Remplacez-le par votre référentiel et `0.1.0` par le tag de votre graphique Helm.

```
helm install ecr-chart-demo oci://aws_account_id.dkr.ecr.region.amazonaws.com/helm-test-chart --version 0.1.0
```

La sortie doit être similaire à ceci :

```
NAME: ecr-chart-demo
LAST DEPLOYED: Tue May 31 17:38:56 2022
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
```

3. Vérifiez l'installation des Charts de Helm.

```
helm list -n default
```

Exemple de sortie :

| NAME           | NAMESPACE             | REVISION    | UPDATED                             |
|----------------|-----------------------|-------------|-------------------------------------|
| STATUS         | CHART                 | APP VERSION |                                     |
| ecr-chart-demo | default               | 1           | 2022-06-01 15:56:40.128669157 +0000 |
| UTC deployed   | helm-test-chart-0.1.0 | 1.16.0      |                                     |

4. (Facultatif) Consultez les Charts de Helm installés ConfigMap.

```
kubectl describe configmap helm-test-chart-configmap
```

5. Lorsque vous aurez terminé, vous pourrez supprimer la version des Charts de Helm de votre cluster.

```
helm uninstall ecr-chart-demo
```

# Signer des images dans Amazon ECR

Amazon ECR s'intègre AWS Signer pour vous permettre de signer les images de vos conteneurs de deux manières : signature gérée (automatique, recommandée) et signature manuelle (côté client). Vous pouvez stocker à la fois les images de vos conteneurs et les signatures dans vos référentiels privés.

## Choisissez une méthode de signature

Amazon ECR prend en charge deux méthodes pour signer les images de conteneurs :

### Signature gérée (recommandée)

La signature gérée génère automatiquement des signatures cryptographiques lorsque les images sont transmises à Amazon ECR. Cette méthode simplifie la configuration. La signature gérée est l'approche recommandée pour la plupart des utilisateurs. Pour de plus amples informations, veuillez consulter [Signature gérée](#).

### Signature manuelle

La signature manuelle utilise la CLI Notation et le AWS Signer plug-in pour signer les images avant de les transférer vers Amazon ECR. Cette méthode permet de mieux contrôler le processus de signature et est utile lorsque vous devez signer des images en dehors du flux de travail push ou que vous avez besoin d'un contrôle précis des opérations de signature. Pour de plus amples informations, veuillez consulter [Signature manuelle](#).

## Considérations

Les points suivants doivent être pris en compte lors de l'utilisation de la signature d'image Amazon ECR :

- Les signatures stockées dans votre référentiel sont prises en compte dans les quotas de service correspondant au nombre maximum d'images par référentiel. Chaque signature compte pour 1 artefact par rapport au quota d'images par dépôt. Pour de plus amples informations, veuillez consulter [Service Quotas Amazon ECR](#).
- Lorsque des artefacts de référence sont présents dans un référentiel, les politiques de cycle de vie d'Amazon ECR nettoient automatiquement ces artefacts dans les 24 heures suivant la suppression de l'image d'objet.

# Signature gérée

La signature gérée par Amazon ECR signe automatiquement les images de vos conteneurs en générant des signatures cryptographiques à l'aide de [AWS Signer](#) lorsque les images sont transmises à Amazon ECR. Cela élimine le besoin d'installer et de configurer des outils côté client et vous permet de gérer de manière centralisée la signature en tant que configuration de registre.

## Conditions préalables

Pour configurer la signature gérée, vous créez une configuration de signature avec Amazon ECR qui fait référence à un ou plusieurs profils de signature de signataires et, éventuellement, à des filtres de référentiel qui limitent les référentiels dont les images doivent être signées. Une fois configurée, Amazon ECR Managed Signing signe automatiquement les images lorsqu'elles sont transmises en utilisant l'identité de l'entité qui les diffuse.

Avant de pouvoir configurer la signature gérée, vous devez disposer des éléments suivants :

- Un profil de signature de signataire : créez au moins un [profil de signature de](#) signataire. Un profil de signature est une ressource de AWS signature unique que vous pouvez utiliser pour effectuer des opérations de signature dans Amazon ECR. Les profils de signature vous permettent de signer et de vérifier des artefacts de code, tels que des images de conteneur et des ensembles de AWS Lambda déploiement. Chaque profil de signature désigne la plate-forme de signature à laquelle signer, un identifiant de plate-forme et d'autres informations spécifiques à la plate-forme. Par exemple, l'ARN d'un profil de signature ressemble à ceci : `arn:partition:signer:region:account-id:/signing-profiles/profile-name`.
- Autorisations IAM — Le principal IAM qui envoie l'image doit disposer des autorisations IAM nécessaires pour accéder au profil de signature du signataire concerné et au référentiel ECR correspondant. Vous devez modifier la politique basée sur l'identité pour le principal IAM afin d'inclure des autorisations pour les opérations de référentiel ECR et les opérations de signature des signataires. L'exemple de politique suivant montre les autorisations requises :

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "UploadSignaturePermissions",
```

```

 "Effect": "Allow",
 "Action": [
 "ecr:CompleteLayerUpload",
 "ecr:UploadLayerPart",
 "ecr:InitiateLayerUpload",
 "ecr:BatchCheckLayerAvailability",
 "ecr:PutImage"
],
 "Resource": "arn:aws:ecr:region:account-id:repository/repository-name"
},
{
 "Sid": "SignPermissions",
 "Effect": "Allow",
 "Action": [
 "signer:SignPayload"
],
 "Resource": "arn:aws:signer:region:account-id:/signing-profiles/signing-profile-
name"
}
]
}

```

Avec la signature gérée par Amazon ECR, vous pouvez créer plusieurs règles de signature (jusqu'à 10 par registre) afin de renforcer les limites de sécurité. Par exemple, vous pouvez exécuter plusieurs pipelines de génération et souhaitez limiter les référentiels que chaque pipeline peut signer. Dans chaque règle, vous configurez un profil de signature et spécifiez des filtres de nom de référentiel. Lorsqu'une nouvelle image est envoyée, Amazon ECR détermine la règle de signature et le profil de signature autorisés à signer l'image. S'il existe plusieurs correspondances, Amazon ECR génère plusieurs signatures.

#### Note

Si vous vérifiez les signatures manuellement, vous devez tout de même installer la CLI Notation.

#### Note

La signature gérée par Amazon ECR est disponible dans toutes les AWS régions où la signature d'images de conteneurs avec AWS Signer est disponible.

## Prise en main

Procédez comme suit pour configurer la signature gérée. Vous fournissez à Amazon ECR une référence à un profil de signature de signataire et, éventuellement, des filtres qui limitent les référentiels dont les images doivent être signées.

### AWS Management Console

Procédez comme suit pour configurer la signature gérée à l'aide du AWS Management Console.

1. Ouvrez la [console Amazon ECR](#). Dans le volet de navigation de gauche, sélectionnez Registre privé, Fonctionnalités et paramètres, Signature gérée.
2. Sur la page Règles de signature, sélectionnez Créer une règle.
3. Sur la page du profil de signature, sous Sélectionnez un profil de AWS signataire, choisissez Créer un nouveau profil de AWS signataire, entrez un nom de profil et, éventuellement, modifiez la période de validité de la signature. Sélectionnez ensuite Next.
4. Sur la page Filtres, sous Sélectionner les référentiels, entrez un filtre de nom de référentiel. Sélectionnez ensuite Next.
5. Sur la page Réviser et créer, vérifiez le profil du AWS signataire et les filtres de nom du référentiel que vous avez saisis. Si tout semble correct, sélectionnez Enregistrer.

### AWS CLI

Utilisez les AWS CLI commandes suivantes pour configurer la signature gérée.

- Création d'une règle de signature

Créez une configuration de signature avec l'ARN de votre profil de signature. Créez un fichier JSON avec le contenu suivant :

```
{
 "rules": [
 {
 "signingProfileArn": "arn:aws:signer:region:account-id:/signing-
profiles/profile-name",
 "repositoryFilters": [
 {
 "filter": "test*",
 "filterType": "WILDCARD_MATCH"
 }
]
 }
]
}
```

```

 }
]
}

```

Ensuite, exécutez la commande suivante :

```

aws ecr --region region \
 put-signing-configuration \
 --signing-configuration file://signing-config.json

```

Vous devriez voir la réponse de l'API contenant la configuration de signature.

- Afficher votre configuration de signature

Récupérez votre configuration de signature :

```

aws ecr --region region \
 get-signing-configuration

```

Vous devriez voir la réponse de l'API contenant la configuration de signature.

- Vérifier l'état de signature des images

Transférez une image dans votre dépôt. Par exemple :

```

docker pull ubuntu

IMAGE_NAME="account-id.dkr.ecr.region.amazonaws.com/repository-name"
IMAGE_TAG="${IMAGE_NAME}:test-1"

docker tag ubuntu $IMAGE_TAG
docker push $IMAGE_TAG

```

Après avoir appuyé, utilisez votre balise d'image pour vérifier l'état de signature :

```

aws ecr --region region \
 describe-image-signing-status \
 --repository-name repository-name \
 --image-id imageTag=test-1

```

Si le nom du référentiel correspond à votre filtre de référentiel défini dans la configuration de signature, vous devriez voir le statut de signature dans la réponse de l'API. Si le statut est satisfaisant, vous devriez voir une signature envoyée à votre dépôt.

- Supprimer votre configuration de signature

Supprimez votre configuration de signature :

```
aws ecr --region region \
delete-signing-configuration
```

Vous devriez voir la réponse de l'API contenant la configuration de signature supprimée.

## Considérations

Les limites et fonctionnalités suivantes s'appliquent à la signature gérée :

- La signature entre régions n'est pas prise en charge : les profils de signature doivent se trouver dans la même région que votre registre Amazon ECR. Vous ne pouvez pas utiliser le profil de signature d'une région pour signer des images dans un registre situé dans une autre région.
- La signature entre comptes est prise en charge : les profils de signature peuvent se trouver sur des comptes différents de ceux de votre registre Amazon ECR. Cela permet aux entreprises de gérer les profils de signature de manière centralisée tout en permettant aux développeurs d'autres comptes de les utiliser. Pour plus d'informations, consultez la [section Configurer la signature entre comptes pour le signataire](#) dans le guide du AWS Signer développeur.
- Les signatures ne peuvent pas être signées : vous ne pouvez pas signer les signatures elles-mêmes. Seules les images du conteneur peuvent être signées.

## Vérification de signature

Après avoir signé les images de vos conteneurs, vous pouvez vérifier les signatures pour vous assurer que les images n'ont pas été falsifiées et proviennent d'une source fiable. Amazon ECR prend en charge plusieurs méthodes de vérification des signatures :

## Vérification gérée avec Amazon EKS

Amazon EKS fournit une intégration native pour la vérification automatique des signatures. Lorsque vous configurez la vérification des signatures dans vos clusters Amazon EKS, le service vérifie automatiquement les signatures d'image avant d'autoriser l'exécution des conteneurs. Pour plus d'informations sur la configuration de la vérification des signatures, consultez la section [Valider les signatures d'images de conteneurs lors du déploiement](#) dans le guide de l'utilisateur Amazon EKS.

## Contrôleur d'admission Lambda pour Amazon ECS

Amazon ECS fournit des hooks du cycle de vie des services qui vous permettent d'exécuter une logique personnalisée lors des déploiements de services. Ces hooks peuvent déclencher des AWS Lambda fonctions à des moments spécifiques du processus de déploiement, ce qui vous permet de valider les signatures des images des conteneurs avant d'autoriser le démarrage des services. Pour plus d'informations, consultez [Vérifier les signatures d'image de conteneur pour Amazon ECS](#) dans le manuel du AWS Signer développeur.

## Vérification manuelle avec Notation CLI

Vous pouvez vérifier les signatures manuellement à l'aide de la CLI Notation. Cette méthode nécessite l'installation et la configuration de la CLI Notation sur votre machine locale ou dans votre environnement de vérification. Pour obtenir des instructions détaillées sur la vérification d'une image à l'aide de la Notation CLI, voir [Vérifier une image localement après signature](#) dans le manuel du AWS Signer développeur.

## Configuration de l'authentification pour le client Notation

Si vous utilisez la signature manuelle ou si vous vérifiez les signatures manuellement à l'aide de la CLI Notation, vous devez configurer le client Notation afin qu'il puisse s'authentifier auprès d'Amazon ECR. Si Docker est installé sur le même hôte que le client Notation, ce dernier réutilisera la même méthode d'authentification que celle utilisée pour le client Docker. Le Docker login et logout les commandes permettront à la notation sign et aux verify commandes d'utiliser les mêmes informations d'identification, et vous n'aurez pas à authentifier la notation séparément. Pour plus d'informations sur la configuration de votre client Notation pour l'authentification, voir [Authentifier avec des registres conformes à l'OCI-OCI](#) dans la documentation du projet Notary.

Si vous n'utilisez pas Docker ou un autre outil qui utilise des informations d'identification Docker, nous vous recommandons d'utiliser l'assistant des informations d'identification Amazon ECR Docker

comme magasin d'informations d'identification. Pour plus d'informations sur l'installation et la configuration de l'assistant des informations d'identification Amazon ECR Docker, consultez la page [Assistant des informations d'identification Amazon ECR Docker](#) (français non garanti).

## Signature manuelle

La signature manuelle utilise la CLI Notation et le AWS Signer plug-in pour signer les images avant de les transférer vers Amazon ECR. Cette méthode permet de mieux contrôler le processus de signature et est utile lorsque vous devez signer des images en dehors du flux de travail push ou que vous avez besoin d'un contrôle précis des opérations de signature.

Pour obtenir des instructions détaillées sur la signature d'images de conteneurs à l'aide de la CLI Notation AWS Signer, voir [Signer des images de conteneur dans Signer](#) et les rubriques connexes dans le Guide du AWS Signer développeur.

## Conditions préalables

Avant de commencer, les prérequis suivants doivent être respectés.

- Installez et configurez la version la plus récente de l' AWS CLI. Pour plus d'informations, consultez [Installation ou mise à jour de la version la plus récente de l' AWS CLI](#) (langue française non garantie) dans le Guide de l'utilisateur AWS Command Line Interface .
- Installez la CLI Notation et le AWS Signer plugin pour Notation. Pour plus d'informations, consultez [Prérequis pour la signature des images de conteneur](#) (langue française non garantie) dans le Guide du développeur AWS Signer .
- Vous devez avoir une image de conteneur enregistrée dans un référentiel privé Amazon ECR à signer. Pour de plus amples informations, veuillez consulter [Transférer une image vers un référentiel privé Amazon ECR](#).

# Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR

La numérisation d'images Amazon ECR permet d'identifier les vulnérabilités logicielles dans vos images de conteneur. Les types d'analyse suivants sont proposés.

## Important

Si vous passez de la numérisation améliorée à la numérisation de base, les scans précédemment établis ne seront plus disponibles. Vous devrez à nouveau configurer vos scans. Toutefois, si vous revenez à votre type de numérisation précédent, les scans établis seront disponibles.

## Note

Les images archivées ne peuvent pas être numérisées. Les images archivées doivent être restaurées avant de pouvoir être numérisées. Pour plus d'informations sur l'archivage et la restauration d'images, consultez [Archivage d'une image dans Amazon ECR](#).

- Numérisation améliorée : Amazon ECR s'intègre à Amazon Inspector pour fournir une analyse automatique et continue de vos référentiels. Vos images de conteneur sont analysées à la fois pour les vulnérabilités des systèmes d'exploitation et des packages de langage de programmation. Lorsque de nouvelles vulnérabilités apparaissent, les résultats de l'analyse sont mis à jour et Amazon Inspector émet un événement EventBridge pour vous en informer. La numérisation améliorée fournit les avantages suivants :
  - Vulnérabilités des packages liés aux systèmes d'exploitation et aux
  - Deux fréquences de numérisation : scan en mode push et scan continu
- Analyse de base — Amazon ECR utilise une technologie AWS native associée à la base de données Common Vulnerabilities and Exposures (CVEs) pour détecter les vulnérabilités du système d'exploitation.

Avec l'analyse de base, vous configurez vos référentiels pour qu'ils soient analysés au moment de l'envoi (push) ou vous pouvez effectuer des analyses manuelles et Amazon ECR fournit une liste des résultats de l'analyse. La numérisation de base fournit les éléments suivants :

- Analyses du système d'
- Deux fréquences de numérisation : manuelle et numérisation en mode push

#### Important

La nouvelle version d'Amazon ECR Basic Scanning n'utilise pas les `imageScanStatus` attributs `imageScanFindingsSummary` et de la réponse de `DescribeImagesAPI` pour renvoyer les résultats du scan. Utilisez plutôt `DescribeImageScanFindingsAPI`. Pour de plus amples informations, veuillez consulter [DescribeImageScanFindings](#).

## Filtres permettant de choisir les référentiels à analyser dans Amazon ECR

Lorsque vous configurez la numérisation d'images pour votre registre privé, vous pouvez utiliser des filtres pour choisir les référentiels à analyser.

Lorsque l'analyse de base est utilisée, vous pouvez spécifier des filtres d'analyse lors du transfert par push pour indiquer quels référentiels sont configurés pour effectuer une analyse d'image lorsque de nouvelles images sont transférées par push. Tous les référentiels ne correspondant pas à une analyse de base sur filtre push seront réglés sur la fréquence de numérisation manuelle, ce qui signifie que pour effectuer une analyse, vous devez déclencher l'analyse manuellement.

Lorsque l'analyse améliorée est utilisée, vous pouvez spécifier des filtres distincts pour l'analyse lors du transfert par push et l'analyse continue. L'analyse sera désactivée pour tous les référentiels ne correspondant pas à un filtre d'analyse améliorée. Si vous utilisez l'analyse améliorée et spécifiez des filtres distincts pour l'analyse lors du transfert par push et l'analyse continue et qu'un même référentiel correspond aux critères des deux filtres, Amazon ECR applique le filtre d'analyse continue plutôt que le filtre d'analyse lors du transfert par push pour ce référentiel.

## Filtrer les caractères génériques

Lorsqu'un filtre est spécifié, un filtre sans caractère générique correspondra à tous les noms de référentiel qui contiennent le filtre. Un filtre avec un caractère générique (\*) correspond à tout nom de référentiel où le caractère générique remplace zéro ou plusieurs caractères dans le nom du référentiel.

Le tableau suivant fournit des exemples où les noms de référentiels sont exprimés sur l'axe horizontal et les exemples de filtres sont spécifiés sur l'axe vertical.

|           | prod | repo-prod | prod-repo | repo-prod-repo | prodrepo |
|-----------|------|-----------|-----------|----------------|----------|
| prod      | Oui  | Oui       | Oui       | Oui            | Oui      |
| *prod     | Oui  | Oui       | Non       | Non            | Non      |
| prod*     | Oui  | Non       | Oui       | Non            | Oui      |
| *prod*    | Oui  | Oui       | Oui       | Oui            | Oui      |
| prod*repo | Non  | Non       | Oui       | Non            | Oui      |

## Scannez les images pour détecter les vulnérabilités du système d'exploitation et des packages de langage de programmation dans Amazon ECR

L'analyse améliorée d'Amazon ECR est une intégration avec Amazon Inspector qui fournit une analyse de vulnérabilité pour vos images de conteneur. Vos images de conteneur sont analysées à la fois pour les vulnérabilités des systèmes d'exploitation et des packages de langage de programmation. Vous pouvez afficher les résultats de l'analyse directement avec Amazon ECR et avec Amazon Inspector. Pour plus d'informations sur Amazon Inspector, consultez [Analyse des images de conteneur avec Amazon Inspector](#) dans le Guide de l'utilisateur Amazon Inspector.

Avec l'analyse améliorée, vous pouvez choisir les référentiels qui sont configurés pour une analyse automatique et continue et ceux qui sont configurés pour une analyse sur demande. Cela se fait en définissant des filtres d'analyse.

### Considérations relatives à l'analyse améliorée

Tenez compte des points suivants avant d'activer le scan amélioré Amazon ECR.

- L'utilisation de cette fonctionnalité n'entraîne aucun coût supplémentaire pour Amazon ECR, mais Amazon Inspector facture la numérisation de vos images. Cette fonctionnalité est disponible dans les régions où Amazon Inspector est pris en charge. Pour en savoir plus, consultez :

- Tarification Amazon Inspector — [Tarification Amazon Inspector](#).
- Régions prises en charge par Amazon Inspector : [régions et points de terminaison](#).
- La numérisation améliorée Amazon ECR montre comment les images sont utilisées sur Amazon EKS et Amazon ECS. Vous pouvez voir quand les images ont été utilisées pour la dernière fois et identifier le nombre de clusters utilisant chaque image. Ces informations vous aident à prioriser la correction des vulnérabilités pour les images activement utilisées. Vous pouvez rapidement déterminer quels clusters sont susceptibles d'être affectés par des vulnérabilités récemment découvertes. Pour plus d'informations sur la manière de demander ces informations et d'afficher la réponse, consultez [DescribeImageScanFindings](#).
- Amazon Inspector prend en charge l'analyse pour des systèmes d'exploitation spécifiques. Pour obtenir la liste complète, consultez [Systèmes d'exploitation pris en charge – Analyse Amazon ECR](#) dans le Guide de l'utilisateur Amazon Inspector.
- Amazon Inspector utilise un rôle IAM lié à un service, qui fournit les autorisations nécessaires pour fournir une analyse améliorée pour vos référentiels. Le rôle IAM lié à un service est créé automatiquement par Amazon Inspector lorsque l'analyse améliorée est activée pour votre registre privé. Pour plus d'informations, consultez [Utilisation des rôles liés à un service pour Amazon Inspector](#) dans le Guide de l'utilisateur d'Amazon Inspector.
- Lorsque vous activez initialement la numérisation améliorée pour votre registre privé, Amazon Inspector reconnaît uniquement les images envoyées à Amazon ECR au cours des 14 derniers jours, en fonction de l'horodatage des images transmises. Les images plus anciennes auront le statut d'analyse `SCAN_ELIGIBILITY_EXPIRED`. Si vous souhaitez que ces images soient analysées par Amazon Inspector, vous devez les envoyer à nouveau dans votre référentiel.
- Lorsque l'analyse améliorée est activée pour votre registre privé Amazon ECR, les référentiels correspondant aux filtres d'analyse sont analysés en utilisant uniquement l'analyse améliorée. Tous les référentiels qui ne correspondent pas à un filtre auront une fréquence d'analyse `Off` et ne seront pas analysés. Les analyses manuelles qui utilisent l'analyse améliorée ne sont pas prises en charge. Pour de plus amples informations, veuillez consulter [Filtres permettant de choisir les référentiels à analyser dans Amazon ECR](#).
- Si vous spécifiez des filtres distincts pour l'analyse lors du transfert par push et l'analyse continue et qu'un même référentiel correspond aux critères des deux filtres, Amazon ECR applique le filtre d'analyse continue plutôt que le filtre d'analyse lors du transfert par push pour ce référentiel.
- Lorsque le scan amélioré est activé, Amazon ECR envoie un événement EventBridge lorsque la fréquence d'analyse d'un référentiel est modifiée. Amazon Inspector émet des événements EventBridge lorsqu'une numérisation initiale est terminée et lorsqu'un résultat de numérisation d'image est créé, mis à jour ou fermé.

## Modification de la durée de numérisation améliorée pour les images dans Amazon Inspector

Après avoir activé la numérisation améliorée, Amazon ECR analyse en permanence les images récemment envoyées pendant la durée configurée. Par défaut, Amazon Inspector surveille vos référentiels jusqu'à ce que les images soient supprimées ou que la numérisation améliorée soit désactivée. Vous pouvez configurer à la fois la durée de la date de diffusion (jusqu'à Lifetime) et la durée de la nouvelle analyse dans la console Amazon Inspector en fonction des besoins de votre environnement. Lorsque la durée d'analyse d'un référentiel est écoulée, l'état de l'analyse s'affiche sous la forme. `SCAN_ELIGIBILITY_EXPIRED` Pour plus d'informations sur la configuration des paramètres de durée de nouvelle analyse pour Amazon ECR dans Amazon Inspector, consultez [Configuration de la durée de nouvelle analyse Amazon ECR](#) dans le guide de l'utilisateur d'Amazon Inspector.

## Autorisations IAM requises pour une numérisation améliorée dans Amazon ECR

Le scan amélioré Amazon ECR nécessite un rôle IAM lié au service Amazon Inspector et le principal IAM qui active et utilise le scan amélioré doit être autorisé à appeler l'Amazon Inspector nécessaire à la numérisation. APIs Le rôle IAM lié à un service Amazon Inspector est créé automatiquement par Amazon Inspector lorsque l'analyse améliorée est activée pour votre registre privé. Pour plus d'informations, consultez [Utilisation des rôles liés à un service pour Amazon Inspector](#) dans le Guide de l'utilisateur d'Amazon Inspector.

La politique IAM suivante accorde les autorisations requises pour activer et utiliser l'analyse améliorée. Elle comprend l'autorisation nécessaire à Amazon Inspector pour créer le rôle IAM lié à un service ainsi que les autorisations API d'Amazon Inspector nécessaires pour activer et désactiver l'analyse améliorée et récupérer les résultats de l'analyse.

### JSON

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

```

 "inspector2:Enable",
 "inspector2:Disable",
 "inspector2:ListFindings",
 "inspector2:ListAccountPermissions",
 "inspector2:ListCoverage"
],
 "Resource": "*"
},
{
 "Effect": "Allow",
 "Action": "iam:CreateServiceLinkedRole",
 "Resource": "*",
 "Condition": {
 "StringEquals": {
 "iam:AWSServiceName": [
 "inspector2.amazonaws.com"
]
 }
 }
}
]
}

```

## Configuration de la numérisation améliorée pour les images dans Amazon ECR

Configurez l'analyse améliorée par région pour votre registre privé.

Vérifiez que vous disposez des autorisations IAM appropriées pour configurer le scan amélioré. Pour plus d'informations, consultez [Autorisations IAM requises pour une numérisation améliorée dans Amazon ECR](#).

### AWS Management Console

Pour activer l'analyse améliorée de votre registre privé

1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez définir la configuration d'analyse.
3. Dans le volet de navigation, choisissez Registre privé, puis Paramètres.

4. Sur la page Scanning configuration (Configuration de l'analyse), pour Scan type (Type d'analyse), choisissez Enhanced scanning (Analyse améliorée).

Par défaut, lorsque l'option Analyse améliorée est sélectionnée, tous vos référentiels sont analysés en continu.

5. Pour choisir des référentiels spécifiques à analyser en continu, décochez la case Analyser en continu tous les référentiels, puis définissez vos filtres :

 Important

Les filtres sans caractère générique correspondent à tous les noms de référentiel qui contiennent le filtre. Les filtres avec des caractères génériques (\*) correspondent à un nom de référentiel où le caractère générique remplace zéro ou plusieurs caractères dans le nom du référentiel. Pour voir des exemples du comportement des filtres, reportez-vous à [the section called "Filtrer les caractères génériques"](#).

- a. Entrez un filtre basé sur les noms des référentiels, puis choisissez Ajouter un filtre.
  - b. Décidez quels référentiels numériser lorsqu'une image est envoyée :
    - Pour analyser tous les référentiels en mode push, sélectionnez Analyser en mode push tous les référentiels.
    - Pour choisir des référentiels spécifiques à analyser en mode push, entrez un filtre basé sur les noms des référentiels, puis choisissez Ajouter un filtre.
6. Choisissez Enregistrer.
  7. Répétez ces étapes dans chaque région dans laquelle vous voulez activer l'analyse améliorée.

## AWS CLI

Utilisez la AWS CLI commande suivante pour activer l'analyse améliorée de votre registre privé à l'aide du AWS CLI. Vous pouvez spécifier des filtres d'analyse à l'aide de l'objet `rules`.

- [put-registry-scanning-configuration](#) (AWS CLI)

L'exemple suivant active l'analyse améliorée pour votre registre privé. Par défaut, lorsqu'aucune règle n'est spécifiée, Amazon ECR définit la configuration d'analyse sur une analyse continue pour tous les référentiels.

```
aws ecr put-registry-scanning-configuration \
 --scan-type ENHANCED \
 --region us-east-2
```

L'exemple suivant active l'analyse améliorée pour votre registre privé et spécifie un filtre d'analyse. Le filtre d'analyse dans l'exemple active l'analyse continue pour tous les référentiels avec prod dans leur nom.

```
aws ecr put-registry-scanning-configuration \
 --scan-type ENHANCED \
 --rules '[{"repositoryFilters" : [{"filter": "prod", "filterType" :
 "WILDCARD"}], "scanFrequency" : "CONTINUOUS_SCAN"}]' \
 --region us-east-2
```

L'exemple suivant active l'analyse améliorée pour votre registre privé et spécifie plusieurs filtres d'analyse. Les filtres d'analyse de l'exemple activent l'analyse continue pour tous les référentiels dont le nom contient prod et activent l'analyse lors de l'envoi (push) uniquement pour tous les autres référentiels.

```
aws ecr put-registry-scanning-configuration \
 --scan-type ENHANCED \
 --rules '[{"repositoryFilters" : [{"filter": "prod", "filterType" :
 "WILDCARD"}], "scanFrequency" : "CONTINUOUS_SCAN"}, {"repositoryFilters" :
 [{"filter": "*", "filterType" : "WILDCARD"}], "scanFrequency" : "SCAN_ON_PUSH"}]' \
 --region us-west-2
```

## EventBridge événements envoyés pour une analyse améliorée dans Amazon ECR

Lorsque le scan amélioré est activé, Amazon ECR envoie un événement EventBridge lorsque la fréquence d'analyse d'un référentiel est modifiée. Amazon Inspector envoie des événements EventBridge lorsqu'une numérisation initiale est terminée et lorsqu'un résultat de numérisation d'image est créé, mis à jour ou fermé.

## Événement pour un changement de fréquence d'analyse d'un référentiel

Lorsque l'analyse améliorée est activée pour votre registre, l'événement suivant est envoyé par Amazon ECR lorsqu'il y a un changement avec une ressource dont l'analyse améliorée est activée. Cela inclut la création de nouveaux référentiels, la modification de la fréquence d'analyse d'un référentiel ou la création ou la suppression d'images dans des référentiels dont l'analyse améliorée est activée. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR](#).

```
{
 "version": "0",
 "id": "0c18352a-a4d4-6853-ef53-0abEXAMPLE",
 "detail-type": "ECR Scan Resource Change",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2021-10-14T20:53:46Z",
 "region": "us-east-1",
 "resources": [],
 "detail": {
 "action-type": "SCAN_FREQUENCY_CHANGE",
 "repositories": [{
 "repository-name": "repository-1",
 "repository-arn": "arn:aws:ecr:us-east-1:123456789012:repository/repository-1",
 "scan-frequency": "SCAN_ON_PUSH",
 "previous-scan-frequency": "MANUAL"
 },
 {
 "repository-name": "repository-2",
 "repository-arn": "arn:aws:ecr:us-east-1:123456789012:repository/repository-2",
 "scan-frequency": "CONTINUOUS_SCAN",
 "previous-scan-frequency": "SCAN_ON_PUSH"
 },
 {
 "repository-name": "repository-3",
 "repository-arn": "arn:aws:ecr:us-east-1:123456789012:repository/repository-3",
 "scan-frequency": "CONTINUOUS_SCAN",
 "previous-scan-frequency": "SCAN_ON_PUSH"
 }
],
 "resource-type": "REPOSITORY",
 "scan-type": "ENHANCED"
}
```

```
}
```

### Événement pour une analyse initiale d'image (analyse améliorée)

Lorsque l'analyse améliorée est activée pour votre registre, l'événement suivant est envoyé par Amazon Inspector lorsque l'analyse initiale de l'image est terminée. Le paramètre `finding-severity-counts` ne retournera une valeur pour un niveau de gravité que s'il en existe un. Par exemple, si l'image ne contient pas de résultats au niveau CRITICAL, aucun nombre critique ne sera renvoyé. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités du système d'exploitation et des packages de langage de programmation dans Amazon ECR](#).

Modèle d'événement :

```
{
 "source": ["aws.inspector2"],
 "detail-type": ["Inspector2 Scan"]
}
```

Exemple de sortie :

```
{
 "version": "0",
 "id": "739c0d3c-4f02-85c7-5a88-94a9EXAMPLE",
 "detail-type": "Inspector2 Scan",
 "source": "aws.inspector2",
 "account": "123456789012",
 "time": "2021-12-03T18:03:16Z",
 "region": "us-east-2",
 "resources": [
 "arn:aws:ecr:us-east-2:123456789012:repository/amazon/amazon-ecs-sample"
],
 "detail": {
 "scan-status": "INITIAL_SCAN_COMPLETE",
 "repository-name": "arn:aws:ecr:us-east-2:123456789012:repository/amazon/amazon-ecs-sample",
 "finding-severity-counts": {
 "CRITICAL": 7,
 "HIGH": 61,
 "MEDIUM": 62,
 "TOTAL": 158
 }
 },
}
```

```

 "image-digest":
 "sha256:36c7b282abd0186e01419f2e58743e1bf635808231049bbc9d77e5EXAMPLE",
 "image-tags": [
 "latest"
]
 }
}

```

### Événement pour une mise à jour de découverte d'analyse d'image (analyse améliorée)

Lorsque l'analyse améliorée est activée pour votre registre, l'événement suivant est envoyé par Amazon Inspector lorsque la découverte d'analyse d'image est créée, mise à jour ou fermée. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités du système d'exploitation et des packages de langage de programmation dans Amazon ECR](#).

Modèle d'événement :

```

{
 "source": ["aws.inspector2"],
 "detail-type": ["Inspector2 Finding"]
}

```

Exemple de sortie :

```

{
 "version": "0",
 "id": "42dbea55-45ad-b2b4-87a8-afaEXAMPLE",
 "detail-type": "Inspector2 Finding",
 "source": "aws.inspector2",
 "account": "123456789012",
 "time": "2021-12-03T18:02:30Z",
 "region": "us-east-2",
 "resources": [
 "arn:aws:ecr:us-east-2:123456789012:repository/amazon/amazon-ecs-sample/sha256:36c7b282abd0186e01419f2e58743e1bf635808231049bbc9d77eEXAMPLE"
],
 "detail": {
 "awsAccountId": "123456789012",
 "description": "In libssh2 v1.9.0 and earlier versions, the SSH_MSG_DISCONNECT logic in packet.c has an integer overflow in a bounds check, enabling an attacker to specify an arbitrary (out-of-bounds) offset for a subsequent memory read. A crafted SSH server may be able to disclose sensitive information or cause a denial of service condition on the client system when a user connects to the server.",
 }
}

```

```
"findingArn": "arn:aws:inspector2:us-east-2:123456789012:finding/
be674aadd0f75ac632055EXAMPLE",
"firstObservedAt": "Dec 3, 2021, 6:02:30 PM",
"inspectorScore": 6.5,
"inspectorScoreDetails": {
 "adjustedCvss": {
 "adjustments": [],
 "cvssSource": "REDHAT_CVE",
 "score": 6.5,
 "scoreSource": "REDHAT_CVE",
 "scoringVector": "CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:N/A:N",
 "version": "3.0"
 }
},
"lastObservedAt": "Dec 3, 2021, 6:02:30 PM",
"packageVulnerabilityDetails": {
 "cvss": [
 {
 "baseScore": 6.5,
 "scoringVector": "CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:N/A:N",
 "source": "REDHAT_CVE",
 "version": "3.0"
 },
 {
 "baseScore": 5.8,
 "scoringVector": "AV:N/AC:M/Au:N/C:P/I:N/A:P",
 "source": "NVD",
 "version": "2.0"
 },
 {
 "baseScore": 8.1,
 "scoringVector": "CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:N/A:H",
 "source": "NVD",
 "version": "3.1"
 }
],
 "referenceUrls": [
 "https://access.redhat.com/errata/RHSA-2020:3915"
],
 "source": "REDHAT_CVE",
 "sourceUrl": "https://access.redhat.com/security/cve/CVE-2019-17498",
 "vendorCreatedAt": "Oct 16, 2019, 12:00:00 AM",
 "vendorSeverity": "Moderate",
 "vulnerabilityId": "CVE-2019-17498",
}
```

```
 "vulnerablePackages": [
 {
 "arch": "X86_64",
 "epoch": 0,
 "name": "libssh2",
 "packageManager": "OS",
 "release": "12.amzn2.2",
 "sourceLayerHash":
"sha256:72d97abdfae3b3c933ff41e39779cc72853d7bd9dc1e4800c5294dEXAMPLE",
 "version": "1.4.3"
 }
],
 "remediation": {
 "recommendation": {
 "text": "Update all packages in the vulnerable packages section to
their latest versions."
 }
 },
 "resources": [
 {
 "details": {
 "awsEcrContainerImage": {
 "architecture": "amd64",
 "imageHash":
"sha256:36c7b282abd0186e01419f2e58743e1bf635808231049bbc9d77e5EXAMPLE",
 "imageTags": [
 "latest"
],
 "platform": "AMAZON_LINUX_2",
 "pushedAt": "Dec 3, 2021, 6:02:13 PM",
 "lastInUseAt": "Dec 3, 2021, 6:02:13 PM",
 "inUseCount": 1,
 "registry": "123456789012",
 "repositoryName": "amazon/amazon-ecs-sample"
 }
 },
 "id": "arn:aws:ecr:us-east-2:123456789012:repository/amazon/amazon-ecs-
sample/sha256:36c7b282abd0186e01419f2e58743e1bf635808231049bbc9d77EXAMPLE",
 "partition": "N/A",
 "region": "N/A",
 "type": "AWS_ECR_CONTAINER_IMAGE"
 }
],
],
```

```
"severity": "MEDIUM",
"status": "ACTIVE",
"title": "CVE-2019-17498 - libssh2",
"type": "PACKAGE_VULNERABILITY",
"updatedAt": "Dec 3, 2021, 6:02:30 PM"
}
}
```

## Extraction des résultats pour des scans améliorés dans Amazon ECR

Vous pouvez récupérer les résultats de numérisation de la dernière numérisation d'image améliorée terminée, puis ouvrir les résultats dans Amazon Inspector pour obtenir plus de détails. Les vulnérabilités logicielles découvertes sont répertoriées par gravité sur la base de données Common Vulnerabilities and Exposures (CVEs).

Pour en savoir plus sur le dépannage de certains problèmes courants lors de la numérisation des images, consultez [Résolution des problèmes liés à la numérisation d'images dans Amazon ECR](#).

### AWS Management Console

Suivez les étapes ci-dessous pour récupérer les résultats de numérisation d'images à l'aide de la AWS Management Console.

Pour récupérer les résultats de numérisation d'images

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région où votre référentiel existe.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Dans la page Référentiels, choisissez le référentiel qui contient l'image pour laquelle récupérer les résultats de numérisation.
5. Sur la page Images, sous la colonne Image tag, sélectionnez la balise image pour récupérer les résultats de numérisation.
6. Pour obtenir plus de détails dans la console Amazon Inspector, choisissez le nom de la vulnérabilité dans la colonne Nom.

## AWS CLI

Utilisez la AWS CLI commande suivante pour récupérer les résultats de numérisation d'images à l'aide du AWS CLI. Vous pouvez spécifier une image à l'aide de `imageTag` ou `imageDigest`, qui peuvent être obtenus à l'aide de la commande CLI [list-images](#).

- [describe-image-scan-findings](#) (AWS CLI)

L'exemple suivant utilise une étiquette d'image.

```
aws ecr describe-image-scan-findings \
 --repository-name name \
 --image-id imageTag=tag_name \
 --region us-east-2
```

L'exemple suivant utilise un résumé d'image.

```
aws ecr describe-image-scan-findings \
 --repository-name name \
 --image-id imageDigest=sha256_hash \
 --region us-east-2
```

## Scannez les images pour détecter les vulnérabilités du système d'exploitation dans Amazon ECR

Le scan de base d'Amazon ECR utilise une technologie AWS native pour scanner les images de vos conteneurs afin de détecter les vulnérabilités logicielles. L'analyse de base permet de détecter les vulnérabilités sur un large éventail de systèmes d'exploitation courants, en fournissant plus de 50 flux de données afin de détecter les vulnérabilités et les expositions courantes (CVEs). Ces sources incluent les avis de sécurité des fournisseurs, les flux de données, les flux de renseignements sur les menaces, ainsi que la base de données nationale sur les vulnérabilités (NVD) et le MITRE.

La numérisation de base Amazon ECR est prise en charge dans toutes les régions répertoriées dans la section [AWS Services par région](#).

Amazon ECR utilise la gravité d'un CVE de la source de distribution en amont si disponible. Sinon, le score CVSS (Common Vulnerability Scoring System) est utilisé. Le score CVSS peut être utilisé

pour obtenir l'évaluation de gravité de la vulnérabilité NVD. Pour en savoir plus, consultez [NVD Vulnerability Gravity Ratings](#).

L'analyse de base d'Amazon ECR prend en charge les filtres pour spécifier les référentiels à analyser en mode push. Tous les référentiels qui ne correspondent pas à un filtre de numérisation en mode push sont définis sur la fréquence d'analyse manuelle, ce qui signifie que vous devez démarrer l'analyse manuellement. Une image peut être numérisée une fois toutes les 24 heures. Les 24 heures incluent le scan initial sur push, si configuré, et tous les scans manuels. Avec la numérisation de base, vous pouvez numériser jusqu'à 100 000 images par 24 heures dans un registre donné.

Vous pouvez récupérer les derniers résultats de numérisation d'image achevée pour chaque image. Lorsqu'une numérisation d'image est terminée, Amazon ECR envoie un événement à Amazon EventBridge. Pour de plus amples informations, veuillez consulter [Événements Amazon ECR et EventBridge](#).

## Support du système d'exploitation pour la numérisation de base

Pour des raisons de sécurité et pour garantir une couverture continue, nous vous recommandons de continuer à utiliser les versions prises en charge d'un système d'exploitation. Conformément à la politique du fournisseur, les systèmes d'exploitation abandonnés ne sont plus mis à jour avec des correctifs et, dans de nombreux cas, de nouveaux avis de sécurité ne sont plus publiés à leur sujet. En outre, certains fournisseurs suppriment les alertes de sécurité et les détections existantes de leurs flux lorsqu'un système d'exploitation concerné atteint la fin du support standard. Lorsqu'une distribution perd le support de son fournisseur, Amazon ECR peut ne plus prendre en charge l'analyse des vulnérabilités. Tous les résultats générés par Amazon ECR concernant un système d'exploitation abandonné doivent être utilisés à titre informatif uniquement. Pour obtenir la liste complète des systèmes d'exploitation et des versions pris en charge, consultez la section [Systèmes d'exploitation pris en charge - Amazon Inspector scan](#) dans le guide de l'utilisateur d'Amazon Inspector.

## Configuration de la numérisation de base pour les images dans Amazon ECR

Par défaut, Amazon ECR active le scan de base pour tous les registres privés. Par conséquent, à moins que vous n'ayez modifié les paramètres de numérisation de votre registre privé, il n'est pas nécessaire d'activer le scan de base.

Vous pouvez utiliser les étapes suivantes pour définir un ou plusieurs filtres de scan sur push.

## Pour activer le scan de base pour votre registre privé

1. [Ouvrez la console Amazon ECR sur private-registry/repositories https://console.aws.amazon.com/ecr/](https://console.aws.amazon.com/ecr/)
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez définir la configuration d'analyse.
3. Dans le volet de navigation, choisissez Registre privé, Numérisation.
4. Dans la page Scanning configuration (Configuration de l'analyse), pour Scan type (Type d'analyse), choisissez Basic scanning (Analyse de base).
5. Par défaut, tous vos référentiels sont configurés pour une analyse manuelle. Vous pouvez éventuellement configurer l'analyse lors de l'envoi (push) en spécifiant des filtres d'analyse lors de l'envoi (push). Vous pouvez définir l'analyse lors de l'envoi (push) pour tous les référentiels ou pour des référentiels individuels. Pour de plus amples informations, veuillez consulter [Filtres permettant de choisir les référentiels à analyser dans Amazon ECR](#).

### Note

Si le scan en mode push est activé pour un dépôt, des scans sont également effectués sur les images restaurées après avoir été archivées. Aucune ancienne numérisation ne sera disponible à partir de l'image restaurée.

## Numérisation manuelle d'une image pour détecter les vulnérabilités du système d'exploitation dans Amazon ECR

Si vos référentiels ne sont pas configurés pour numériser en mode push, vous pouvez lancer manuellement des numérisations d'images. Une image peut être numérisée une fois toutes les 24 heures. Les 24 heures incluent le scan initial sur push, si configuré, et tous les scans manuels.

Pour en savoir plus sur le dépannage de certains problèmes courants lors de la numérisation des images, consultez [Résolution des problèmes liés à la numérisation d'images dans Amazon ECR](#).

### AWS Management Console

Suivez les étapes suivantes pour lancer la numérisation manuelle d'une image à l'aide de la AWS Management Console.

1. [Ouvrez la console Amazon ECR sur private-registry/repositories https://console.aws.amazon.com/ecr/](https://console.aws.amazon.com/ecr/private-registry/repositories)
2. Dans la barre de navigation, choisissez la région dans laquelle vous souhaitez créer votre référentiel.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Dans la page Référentiels, choisissez le référentiel qui contient l'image à numériser.
5. Dans la page Images sélectionnez l'image à numériser, puis choisissez Numériser.

## AWS CLI

- [start-image-scan](#) (AWS CLI)

L'exemple suivant utilise une étiquette d'image.

```
aws ecr start-image-scan --repository-name name --image-id imageTag=tag_name --region us-east-2
```

L'exemple suivant utilise un résumé d'image.

```
aws ecr start-image-scan --repository-name name --image-id imageDigest=sha256_hash --region us-east-2
```

## AWS Tools for Windows PowerShell

- [Obtenez- ECRImage ScanFinding](#) (AWS Tools for Windows PowerShell)

L'exemple suivant utilise une étiquette d'image.

```
Start-ECRImageScan -RepositoryName name -ImageId_ImageTag tag_name -Region us-east-2 -Force
```

L'exemple suivant utilise un résumé d'image.

```
Start-ECRImageScan -RepositoryName name -ImageId_ImageDigest sha256_hash -Region us-east-2 -Force
```

## Extraction des résultats pour les scans de base dans Amazon ECR

Vous pouvez récupérer les résultats de numérisation de la dernière numérisation d'image de base terminée. Les vulnérabilités logicielles découvertes sont répertoriées par gravité sur la base de données Common Vulnerabilities and Exposures (CVEs).

Pour en savoir plus sur le dépannage de certains problèmes courants lors de la numérisation des images, consultez [Résolution des problèmes liés à la numérisation d'images dans Amazon ECR](#).

### AWS Management Console

Suivez les étapes ci-dessous pour récupérer les résultats de numérisation d'images à l'aide de la AWS Management Console.

Pour récupérer les résultats de numérisation d'images

1. [Ouvrez la console Amazon ECR sur private-registry/repositories https://console.aws.amazon.com/ecr/](https://console.aws.amazon.com/ecr/)
2. Dans la barre de navigation, choisissez la région dans laquelle vous souhaitez créer votre référentiel.
3. Dans le panneau de navigation, choisissez Référentiels.
4. Dans la page Référentiels, choisissez le référentiel qui contient l'image pour laquelle récupérer les résultats de numérisation.
5. Sur la page Images, sous la colonne Image tag, sélectionnez la balise image pour récupérer les résultats de numérisation.

### AWS CLI

Utilisez la AWS CLI commande suivante pour récupérer les résultats de numérisation d'images à l'aide du AWS CLI. Vous pouvez spécifier une image à l'aide de `imageTag` ou `imageDigest`, qui peuvent être obtenus à l'aide de la commande CLI [list-images](#).

- [describe-image-scan-findings](#) (AWS CLI)

L'exemple suivant utilise une étiquette d'image.

```
aws ecr describe-image-scan-findings --repository-name name --image-id
imageTag=tag_name --region us-east-2
```

L'exemple suivant utilise un résumé d'image.

```
aws ecr describe-image-scan-findings --repository-name name --image-id
imageDigest=sha256_hash --region us-east-2
```

AWS Tools for Windows PowerShell

- [Obtenez- ECRImage ScanFinding](#) (AWS Tools for Windows PowerShell)

L'exemple suivant utilise une étiquette d'image.

```
Get-ECRImageScanFinding -RepositoryName name -ImageId_ImageTag tag_name -
Region us-east-2
```

L'exemple suivant utilise un résumé d'image.

```
Get-ECRImageScanFinding -RepositoryName name -ImageId_ImageDigest sha256_hash -
Region us-east-2
```

## Résolution des problèmes liés à la numérisation d'images dans Amazon ECR

Les échecs de numérisation d'images suivants sont les plus courants. Vous pouvez afficher de telles erreurs dans la console Amazon ECR en affichant les détails de l'image, via l'API ou AWS CLI en utilisant l' `DescribeImageScanFindingsAPI`.

### UnsupportedImageError

Vous pouvez recevoir une erreur `UnsupportedImageError` lors de la tentative d'effectuer une analyse de base sur une image créée à l'aide d'un système d'exploitation pour lequel Amazon ECR ne prend pas en charge l'analyse de base d'images. Amazon ECR prend en charge l'analyse des vulnérabilités de paquets pour les principales versions des distributions Amazon Linux, Amazon Linux 2, Debian, Ubuntu, CentOS, Oracle Linux, Alpine et RHEL Linux. Une fois qu'une distribution perd le support de son fournisseur, Amazon ECR peut ne plus prendre en charge la recherche de vulnérabilités. Amazon ECR ne prend pas en charge l'analyse d'images créées à partir de l'image [Docker scratch](#).

**⚠ Important**

Lorsque vous utilisez l'analyse améliorée, Amazon Inspector prend en charge l'analyse pour des systèmes d'exploitation et des types de médias spécifiques. Pour obtenir la liste complète, veuillez consulter la rubrique [Systèmes d'exploitation et types de médias pris en charge](#) dans le Guide de l'utilisateur Amazon Inspector.

Un niveau de gravité UNDEFINED est renvoyé

Vous pouvez recevoir un résultat d'analyse dont le niveau de gravité est UNDEFINED. Les raisons les plus courantes sont les suivantes :

- La source CVE n'a pas attribué de priorité à la vulnérabilité.
- La vulnérabilité a reçu une priorité non reconnue par Amazon ECR.

Pour déterminer la gravité et la description d'une vulnérabilité, vous pouvez afficher le CVE directement à partir de la source.

## Présentation des statuts d'analyse **SCAN\_ELIGIBILITY\_EXPIRED**

Lorsque l'analyse améliorée à l'aide d'Amazon Inspector est activée pour votre registre privé et que vous consultez les vulnérabilités d'analyse, vous pouvez voir un état d'analyse de SCAN\_ELIGIBILITY\_EXPIRED. Les raisons les plus courantes sont les suivantes :

- Lorsque vous activez pour la première fois l'analyse améliorée pour votre registre privé, Amazon Inspector ne reconnaît que les images envoyées à Amazon ECR au cours des 30 derniers jours, en fonction de l'horodatage de l'image. Les images plus anciennes auront le statut d'analyse SCAN\_ELIGIBILITY\_EXPIRED. Si vous souhaitez que ces images soient analysées par Amazon Inspector, vous devez les envoyer à nouveau dans votre référentiel.
- Si la durée de ré-analyse ECR est modifiée dans la console Amazon Inspector et que ce délai est écoulé, l'état d'analyse de l'image devient inactive avec un code de motif de expired, et toutes les résultats associées à l'image sont programmées pour être fermées. Cela a pour effet que la console Amazon ECR indique l'état d'analyse comme SCAN\_ELIGIBILITY\_EXPIRED.

# Synchroniser un registre en amont avec un registre privé Amazon ECR

À l'aide des règles de cache d'extraction, vous pouvez synchroniser le contenu d'un registre en amont avec votre registre privé Amazon ECR.

Amazon ECR prend actuellement en charge la création de règles de cache d'extraction pour les registres en amont suivants :

- Amazon ECR Public, registre d'images de conteneurs Kubernetes et Quay (aucune authentification n'est requise)
- Docker Hub, Microsoft Azure Container Registry, GitHub Container Registry, GitLab Container Registry et Chainguard Registry (nécessite une authentification secrète) AWS Secrets Manager
- Amazon ECR (nécessite une authentification avec le rôle AWS IAM)

Pour GitLab Container Registry, Amazon ECR prend en charge l'extraction du cache uniquement avec GitLab l'offre Software as a Service (SaaS). Pour plus d'informations sur l'utilisation GitLab de l'offre SaaS, rendez-vous sur [GitLab.com](https://gitlab.com).

Pour les registres en amont qui nécessitent une authentification par secret (comme Docker Hub), vous devez stocker vos informations d'identification dans un AWS Secrets Manager secret. Vous pouvez utiliser la console Amazon ECR pour créer des secrets Secrets Manager pour chaque registre en amont authentifié. Pour plus d'informations sur la création d'un secret Secrets Manager à l'aide de la console Secrets Manager, consultez [Stockage AWS Secrets Manager secret des informations d'identification de votre référentiel en amont](#).

Pour Amazon ECR, vous devez créer un rôle IAM si les registres Amazon ECR en amont et en aval appartiennent à un compte différent. AWS Pour plus d'informations sur la création d'un rôle IAM, consultez [Politiques IAM requises pour que l'ECR entre comptes puisse extraire le cache de l'ECR](#).

Après avoir créé une règle de cache d'extraction pour le registre en amont, extrayez une image de ce registre en amont à l'aide de l'URI de votre registre privé Amazon ECR. Amazon ECR crée ensuite un référentiel et met cette image en cache dans votre registre privé. Pour les demandes d'extraction ultérieures de l'image mise en cache avec une balise donnée, Amazon ECR vérifie dans le registre en amont la présence d'une nouvelle version de l'image avec cette balise spécifique et tente de mettre à jour l'image dans votre registre privé au moins une fois toutes les 24 heures.

## Modèles de création de référentiels

Amazon ECR a ajouté la prise en charge des modèles de création de référentiels, ce qui vous permet de définir les configurations initiales pour les nouveaux référentiels créés par Amazon ECR en votre nom à l'aide de règles de cache d'extraction. Chaque modèle contient un préfixe d'espace de noms de référentiel qui est utilisé pour faire correspondre les nouveaux référentiels à un modèle spécifique. Les modèles peuvent spécifier la configuration de tous les paramètres du référentiel, y compris les politiques d'accès basées sur les ressources, l'immuabilité des balises, le chiffrement et les politiques de cycle de vie. Les paramètres d'un modèle de création de référentiel ne sont appliqués que lors de la création du référentiel et n'ont aucun effet sur les référentiels existants ou les référentiels créés à l'aide d'une autre méthode. Pour de plus amples informations, veuillez consulter [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).

## Considérations relatives à l'utilisation des règles du cache d'extraction

Tenez compte des points suivants lorsque vous utilisez les règles de cache d'extraction d'Amazon ECR.

- La création de règles de mise en cache par extraction n'est pas prise en charge dans les régions suivantes.
  - Chine (Beijing) (`cn-north-1`)
  - Chine (Ningxia) (`cn-northwest-1`)
  - AWS GovCloud (USA Est) (`us-gov-east-1`)
  - AWS GovCloud (US-Ouest) (`us-gov-west-1`)
- AWS Lambda ne prend pas en charge l'extraction d'images de conteneurs depuis Amazon ECR à l'aide d'une règle de cache d'extraction.
- Lors de l'extraction d'images à l'aide de la mise en cache par extraction, les points de terminaison de service FIPS d'Amazon ECR ne sont pas pris en charge la première fois qu'une image est extraite. L'utilisation des points de terminaison de service FIPS d'Amazon ECR fonctionne cependant pour les extractions suivantes.
- Pour les référentiels en amont qui nécessitent une authentification, lorsqu'une image est extraite via l'URI du registre privé Amazon ECR pour la première fois ou pour mettre à jour le cache, les extractions d'image sont lancées par l'utilisateur associé aux informations d'identification

configurées dans la règle de cache d'extraction. Les extractions suivantes renverront l'image directement depuis le cache dans le registre privé du client.

- Pour les référentiels en amont qui ne nécessitent pas d'authentification, lorsqu'une image est extraite via l'URI du registre privé Amazon ECR, les extractions d'image sont initiées par des AWS adresses IP.
- Lorsqu'un client extrait une image mise en cache via l'URI du registre privé Amazon ECR, Amazon ECR vérifie s'il a validé l'image par rapport au registre en amont au cours des dernières 24 heures. Si le délai de 24 heures a expiré, Amazon ECR envoie une demande en amont pour vérifier s'il existe une version plus récente et met à jour le cache s'il en existe une. Si la fenêtre n'a pas expiré, Amazon ECR diffuse l'image mise en cache sans contacter l'opérateur en amont.
- L'appel de l'`ListImageReferrersAPI` à un référentiel créé par un cache d'extraction renvoie les artefacts de référence conformes à l'OCI-cache dans le cache privé.
- Amazon ECR vérifie si les artefacts du référent ont été mis à jour au cours des 6 dernières heures. Si le délai de 6 heures a expiré, Amazon ECR envoie une demande en amont pour vérifier les nouvelles versions et met à jour le cache si elles existent.
- Si Amazon ECR ne peut pas mettre à jour l'image depuis le registre en amont pour une raison quelconque et que l'image est extraite, la dernière image mise en cache sera quand même extraite.
- Lors de la création du secret Secrets Manager qui contient les informations d'identification du registre en amont, le nom du secret doit utiliser le préfixe `ecr-pullthroughcache/`. Le secret doit également se trouver dans le même compte et dans la même région que ceux dans lesquels est créée la règle de mise en cache par extraction.
- Lorsqu'une image multi-architecture est extraite à l'aide d'une règle mise en cache par extraction, la liste de manifestes et chaque image référencée dans la liste de manifestes sont extraites vers le référentiel Amazon ECR. Si vous ne souhaitez extraire qu'une architecture spécifique, vous pouvez extraire l'image en utilisant le résumé d'image ou l'identification associée à l'architecture plutôt que l'identification associée à la liste de manifestes.
- Amazon ECR utilise un rôle IAM lié à un service, qui fournit les autorisations nécessaires à Amazon ECR pour créer le référentiel, récupérer la valeur du secret Secrets Manager pour l'authentification et transmettre l'image mise en cache en votre nom. Le rôle IAM lié à un service est créé automatiquement lorsqu'une règle de mise en cache par extraction est créée. Pour de plus amples informations, veuillez consulter [Rôle lié à un service Amazon ECR pour la mise en cache par extraction](#).
- Par défaut, l'utilisateur, le principal IAM qui extrait l'image mise en cache a les autorisations qui lui sont accordées par sa politique IAM. Vous pouvez utiliser la politique d'autorisations du

registre privé Amazon ECR pour étendre les autorisations d'une entité IAM. Pour de plus amples informations, veuillez consulter [Utilisation des autorisations de registre](#).

- Les référentiels Amazon ECR créés à l'aide du flux de travail de mise en cache par extraction sont traités comme tout autre référentiel Amazon ECR. Toutes les fonctions du référentiel, telles que la réplication et l'analyse d'images, sont prises en charge.
- Lorsqu'Amazon ECR crée un nouveau référentiel en votre nom à l'aide d'une action de mise en cache par extraction, les paramètres par défaut suivants sont appliqués au référentiel, sauf s'il existe un modèle de création de référentiel correspondant. Vous pouvez utiliser un modèle de création de référentiel pour définir les paramètres appliqués aux référentiels créés par Amazon ECR en votre nom. Pour de plus amples informations, veuillez consulter [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).
- Immuabilité des balises — L'immuabilité des balises indique si les balises d'image peuvent être remplacées. Par défaut, les balises d'image sont modifiables (elles peuvent être remplacées). Vous pouvez modifier le comportement des balises en configurant des filtres d'exclusion de balises soit dans la zone de texte Exclusion de balises mutables lorsque Mutable est sélectionné, soit dans la zone de texte Exclusion de balise immuable lorsque Immutable est sélectionné.
- Chiffrement — Le AES256 chiffrement par défaut est utilisé.
- Autorisations de dépôt : omises, aucune politique d'autorisation de dépôt n'est appliquée.
- Politique de cycle de vie : omise, aucune politique de cycle de vie n'est appliquée.
- Balises de ressource : omises, aucune balise de ressource n'est appliquée.
- L'activation de l'immuabilité des balises d'image pour des référentiels à l'aide d'une règle de mise en cache par extraction empêchera Amazon ECR de mettre à jour les images à l'aide de la même balise.
- Lorsqu'une image est extraite à l'aide de la règle du cache d'extraction pour la première fois, un itinéraire vers Internet peut être nécessaire. Dans certaines circonstances, un itinéraire vers Internet est nécessaire, il est donc préférable de configurer un itinéraire pour éviter toute défaillance. Ainsi, si vous avez configuré Amazon ECR pour utiliser un point de terminaison AWS PrivateLink VPC d'interface, vous devez vous assurer que le premier pull dispose d'un itinéraire vers Internet. Pour ce faire, vous pouvez créer un sous-réseau public dans le même VPC, avec une passerelle Internet, puis acheminer tout le trafic sortant vers Internet depuis leur sous-réseau privé vers le sous-réseau public. Les extractions d'image suivantes à l'aide de la règle de cache d'extraction ne l'exigent pas. Pour de plus amples informations, consultez [Exemples d'options de routage](#) dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

# Autorisations IAM requises pour synchroniser un registre en amont avec un registre privé Amazon ECR

Outre les autorisations d'API Amazon ECR nécessaires pour s'authentifier auprès d'un registre privé et pour transférer et extraire des images, les autorisations supplémentaires suivantes sont nécessaires pour utiliser efficacement les règles de mise en cache par extraction.

- `ecr:CreatePullThroughCacheRule` – Accorde l'autorisation de créer une règle de cache par extraction. Cette autorisation doit être accordée via une politique IAM basée sur l'identité.
- `ecr:BatchImportUpstreamImage` – Accorde l'autorisation de récupérer l'image externe et de l'importer dans votre registre privé. Cette autorisation peut être accordée à l'aide de la politique d'autorisation du registre privé, d'une politique IAM basée sur l'identité ou de la politique d'autorisations de référentiel basée sur les ressources. Pour plus d'informations sur l'utilisation d'autorisations de référentiel, consultez [Politiques relatives aux référentiels privés dans Amazon ECR](#).
- `ecr:CreateRepository` – Accorde l'autorisation de créer un référentiel dans un registre privé. Cette autorisation est requise si le référentiel stockant les images mises en cache n'existe pas déjà. Cette autorisation peut être accordée soit par une politique IAM basée sur l'identité, soit par la politique d'autorisation du registre privé.

## Utilisation des autorisations de registre

Les autorisations du registre privé Amazon ECR peuvent être utilisées pour étendre les autorisations des entités IAM individuelles à utiliser la mise en cache par extraction. Si une entité IAM dispose de plus d'autorisations accordées par une politique IAM que celles accordées par la politique d'autorisations de registre, la politique IAM a la priorité. Par exemple, si un utilisateur dispose des autorisations `ecr:*`, aucune autorisation supplémentaire n'est nécessaire au niveau du registre.

Pour créer une politique d'autorisations de registre privée (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région dans laquelle vous souhaitez configurer votre instruction d'autorisations de registre privé.
3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Registry permissions (Autorisations du registre).

4. Dans la page Registry permissions (Autorisations de registre), choisissez Generate statement (Générer une instruction).
5. Pour chaque instruction de politique d'autorisations de mise en cache par extraction que vous souhaitez créer, procédez comme suit.
  - a. Pour Policy type (Type de politique), choisissez Pull through cache policy (Politique de mise en cache par extraction).
  - b. Pour Statement id (ID d'instruction), fournissez un nom pour l'instruction de politique de mise en cache par extraction.
  - c. Pour IAM entities (Entités IAM), indiquez les utilisateurs, groupes ou rôles à inclure dans la politique.
  - d. Pour Repository namespace (Espace de noms de référentiel), sélectionnez la règle de mise en cache par extraction à laquelle associer la politique.
  - e. Pour Repository names (Noms de référentiel), spécifiez le nom de base du référentiel pour lequel appliquer la règle. Par exemple, si vous voulez spécifier le référentiel Amazon Linux sur Amazon ECR Public, le nom du référentiel sera `amazonlinux`.

Pour créer une politique d'autorisations de registre privée (AWS CLI)

Utilisez la AWS CLI commande suivante pour spécifier les autorisations de registre privé à l'aide du AWS CLI.

1. Créez un fichier local nommé `ptc-registry-policy.json` avec le contenu de la politique de référentiel. L'exemple suivant accorde à `ecr-pull-through-cache-user` l'autorisation de créer un référentiel et d'extraire une image depuis Amazon ECR Public, qui est la source en amont associée à la règle de cache par extraction précédemment créée.

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "PullThroughCacheFromReadOnlyRole",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:user/ecr-pull-through-cache-user"
 }
 }
]
}
```

```
 },
 "Action": [
 "ecr:CreateRepository",
 "ecr:BatchImportUpstreamImage"
],
 "Resource": "arn:aws:ecr:us-east-1:111122223333:repository/ecr-public/
*"
 }
]
```

### Important

L'autorisation `ecr-CreateRepository` n'est requise que si le référentiel stockant les images mises en cache n'existe pas déjà. Par exemple, si l'action de création du référentiel et les actions d'extraction d'image sont effectuées par des IAM principaux distincts tels qu'un administrateur et un développeur.

2. Utilisez la [put-registry-policy](#) commande pour définir la politique de registre.

```
aws ecr put-registry-policy \
 --policy-text file://ptc-registry.policy.json
```

## Étapes suivantes

Une fois que vous êtes prêt à commencer à utiliser les règles de mise en cache par extraction, procédez comme suit.

- Créez une règle de mise en cache par extraction. Pour de plus amples informations, veuillez consulter [Création d'une règle de cache d'extraction dans Amazon ECR](#).
- Créez un modèle de création de référentiel. Un modèle de création de référentiel vous permet de définir les paramètres à utiliser pour les nouveaux référentiels créés par Amazon ECR en votre nom lors d'une action de mise en cache par extraction. Pour de plus amples informations, veuillez consulter [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).

# Configuration des autorisations entre comptes ECR et ECR PTC

La fonction d'extraction du cache entre Amazon ECR et Amazon ECR (ECR to ECR) permet la synchronisation automatique des images entre les régions, les AWS comptes, ou les deux. Avec ECR to ECR PTC, vous pouvez transférer des images vers votre registre Amazon ECR principal et configurer une règle de cache d'extraction pour mettre en cache les images dans les registres Amazon ECR en aval.

## Politiques IAM requises pour que l'ECR entre comptes puisse extraire le cache de l'ECR

Pour mettre en cache des images entre les registres Amazon ECR de différents AWS comptes, créez un rôle IAM dans le compte en aval et configurez les politiques de cette section afin de fournir les autorisations suivantes :

- Amazon ECR a besoin d'autorisations pour extraire des images du registre Amazon ECR en amont en votre nom. Vous pouvez accorder ces autorisations en créant un rôle IAM, puis en le spécifiant dans votre règle de cache d'extraction.
- Le propriétaire du registre en amont doit également accorder au propriétaire du registre du cache les autorisations requises pour intégrer les images dans les politiques de ressources.

### Stratégies

- [Création d'un rôle IAM pour définir les autorisations du cache d'extraction](#)
- [Création d'une politique de confiance pour le rôle IAM](#)
- [Création d'une politique de ressources dans le registre Amazon ECR en amont](#)

## Création d'un rôle IAM pour définir les autorisations du cache d'extraction

L'exemple suivant montre une politique d'autorisation qui accorde à un rôle IAM l'autorisation d'extraire des images du registre Amazon ECR en amont en votre nom. Lorsqu'Amazon ECR assume le rôle, il reçoit les autorisations spécifiées dans cette politique.

### JSON

```
{
```

```
"Version": "2012-10-17",
"Statement": [
 {
 "Sid": "VisualEditor0",
 "Effect": "Allow",
 "Action": [
 "ecr:GetDownloadUrlForLayer",
 "ecr:GetAuthorizationToken",
 "ecr:BatchImportUpstreamImage",
 "ecr:BatchGetImage",
 "ecr:GetImageCopyStatus",
 "ecr:InitiateLayerUpload",
 "ecr:UploadLayerPart",
 "ecr:CompleteLayerUpload",
 "ecr:PutImage"
],
 "Resource": "*"
 }
]
```

## Création d'une politique de confiance pour le rôle IAM

L'exemple suivant montre une politique de confiance qui identifie le cache d'extraction Amazon ECR en tant que principal de AWS service habilité à assumer ce rôle.

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Principal": {
 "Service": "pullthroughcache.ecr.amazonaws.com"
 },
 "Action": "sts:AssumeRole"
 }
]
}
```

## Création d'une politique de ressources dans le registre Amazon ECR en amont

Le propriétaire du registre Amazon ECR en amont doit également ajouter une politique de registre ou une politique de référentiel pour accorder au propriétaire du registre en aval les autorisations requises pour effectuer les actions suivantes.

### Note

La politique de ressources suivante est requise uniquement pour les configurations de cache d'extraction ECR à ECR entre comptes. Pour le cache d'extraction entre régions pour un même compte, vous n'avez besoin que de la politique de rôle IAM et de la politique de confiance présentées dans les sections précédentes. L'autorisation principale du compte root n'est pas requise lorsque les registres en amont et en aval se trouvent dans le même AWS compte.

```
{
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::444455556666:root"
 },
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer",
 "ecr:BatchImportUpstreamImage",
 "ecr:GetImageCopyStatus"
],
 "Resource": "arn:aws:ecr:region:111122223333:repository/*"
}
```

## Création d'une règle de cache d'extraction dans Amazon ECR

Pour chaque registre en amont contenant des images que vous souhaitez mettre en cache dans votre registre privé Amazon ECR, vous devez créer une règle de cache d'extraction.

Pour les registres en amont qui nécessitent une authentification à l'aide de secrets, vous devez stocker les informations d'identification dans un secret Secrets Manager. Vous pouvez utiliser un secret existant ou en créer un nouveau. Vous pouvez créer le secret Secrets Manager dans la console Amazon ECR ou dans la console Secrets Manager. Pour créer un secret Secrets Manager

à l'aide de la console Secrets Manager au lieu de la console Amazon ECR, consultez [Stockage AWS Secrets Manager secret des informations d'identification de votre référentiel en amont](#).

## Conditions préalables

- Vérifiez que vous disposez des autorisations IAM appropriées pour créer des règles de cache d'extraction. Pour plus d'informations, consultez [Autorisations IAM requises pour synchroniser un registre en amont avec un registre privé Amazon ECR](#).
- Pour les registres en amont qui nécessitent une authentification à l'aide de secrets : si vous souhaitez utiliser un secret existant, vérifiez que le secret Secrets Manager répond aux exigences suivantes :
  - Le nom du secret commence par `ecr-pullthroughcache/`. Affiche AWS Management Console uniquement les secrets de Secrets Manager avec le `ecr-pullthroughcache/` préfixe.
  - Le compte et la région dans lesquels se trouve le secret doivent correspondre au compte et à la région dans lesquels se trouve la règle de cache d'extraction.

## Pour créer une règle de mise en cache par extraction (AWS Management Console)

Les étapes suivantes montrent comment créer une règle de mise en cache par extraction et un secret Secrets Manager à l'aide de la console Amazon ECR. Pour créer un secret à l'aide de la console Secrets Manager, consultez [Stockage AWS Secrets Manager secret des informations d'identification de votre référentiel en amont](#).

Pour Amazon ECR Public, le registre de conteneur Kubernetes ou Quay

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de votre registre privé.
3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Pull through cache (Mise en cache par extraction).
4. Sur la page Pull through cache configuration (Configuration de la mise en cache par extraction), cliquez sur Add rule (Ajouter une règle).
5. Sur la page Étape 1 : Spécifier une source pour Registre, choisissez Amazon ECR Public, Kubernetes ou Quay dans la liste des registres en amont, puis choisissez Suivant.

6. Sur la page **Étape 2 : Spécifier une destination pour le Préfixe du référentiel Amazon ECR**, spécifiez le préfixe d'espace de noms du référentiel à utiliser lors de la mise en cache des images extraites du registre public source, puis choisissez **Suivant**. Par défaut, un espace de noms est renseigné mais un espace de noms personnalisé peut également être spécifié.
7. Sur la page **Étape 3 : Vérifier et créer**, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez **Créer**.
8. Répétez l'étape précédente pour chaque mise en cache par extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.

## Pour Docker Hub

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de votre registre privé.
3. Dans le panneau de navigation, choisissez **Private registry (Registre privé)**, **Pull through cache (Mise en cache par extraction)**.
4. Sur la page **Pull through cache configuration (Configuration de la mise en cache par extraction)**, cliquez sur **Add rule (Ajouter une règle)**.
5. Sur la page **Étape 1 : Spécifier une source pour Registre**, choisissez **Docker Hub**, **Suivant**.
6. Sur la page **Étape 2 : Configuration de l'authentification pour les Informations d'identification en amont**, vous devez stocker vos informations d'identification pour Docker Hub dans un secret AWS Secrets Manager . Vous pouvez spécifier un secret existant ou utiliser la console Amazon ECR pour créer un nouveau secret.
  - a. Pour utiliser un secret existant, choisissez **Utiliser un AWS secret existant**. Pour **Nom du secret**, utilisez le menu déroulant pour sélectionner votre secret existant, puis choisissez **Suivant**.

### Note

Affiche AWS Management Console uniquement les secrets de Secrets Manager dont le nom utilise le `ecr-pullthroughcache/` préfixe. Le secret doit également se trouver dans le même compte et dans la même région que ceux dans lesquels est créée la règle de mise en cache par extraction.


- b. Pour créer un nouveau secret, choisissez **Créer un secret AWS**, procédez comme suit, puis choisissez **Suivant**.
  - i. Pour le **Nom du secret**, spécifiez un nom descriptif pour le secret. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode.
  - ii. Pour l'**e-mail Docker Hub**, spécifiez votre e-mail Docker Hub.
  - iii. Pour le **Jeton d'accès Docker Hub**, spécifiez votre jeton d'accès Docker Hub. Pour plus d'informations sur la création d'un jeton d'accès Docker Hub, consultez la section [Création et gestion des jetons d'accès](#) dans la documentation Docker.
7. Sur la page **Étape 3 : Spécifier une destination pour le Préfixe du référentiel Amazon ECR**, spécifiez l'espace de noms du référentiel à utiliser lors de la mise en cache des images extraites du registre public source, puis choisissez **Suivant**.

Par défaut, un espace de noms est renseigné mais un espace de noms personnalisé peut également être spécifié.
8. Sur la page **Étape 4 : Vérifier et créer**, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez **Créer**.
9. Répétez l'étape précédente pour chaque mise en cache par extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.

#### Pour le registre des GitHub conteneurs

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de votre registre privé.
3. Dans le panneau de navigation, choisissez **Private registry (Registre privé)**, **Pull through cache (Mise en cache par extraction)**.
4. Sur la page **Pull through cache configuration (Configuration de la mise en cache par extraction)**, cliquez sur **Add rule (Ajouter une règle)**.
5. À l'étape 1 : Spécifier une page source, pour **Registry**, choisissez **GitHub Container Registry**, **Next**.
6. Sur la page **Étape 2 : Configuration de l'authentification**, pour les informations d'identification **Upstream**, vous devez enregistrer vos informations d'authentification pour **GitHub Container Registry** dans un **AWS Secrets Manager secret**. Vous pouvez spécifier un secret existant ou utiliser la console Amazon ECR pour créer un nouveau secret.

- a. Pour utiliser un secret existant, choisissez Utiliser un AWS secret existant. Pour Nom du secret, utilisez le menu déroulant pour sélectionner votre secret existant, puis choisissez Suivant.

 Note

Affiche AWS Management Console uniquement les secrets de Secrets Manager dont le nom utilise le `ecr-pullthroughcache/` préfixe. Le secret doit également se trouver dans le même compte et dans la même région que ceux dans lesquels est créée la règle de mise en cache par extraction.

- b. Pour créer un nouveau secret, choisissez Créer un secret AWS , procédez comme suit, puis choisissez Suivant.
  - i. Pour le Nom du secret, spécifiez un nom descriptif pour le secret. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode.
  - ii. Pour le nom d'utilisateur du registre des GitHub conteneurs, spécifiez votre nom d'utilisateur du registre des GitHub conteneurs.
  - iii. Pour le GitHub jeton d'accès au registre des GitHub conteneurs, spécifiez votre jeton d'accès au registre des conteneurs. Pour plus d'informations sur la création d'un jeton d' GitHub accès, consultez [la section Gestion de vos jetons d'accès personnels](#) dans la GitHub documentation.
7. Sur la page Étape 3 : Spécifier une destination pour le Préfixe du référentiel Amazon ECR, spécifiez l'espace de noms du référentiel à utiliser lors de la mise en cache des images extraites du registre public source, puis choisissez Suivant.


Par défaut, un espace de noms est renseigné mais un espace de noms personnalisé peut également être spécifié.

8. Sur la page Étape 4 : Vérifier et créer, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez Créer.
9. Répétez l'étape précédente pour chaque mise en cache par extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.


## Pour Microsoft Azure Container Registry

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/>l'adresse.

2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de votre registre privé.
3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Pull through cache (Mise en cache par extraction).
4. Sur la page Pull through cache configuration (Configuration de la mise en cache par extraction), cliquez sur Add rule (Ajouter une règle).
5. Sur la page Étape 1 : Spécifier une source, procédez comme suit.
  - a. Pour Registre, choisissez Microsoft Azure Container Registry
  - b. Pour URL du registre source, spécifiez le nom de votre Microsoft Azure Container Registry, puis choisissez Suivant.

 Important

Il vous suffit de spécifier le préfixe, car le suffixe `.azurecr.io` est complété en votre nom.

6. Sur la page Étape 2 : Configuration de l'authentification pour les Informations d'identification en amont, vous devez stocker vos informations d'identification pour Microsoft Azure Container Registry dans un secret AWS Secrets Manager . Vous pouvez spécifier un secret existant ou utiliser la console Amazon ECR pour créer un nouveau secret.
    - a. Pour utiliser un secret existant, choisissez Utiliser un AWS secret existant. Pour Nom du secret, utilisez le menu déroulant pour sélectionner votre secret existant, puis choisissez Suivant.
-  Note
- Affiche AWS Management Console uniquement les secrets de Secrets Manager dont le nom utilise le `ecr-pullthroughcache/` préfixe. Le secret doit également se trouver dans le même compte et dans la même région que ceux dans lesquels est créée la règle de mise en cache par extraction.
- b. Pour créer un nouveau secret, choisissez Créer un secret AWS , procédez comme suit, puis choisissez Suivant.
    - i. Pour le Nom du secret, spécifiez un nom descriptif pour le secret. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode.

- ii. Pour le Nom d'utilisateur Microsoft Azure Container Registry, spécifiez votre nom d'utilisateur Microsoft Azure Container Registry.
  - iii. Pour le Jeton d'accès à Microsoft Azure Container Registry, spécifiez votre jeton d'accès à Microsoft Azure Container Registry. Pour plus d'informations sur la création d'un jeton d'accès à Microsoft Azure Container Registry, consultez la section [Créer un jeton - portail](#) dans la documentation Microsoft Azure.
7. Sur la page Étape 3 : Spécifier une destination pour le Préfixe du référentiel Amazon ECR, spécifiez l'espace de noms du référentiel à utiliser lors de la mise en cache des images extraites du registre public source, puis choisissez Suivant.


Par défaut, un espace de noms est renseigné mais un espace de noms personnalisé peut également être spécifié.

8. Sur la page Étape 4 : Vérifier et créer, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez Créer.
9. Répétez l'étape précédente pour chaque mise en cache par extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.

#### Pour le registre des GitLab conteneurs

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de votre registre privé.
3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Pull through cache (Mise en cache par extraction).
4. Sur la page Pull through cache configuration (Configuration de la mise en cache par extraction), cliquez sur Add rule (Ajouter une règle).
5. À l'étape 1 : Spécifier une page source, pour Registry, choisissez GitLab Container Registry, Next.
6. Sur la page Étape 2 : Configuration de l'authentification, pour les informations d'identification Upstream, vous devez enregistrer vos informations d'authentification pour GitLab Container Registry dans un AWS Secrets Manager secret. Vous pouvez spécifier un secret existant ou utiliser la console Amazon ECR pour créer un nouveau secret.
  - a. Pour utiliser un secret existant, choisissez Utiliser un AWS secret existant. Pour Nom du secret, utilisez le menu déroulant pour sélectionner votre secret existant, puis choisissez

Suivant. Pour plus d'informations sur la création d'un secret Secrets Manager à l'aide de la console Secrets Manager, consultez [Stockage AWS Secrets Manager secret des informations d'identification de votre référentiel en amont](#).

 Note

Affiche AWS Management Console uniquement les secrets de Secrets Manager dont le nom utilise le `ecr-pullthroughcache/` préfixe. Le secret doit également se trouver dans le même compte et dans la même région que ceux dans lesquels est créée la règle de mise en cache par extraction.


- b. Pour créer un nouveau secret, choisissez **Créer un secret AWS**, procédez comme suit, puis choisissez **Suivant**.
  - i. Pour le **Nom du secret**, spécifiez un nom descriptif pour le secret. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode.
  - ii. Pour le **nom d'utilisateur du registre des GitLab conteneurs**, spécifiez votre nom d'utilisateur du registre des GitLab conteneurs.
  - iii. Pour le **GitLab jeton d'accès au registre des GitLab conteneurs**, spécifiez votre jeton d'accès au registre des conteneurs. Pour plus d'informations sur la création d'un jeton d'accès au registre des GitLab conteneurs, voir [Jetons d'accès personnels, jetons d'accès de groupe](#) ou [jetons d'accès au projet](#) dans la GitLab documentation.
7. Sur la page **Étape 3 : Spécifier une destination pour le Préfixe du référentiel Amazon ECR**, spécifiez l'espace de noms du référentiel à utiliser lors de la mise en cache des images extraites du registre public source, puis choisissez **Suivant**.

Par défaut, un espace de noms est renseigné mais un espace de noms personnalisé peut également être spécifié.
8. Sur la page **Étape 4 : Vérifier et créer**, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez **Créer**.
9. Répétez l'étape précédente pour chaque mise en cache par extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.

Pour le registre privé Amazon ECR intégré à votre compte AWS

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/>l'adresse.

2. Dans la barre de navigation, choisissez la région dans laquelle vous souhaitez configurer les paramètres de votre registre privé.
3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Pull through cache (Mise en cache par extraction).
4. Sur la page Pull through cache configuration (Configuration de la mise en cache par extraction), cliquez sur Add rule (Ajouter une règle).
5. Sur la page Étape 1 : Spécifier en amont, pour Registry, choisissez Amazon ECR Private et This account. Pour Région, sélectionnez la région pour le registre Amazon ECR en amont, puis choisissez Next.
6. Sur la page Étape 2 : Spécifier les espaces de noms, pour l'espace de noms du cache, choisissez de créer des référentiels de cache extractibles avec un préfixe spécifique ou sans préfixe. Si vous sélectionnez Un préfixe spécifique, vous devez spécifier un nom de préfixe à utiliser dans le cadre de l'espace de noms pour la mise en cache des images depuis le registre en amont.
7. Pour l'espace de noms Upstream, choisissez si vous souhaitez extraire un préfixe spécifique qui existe dans le registre en amont. Si vous ne sélectionnez aucun préfixe, vous pouvez effectuer une extraction depuis n'importe quel dépôt du registre en amont. Spécifiez le préfixe du référentiel en amont si vous y êtes invité, puis choisissez Next.

 Note


Pour en savoir plus sur la personnalisation du cache et des espaces de noms en amont, consultez. [Personnalisation des préfixes de référentiel pour l'ECR vers le cache d'extraction ECR](#)

8. Sur la page Étape 3 : Vérifier et créer, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez Créer.
9. Répétez ces étapes pour chaque cache d'extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.

### Pour le registre privé Amazon ECR depuis un autre compte AWS


1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de votre registre privé.

3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Pull through cache (Mise en cache par extraction).
4. Sur la page Pull through cache configuration (Configuration de la mise en cache par extraction), cliquez sur Add rule (Ajouter une règle).
5. Sur la page Étape 1 : Spécifier en amont, pour Registry, choisissez Amazon ECR Private and Cross account. Pour Région, sélectionnez la région pour le registre Amazon ECR en amont. Pour Account, spécifiez l'ID de AWS compte pour le registre Amazon ECR en amont, puis choisissez Next.
6. Sur la page Étape 2 : Spécifier les autorisations, pour le rôle IAM, sélectionnez un rôle à utiliser pour l'accès au cache par extraction entre comptes, puis choisissez Créer.

 Note

Assurez-vous de sélectionner le rôle IAM qui utilise les autorisations créées dans [Politiques IAM requises pour que l'ECR entre comptes puisse extraire le cache de l'ECR](#).

7. Sur la page Étape 3 : Spécifier les espaces de noms, pour l'espace de noms du cache, choisissez de créer des référentiels de cache extractibles avec un préfixe spécifique ou sans préfixe. Si vous sélectionnez Un préfixe spécifique, vous devez spécifier un nom de préfixe à utiliser dans le cadre de l'espace de noms pour la mise en cache des images depuis le registre en amont.
8. Pour l'espace de noms Upstream, choisissez si vous souhaitez extraire un préfixe spécifique qui existe dans le registre en amont. Si vous ne sélectionnez aucun préfixe, vous pouvez effectuer une extraction depuis n'importe quel dépôt du registre en amont. Spécifiez le préfixe du référentiel en amont si vous y êtes invité, puis choisissez Next.

 Note

Pour en savoir plus sur la personnalisation du cache et des espaces de noms en amont, consultez. [Personnalisation des préfixes de référentiel pour l'ECR vers le cache d'extraction ECR](#)

9. Sur la page Étape 4 : Vérifier et créer, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez Créer.

10. Répétez ces étapes pour chaque cache d'extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.

### Pour Chainguard Registry

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/>l'adresse.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de votre registre privé.
3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Pull through cache (Mise en cache par extraction).
4. Sur la page Pull through cache configuration (Configuration de la mise en cache par extraction), cliquez sur Add rule (Ajouter une règle).
5. À l'étape 1 : Spécifiez une page source, pour Registry, choisissez Chainguard Registry, Next.
6. Sur la page Étape 2 : Configurer l'authentification, pour les informations d'identification Upstream, vous devez stocker vos informations d'authentification pour Chainguard Registry dans un AWS Secrets Manager secret. Vous pouvez spécifier un secret existant ou utiliser la console Amazon ECR pour créer un nouveau secret.
  - a. Pour utiliser un secret existant, choisissez Utiliser un AWS secret existant. Pour Nom du secret, utilisez le menu déroulant pour sélectionner votre secret existant, puis choisissez Suivant. Pour plus d'informations sur la création d'un secret Secrets Manager à l'aide de la console Secrets Manager, consultez [Stockage AWS Secrets Manager secret des informations d'identification de votre référentiel en amont](#).
- b. Pour créer un nouveau secret, choisissez Créer un secret AWS , procédez comme suit, puis choisissez Suivant.
  - i. Pour le Nom du secret, spécifiez un nom descriptif pour le secret. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode.

#### Note

Affiche AWS Management Console uniquement les secrets de Secrets Manager dont le nom utilise le `ecr-pullthroughcache/` préfixe. Le secret doit également se trouver dans le même compte et dans la même région que ceux dans lesquels est créée la règle de mise en cache par extraction.

- ii. Pour le nom d'utilisateur Chainguard Registry, spécifiez votre nom d'utilisateur Chainguard Registry.
  - iii. Pour le jeton d'extraction du registre Chainguard, spécifiez votre jeton d'extraction du registre Chainguard. Pour plus d'informations sur la création d'un jeton d'extraction du registre Chainguard, voir [Authentification avec un jeton de traction](#) dans la documentation de Chainguard.
7. À l'étape 3 : Spécifiez une page de destination, pour le préfixe du référentiel Amazon ECR, spécifiez l'espace de noms du référentiel à utiliser lors de la mise en cache des images extraites du registre source, puis choisissez Next.  
  
Par défaut, un espace de noms est renseigné mais un espace de noms personnalisé peut également être spécifié.
8. Sur la page Étape 4 : Vérifier et créer, vérifiez la configuration de la règle de mise en cache par extraction, puis choisissez Créer.
9. Répétez l'étape précédente pour chaque mise en cache par extraction que vous souhaitez créer. Les règles de mise en cache par extraction sont créées séparément pour chaque région.

## Pour créer une règle de mise en cache par extraction (AWS CLI)

Utilisez la AWS CLI commande [create-pull-through-cache-rule](#) pour créer une règle de cache d'extraction pour un registre privé Amazon ECR. Pour les registres en amont qui nécessitent une authentification à l'aide de secrets, vous devez stocker les informations d'identification dans un secret de Secrets Manager. Pour créer un secret à l'aide de la console Secrets Manager, consultez [Stockage AWS Secrets Manager secret des informations d'identification de votre référentiel en amont](#).

Les exemples suivants sont fournis pour chaque registre en amont pris en charge.

### Pour Amazon ECR Public

L'exemple suivant crée une règle de mise en cache par extraction pour le registre public Amazon ECR. Il spécifie un préfixe de référentiel de `ecr-public`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `ecr-public/upstream-repository-name`.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix ecr-public \
 --upstream-registry-url public.ecr.aws \
 --cache-locations us-east-1,eu-west-1,eu-central-1,eu-north-1,eu-south-1,eu-south-2,eu-west-2,eu-west-3,eu-west-4,eu-west-7,eu-west-8,eu-west-9,eu-west-10,eu-west-11,eu-west-12,eu-west-13,eu-west-14,eu-west-15,eu-west-16,eu-west-17,eu-west-18,eu-west-19,eu-west-20,eu-west-21,eu-west-22,eu-west-23,eu-west-24,eu-west-25,eu-west-26,eu-west-27,eu-west-28,eu-west-29,eu-west-30,eu-west-31,eu-west-32,eu-west-33,eu-west-34,eu-west-35,eu-west-36,eu-west-37,eu-west-38,eu-west-39,eu-west-40,eu-west-41,eu-west-42,eu-west-43,eu-west-44,eu-west-45,eu-west-46,eu-west-47,eu-west-48,eu-west-49,eu-west-50,eu-west-51,eu-west-52,eu-west-53,eu-west-54,eu-west-55,eu-west-56,eu-west-57,eu-west-58,eu-west-59,eu-west-60,eu-west-61,eu-west-62,eu-west-63,eu-west-64,eu-west-65,eu-west-66,eu-west-67,eu-west-68,eu-west-69,eu-west-70,eu-west-71,eu-west-72,eu-west-73,eu-west-74,eu-west-75,eu-west-76,eu-west-77,eu-west-78,eu-west-79,eu-west-80,eu-west-81,eu-west-82,eu-west-83,eu-west-84,eu-west-85,eu-west-86,eu-west-87,eu-west-88,eu-west-89,eu-west-90,eu-west-91,eu-west-92,eu-west-93,eu-west-94,eu-west-95,eu-west-96,eu-west-97,eu-west-98,eu-west-99,eu-west-100
```

```
--region us-east-2
```

## Pour Kubernetes Container Registry

L'exemple suivant crée une règle de mise en cache par extraction pour le registre public Kubernetes. Il spécifie un préfixe de référentiel de `kubernetes`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `kubernetes/upstream-repository-name`.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix kubernetes \
 --upstream-registry-url registry.k8s.io \
 --region us-east-2
```

## Pour Quay

L'exemple suivant crée une règle de mise en cache par extraction pour le registre public Quay. Il spécifie un préfixe de référentiel de `quay`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `quay/upstream-repository-name`.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix quay \
 --upstream-registry-url quay.io \
 --region us-east-2
```

## Pour Docker Hub

L'exemple suivant crée une règle de mise en cache par extraction pour le registre Docker Hub. Il spécifie un préfixe de référentiel de `docker-hub`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `docker-hub/upstream-repository-name`. Vous devez spécifier l'Amazon Resource Name (ARN) complet du secret contenant vos informations d'identification Docker Hub.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix docker-hub \
 --upstream-registry-url registry-1.docker.io \
 --credential-arn arn:aws:secretsmanager:us-east-2:111122223333:secret:ecr-
pullthroughcache/example1234 \
 --region us-east-2
```

## Pour le registre des GitHub conteneurs

L'exemple suivant crée une règle de cache d'extraction pour le registre des GitHub conteneurs. Il spécifie un préfixe de référentiel de `github`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `github/upstream-repository-name`. Vous devez spécifier le nom Amazon Resource Name (ARN) complet du secret contenant vos informations d'identification du registre des GitHub conteneurs.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix github \
 --upstream-registry-url ghcr.io \
 --credential-arn arn:aws:secretsmanager:us-east-2:111122223333:secret:ecr-pullthroughcache/example1234 \
 --region us-east-2
```

## Pour Microsoft Azure Container Registry

L'exemple suivant crée une règle de cache d'extraction pour le Microsoft Azure Container Registry. Il spécifie un préfixe de référentiel de `azure`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `azure/upstream-repository-name`. Vous devez spécifier l'Amazon Resource Name (ARN) complet du secret contenant vos informations d'identification Azure Container Registry.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix azure \
 --upstream-registry-url myregistry.azurecr.io \
 --credential-arn arn:aws:secretsmanager:us-east-2:111122223333:secret:ecr-pullthroughcache/example1234 \
 --region us-east-2
```

## Pour le registre des GitLab conteneurs

L'exemple suivant crée une règle de cache d'extraction pour le registre des GitLab conteneurs. Il spécifie un préfixe de référentiel de `gitlab`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `gitlab/upstream-repository-name`. Vous devez spécifier le nom Amazon Resource Name (ARN) complet du secret contenant vos informations d'identification du registre des GitLab conteneurs.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix gitlab \
 --credential-arn arn:aws:secretsmanager:us-east-2:111122223333:secret:ecr-pullthroughcache/example1234 \
 --region us-east-2
```

```
--upstream-registry-url registry.gitlab.com \
--credential-arn arn:aws:secretsmanager:us-east-2:111122223333:secret:ecr-
pullthroughcache/example1234 \
--region us-east-2
```

Pour le registre privé Amazon ECR intégré à votre compte AWS

L'exemple suivant crée une règle de cache d'extraction pour le registre privé Amazon ECR pour plusieurs régions au sein du même AWS compte. Il spécifie un préfixe de référentiel de `ecr`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `ecr/upstream-repository-name`.

```
aws ecr create-pull-through-cache-rule \
--ecr-repository-prefix ecr \
--upstream-registry-url aws_account_id.dkr.ecr.region.amazonaws.com \
--region us-east-2
```

Pour le registre privé Amazon ECR depuis un autre compte AWS

L'exemple suivant crée une règle de cache d'extraction pour le registre privé Amazon ECR pour plusieurs régions au sein du même AWS compte. Il spécifie un préfixe de référentiel de `ecr`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `ecr/upstream-repository-name`. Vous devez spécifier le nom Amazon Resource (ARN) complet du rôle IAM avec les autorisations créées dans [Création d'une règle de cache d'extraction dans Amazon ECR](#).

```
aws ecr create-pull-through-cache-rule \
--ecr-repository-prefix ecr \
--upstream-registry-url aws_account_id.dkr.ecr.region.amazonaws.com \
--custom-role-arn arn:aws:iam::aws_account_id:role/example-role \
--region us-east-2
```

Pour Chainguard Registry

L'exemple suivant crée une règle de cache d'extraction pour le registre Chainguard. Il spécifie un préfixe de référentiel de `chainguard`, ce qui fait que chaque référentiel créé à l'aide de la règle de mise en cache par extraction aura le schéma de dénomination de `chainguard/upstream-repository-name`. Vous devez spécifier le nom complet de la ressource Amazon (ARN) du secret contenant vos informations d'identification du registre Chainguard.

```
aws ecr create-pull-through-cache-rule \
 --ecr-repository-prefix chainguard \
 --upstream-registry-url cgr.dev \
 --credential-arn arn:aws:secretsmanager:us-east-2:111122223333:secret:ecr-
pullthroughcache/example1234 \
 --region us-east-2
```

## Étapes suivantes

Après avoir créé vos règles de cache d'extraction, voici les étapes suivantes :

- Créez un modèle de création de référentiel. Un modèle de création de référentiel vous permet de définir les paramètres à utiliser pour les nouveaux référentiels créés par Amazon ECR en votre nom lors d'une action de mise en cache par extraction. Pour de plus amples informations, veuillez consulter [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).
- Validez vos règles de mise en cache par extraction. Lors de la validation d'une règle de mise en cache par extraction, Amazon ECR établit une connexion réseau avec le registre en amont, vérifie qu'il peut accéder au secret Secrets Manager contenant les informations d'identification du registre en amont et que l'authentification a été réussie. Pour de plus amples informations, veuillez consulter [Validation des règles de cache d'extraction dans Amazon ECR](#).
- Commencez à utiliser vos règles de mise en cache par extraction. Pour de plus amples informations, veuillez consulter [Extraction d'une image à l'aide d'une règle de cache d'extraction dans Amazon ECR](#).

## Validation des règles de cache d'extraction dans Amazon ECR

Après avoir créé une règle de cache d'extraction, pour les registres en amont qui nécessitent une authentification, vous pouvez vérifier que la règle fonctionne correctement. Lors de la validation d'une règle de cache d'extraction, Amazon ECR établit une connexion réseau avec le registre en amont, vérifie qu'il peut accéder au secret Secrets Manager contenant les informations d'identification du registre en amont et vérifie que l'authentification a réussi.

Avant de commencer à utiliser vos règles de cache d'extraction, vérifiez que vous disposez des autorisations IAM appropriées. Pour de plus amples informations, veuillez consulter [Autorisations IAM requises pour synchroniser un registre en amont avec un registre privé Amazon ECR](#).

## Pour valider une règle de mise en cache par extraction (AWS Management Console)

Les étapes suivantes montrent comment valider une règle de mise en cache par extraction à l'aide de la console Amazon ECR.

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région contenant la règle de mise en cache par extraction à valider.
3. Dans le panneau de navigation, choisissez Private registry (Registre privé), Pull through cache (Mise en cache par extraction).
4. Sur la page Configuration de la mise en cache par extraction, sélectionnez la règle de mise en cache par extraction à valider. Utilisez ensuite le menu déroulant Actions et choisissez Afficher les détails.
5. Sur la page des détails de la règle de mise en cache par extraction, utilisez le menu déroulant Actions et choisissez Vérifier l'authentification. Amazon ECR affichera une bannière avec le résultat.
6. Répétez ces étapes pour chaque règle de mise en cache par extraction que vous souhaitez valider.

## Pour valider une règle de mise en cache par extraction (AWS CLI)

La AWS CLI commande [validate-pull-through-cache-rule](#) est utilisée pour valider une règle de cache d'extraction pour un registre privé Amazon ECR. L'exemple suivant utilise le préfixe d'espace de noms `ecr-public`. Remplacez cette valeur par la valeur du préfixe de la règle de mise en cache par extraction à valider.

```
aws ecr validate-pull-through-cache-rule \
 --ecr-repository-prefix ecr-public \
 --region us-east-2
```

Dans la réponse, le paramètre `isValid` indique si la validation a réussi ou non. Si `true`, Amazon ECR a pu accéder au registre en amont et l'authentification a réussi. Si `false`, un problème est survenu et la validation a échoué. Le paramètre `failure` indique la cause.

# Extraction d'une image à l'aide d'une règle de cache d'extraction dans Amazon ECR

Les exemples suivants montrent la syntaxe de commande à utiliser lors de l'extraction d'une image à l'aide d'une règle de mise en cache par extraction. Si vous recevez une erreur lors de l'extraction d'une image en amont à l'aide d'une règle de cache par extraction, consultez [Résolution des problèmes liés au cache d'extraction dans Amazon ECR](#) pour connaître les erreurs les plus courantes et leurs solutions.

Avant de commencer à utiliser vos règles de cache d'extraction, vérifiez que vous disposez des autorisations IAM appropriées. Pour de plus amples informations, veuillez consulter [Autorisations IAM requises pour synchroniser un registre en amont avec un registre privé Amazon ECR](#).

## Note

Les exemples suivants utilisent les valeurs d'espace de noms du référentiel Amazon ECR par défaut qu'il utilise. AWS Management Console Assurez-vous que vous utilisez l'URI du référentiel privé Amazon ECR que vous avez configuré.

## Pour Amazon ECR Public

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/ecr-public/repository_name/
image_name:tag
```

## Registre de conteneurs Kubernetes

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/kubernetes/repository_name/
image_name:tag
```

## Quay

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/quay/repository_name/
image_name:tag
```

## Docker Hub

Pour les images officielles de Docker Hub :

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/docker-hub/
library/image_name:tag
```

### Note

Pour les images officielles de Docker Hub, le préfixe `/library` doit être inclus. Pour tous les autres référentiels Docker Hub, vous devez omettre le préfixe `/library`.

Pour toutes les autres images de Docker Hub :

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/docker-hub/repository_name/
image_name:tag
```

## GitHub Registre des conteneurs

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/github/repository_name/
image_name:tag
```

## Microsoft Azure Container Registry

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/azure/repository_name/
image_name:tag
```

## GitLab Registre des conteneurs

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/gitlab/repository_name/
image_name:tag
```

## Registre Chainguard

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/chainguard/repository_name/
image_name:tag
```

# Stockage AWS Secrets Manager secret des informations d'identification de votre référentiel en amont

Lorsque vous créez une règle de mise en cache par extraction pour un référentiel en amont qui nécessite une authentification, vous devez stocker les informations d'identification dans un secret Secrets Manager. L'utilisation d'un secret Secrets Manager peut entraîner des frais. Pour en savoir plus, consultez [Pricing AWS Secrets Manager](#) (Tarification).

Les procédures suivantes expliquent comment créer un secret Secrets Manager pour chaque référentiel en amont pris en charge. Vous pouvez éventuellement utiliser le flux de travail de création de règles de mise en cache par extraction dans la console Amazon ECR pour créer le secret au lieu de le créer à l'aide de la console Secrets Manager. Pour de plus amples informations, veuillez consulter [Création d'une règle de cache d'extraction dans Amazon ECR](#).

## Docker Hub

Pour créer un secret Secrets Manager pour vos informations d'identification Docker Hub (AWS Management Console)


1. Ouvrez la console Secrets Manager à l'adresse <https://console.aws.amazon.com/secretsmanager/>.
2. Choisissez Store a new secret (Stocker un nouveau secret).
3. Sur la page Choisir un type de secret, procédez comme suit.
  - a. Pour Secret type (Type de secret), choisissez Other type of secret (Autre type de secret).
  - b. Dans les paires clé/valeur, créez deux lignes pour vos informations d'identification Docker Hub. Vous pouvez stocker jusqu'à 65 536 octets dans le secret.
    - i. Pour la première key/value paire, spécifiez `username` comme clé et votre nom d'utilisateur Docker Hub comme valeur.
    - ii. Pour la deuxième key/value paire, spécifiez `accessToken` comme clé et votre jeton d'accès Docker Hub comme valeur. Pour plus d'informations sur la création d'un jeton d'accès Docker Hub, consultez la section [Création et gestion des jetons d'accès](#) dans la documentation Docker.
  - c. Pour la clé de chiffrement, conservez la valeur par défaut de AWS KMS key `aws/secretsmanager`, puis choisissez Suivant. L'utilisation de cette clé n'entraîne aucun coût.

Pour plus d'informations, consultez la section [Chiffrement et déchiffrement de secret dans Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .

 Important

Vous devez utiliser la clé de chiffrement par défaut `aws/secretsmanager` pour chiffrer votre secret. Amazon ECR ne prend pas en charge l'utilisation d'une clé gérée par le client (CMK) pour cela.

4. Sur la page Configurer le secret, procédez comme suit.
  - a. Saisissez un Secret name (Nom de secret) descriptif et une Description. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode et être préfixés par `ecr-pullthroughcache/`.

 Important

Amazon ECR affiche AWS Management Console uniquement les secrets de Secrets Manager dont les noms utilisent le `ecr-pullthroughcache/` préfixe.

- b. (Facultatif) Dans la section Tags (Balises), vous pouvez ajouter une ou plusieurs balises à votre secret. Pour les stratégies de balisage, consultez la [Balise secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Ne stockez pas les informations sensibles dans des balises car elles ne sont pas chiffrées.
    - c. (Facultatif) Dans Resource permissions (Permission de la ressource), pour ajouter une stratégie de ressources à votre secret, choisissez Edit permissions (Modification des autorisations). Pour plus d'informations, consultez la rubrique [Attacher une politique d'autorisation à un secret Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .
    - d. (Facultatif) Dans Répliquer le secret, pour répliquer votre secret vers un autre Région AWS, choisissez Répliquer le secret. Vous pouvez reproduire votre secret maintenant ou revenir et le répliquer ultérieurement. Pour plus d'informations, consultez la rubrique [Réplication d'un secret vers d'autres régions](#) dans le Guide de l'utilisateur AWS Secrets Manager .
    - e. Choisissez Suivant.
5. (Facultatif) Dans la page Configure rotation (Configurer la rotation), vous pouvez activer la rotation automatique. Vous pouvez également garder la rotation désactivée pour l'instant,

puis l'activer plus tard. Pour plus d'informations, consultez la rubrique [Rotation des secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Choisissez Suivant.

6. Dans la page Review (Révision), passez en revue vos paramètres, puis choisissez Store (Stocker).

Secrets Manager revient à la liste des secrets. Si votre nouveau secret n'apparaît pas, choisissez le bouton d'actualisation.

## GitHub Container Registry

Pour créer un secret Secrets Manager pour vos informations d'identification du registre des GitHub conteneurs (AWS Management Console)

1. Ouvrez la console Secrets Manager à l'adresse <https://console.aws.amazon.com/secretsmanager/>.
2. Choisissez Store a new secret (Stocker un nouveau secret).
3. Sur la page Choisir un type de secret, procédez comme suit.
  - a. Pour Secret type (Type de secret), choisissez Other type of secret (Autre type de secret).
  - b. Dans les paires clé/valeur, créez deux lignes pour vos GitHub informations d'identification. Vous pouvez stocker jusqu'à 65 536 octets dans le secret.
    - i. Pour la première key/value paire, spécifiez `username` comme clé et votre GitHub nom d'utilisateur comme valeur.
    - ii. Pour la deuxième key/value paire, spécifiez `accessToken` comme clé et votre jeton GitHub d'accès comme valeur. Pour plus d'informations sur la création d'un jeton d'GitHub accès, consultez [la section Gestion de vos jetons d'accès personnels](#) dans la GitHub documentation.
  - c. Pour la clé de chiffrement, conservez la valeur par défaut de AWS KMS key `aws/secretsmanager`, puis choisissez Suivant. L'utilisation de cette clé n'entraîne aucun coût. Pour plus d'informations, consultez la section [Chiffrement et déchiffrement de secret dans Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .

**⚠ Important**

Vous devez utiliser la clé de chiffrement par défaut `aws/secretsmanager` pour chiffrer votre secret. Amazon ECR ne prend pas en charge l'utilisation d'une clé gérée par le client (CMK) pour cela.

4. Sur la page Configure secret (Configurer le secret), procédez comme suit :
  - a. Saisissez un Secret name (Nom de secret) descriptif et une Description. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode et être préfixés par `ecr-pullthroughcache/`.

**⚠ Important**

Amazon ECR affiche AWS Management Console uniquement les secrets de Secrets Manager dont les noms utilisent le `ecr-pullthroughcache/` préfixe.

- b. (Facultatif) Dans la section Tags (Balises), vous pouvez ajouter une ou plusieurs balises à votre secret. Pour les stratégies de balisage, consultez la [Balise secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Ne stockez pas les informations sensibles dans des balises car elles ne sont pas chiffrées.
    - c. (Facultatif) Dans Resource permissions (Permission de la ressource), pour ajouter une stratégie de ressources à votre secret, choisissez Edit permissions (Modification des autorisations). Pour plus d'informations, consultez la rubrique [Attacher une politique d'autorisation à un secret Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .
    - d. (Facultatif) Dans Répliquer le secret, pour répliquer votre secret vers un autre Région AWS, choisissez Répliquer le secret. Vous pouvez reproduire votre secret maintenant ou revenir et le répliquer ultérieurement. Pour plus d'informations, consultez la rubrique [Réplication d'un secret vers d'autres régions](#) dans le Guide de l'utilisateur AWS Secrets Manager .
    - e. Choisissez Suivant.
5. (Facultatif) Dans la page Configure rotation (Configurer la rotation), vous pouvez activer la rotation automatique. Vous pouvez également garder la rotation désactivée pour l'instant, puis l'activer plus tard. Pour plus d'informations, consultez la rubrique [Rotation des secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Choisissez Suivant.

6. Dans la page Review (Révision), passez en revue vos paramètres, puis choisissez Store (Stocker).

Secrets Manager revient à la liste des secrets. Si votre nouveau secret n'apparaît pas, choisissez le bouton d'actualisation.


## Microsoft Azure Container Registry

Pour créer un secret Secrets Manager pour vos informations d'identification Microsoft Azure Container Registry (AWS Management Console)

1. Ouvrez la console Secrets Manager à l'adresse <https://console.aws.amazon.com/secretsmanager/>.
2. Choisissez Store a new secret (Stocker un nouveau secret).
3. Sur la page Choisir un type de secret, procédez comme suit.
  - a. Pour Secret type (Type de secret), choisissez Other type of secret (Autre type de secret).
  - b. Dans les paires clé/valeur, créez deux lignes pour vos informations d'identification Microsoft Azure. Vous pouvez stocker jusqu'à 65 536 octets dans le secret.
    - i. Pour la première key/value paire, spécifiez `username` comme clé et votre nom d'utilisateur Microsoft Azure Container Registry comme valeur.
    - ii. Pour la deuxième key/value paire, spécifiez `accessToken` comme clé et votre jeton d'accès au Microsoft Azure Container Registry comme valeur. Pour plus d'informations sur la création d'un jeton d'accès à Microsoft Azure, consultez la section [Créer un jeton - portail](#) dans la documentation Microsoft Azure.
  - c. Pour la clé de chiffrement, conservez la valeur par défaut de AWS KMS key `aws/secretsmanager`, puis choisissez Suivant. L'utilisation de cette clé n'entraîne aucun coût. Pour plus d'informations, consultez la section [Chiffrement et déchiffrement de secret dans Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .

### Important

Vous devez utiliser la clé de chiffrement par défaut `aws/secretsmanager` pour chiffrer votre secret. Amazon ECR ne prend pas en charge l'utilisation d'une clé gérée par le client (CMK) pour cela.

4. Sur la page Configure secret (Configurer le secret), procédez comme suit :
    - a. Saisissez un Secret name (Nom de secret) descriptif et une Description. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode et être préfixés par `ecr-pullthroughcache/`.
-  **Important**

Amazon ECR affiche AWS Management Console uniquement les secrets de Secrets Manager dont les noms utilisent le `ecr-pullthroughcache/` préfixe.
- b. (Facultatif) Dans la section Tags (Balises), vous pouvez ajouter une ou plusieurs balises à votre secret. Pour les stratégies de balisage, consultez la [Balise secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Ne stockez pas les informations sensibles dans des balises car elles ne sont pas chiffrées.
    - c. (Facultatif) Dans Resource permissions (Permission de la ressource), pour ajouter une stratégie de ressources à votre secret, choisissez Edit permissions (Modification des autorisations). Pour plus d'informations, consultez la rubrique [Attacher une politique d'autorisation à un secret Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .
    - d. (Facultatif) Dans Répliquer le secret, pour répliquer votre secret vers un autre Région AWS, choisissez Répliquer le secret. Vous pouvez reproduire votre secret maintenant ou revenir et le répliquer ultérieurement. Pour plus d'informations, consultez la rubrique [Réplication d'un secret vers d'autres régions](#) dans le Guide de l'utilisateur AWS Secrets Manager .
    - e. Choisissez Suivant.
  5. (Facultatif) Dans la page Configure rotation (Configurer la rotation), vous pouvez activer la rotation automatique. Vous pouvez également garder la rotation désactivée pour l'instant, puis l'activer plus tard. Pour plus d'informations, consultez la rubrique [Rotation des secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Choisissez Suivant.
  6. Dans la page Review (Révision), passez en revue vos paramètres, puis choisissez Store (Stocker).

Secrets Manager revient à la liste des secrets. Si votre nouveau secret n'apparaît pas, choisissez le bouton d'actualisation.

## GitLab Container Registry

Pour créer un secret Secrets Manager pour vos informations d'identification du registre des GitLab conteneurs (AWS Management Console)

1. Ouvrez la console Secrets Manager à l'adresse <https://console.aws.amazon.com/secretsmanager/>.
2. Choisissez Store a new secret (Stocker un nouveau secret).
3. Sur la page Choisir un type de secret, procédez comme suit.
  - a. Pour Secret type (Type de secret), choisissez Other type of secret (Autre type de secret).
  - b. Dans les paires clé/valeur, créez deux lignes pour vos GitLab informations d'identification. Vous pouvez stocker jusqu'à 65 536 octets dans le secret.
    - i. Pour la première key/value paire, spécifiez `username` comme clé et votre nom d'utilisateur de GitLab Container Registry comme valeur.
    - ii. Pour la deuxième key/value paire, spécifiez `accessToken` comme clé et votre jeton d'accès au registre des GitLab conteneurs comme valeur. Pour plus d'informations sur la création d'un jeton d'accès au registre des GitLab conteneurs, voir [Jetons d'accès personnels](#), [jetons d'accès de groupe](#) ou [jetons d'accès au projet](#) dans la GitLab documentation.
  - c. Pour la clé de chiffrement, conservez la valeur par défaut de AWS KMS key `aws/secretsmanager`, puis choisissez Suivant. L'utilisation de cette clé n'entraîne aucun coût. Pour plus d'informations, consultez la section [Chiffrement et déchiffrement de secret dans Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .

### Important

Vous devez utiliser la clé de chiffrement par défaut `aws/secretsmanager` pour chiffrer votre secret. Amazon ECR ne prend pas en charge l'utilisation d'une clé gérée par le client (CMK) pour cela.

4. Sur la page Configure secret (Configurer le secret), procédez comme suit :
  - a. Saisissez un Secret name (Nom de secret) descriptif et une Description. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode et être préfixés par `ecr-pullthroughcache/`.

**⚠ Important**

Amazon ECR affiche AWS Management Console uniquement les secrets de Secrets Manager dont les noms utilisent le `ecr-pullthroughcache/` préfixe.


- b. (Facultatif) Dans la section Tags (Balises), vous pouvez ajouter une ou plusieurs balises à votre secret. Pour les stratégies de balisage, consultez la [Balise secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Ne stockez pas les informations sensibles dans des balises car elles ne sont pas chiffrées.
  - c. (Facultatif) Dans Resource permissions (Permission de la ressource), pour ajouter une stratégie de ressources à votre secret, choisissez Edit permissions (Modification des autorisations). Pour plus d'informations, consultez la rubrique [Attacher une politique d'autorisation à un secret Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .
  - d. (Facultatif) Dans Répliquer le secret, pour répliquer votre secret vers un autre Région AWS, choisissez Répliquer le secret. Vous pouvez reproduire votre secret maintenant ou revenir et le répliquer ultérieurement. Pour plus d'informations, consultez la rubrique [Réplication d'un secret vers d'autres régions](#) dans le Guide de l'utilisateur AWS Secrets Manager .
  - e. Choisissez Suivant.
5. (Facultatif) Dans la page Configure rotation (Configurer la rotation), vous pouvez activer la rotation automatique. Vous pouvez également garder la rotation désactivée pour l'instant, puis l'activer plus tard. Pour plus d'informations, consultez la rubrique [Rotation des secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Choisissez Suivant.
  6. Dans la page Review (Révision), passez en revue vos paramètres, puis choisissez Store (Stocker).

Secrets Manager revient à la liste des secrets. Si votre nouveau secret n'apparaît pas, choisissez le bouton d'actualisation.

## Chainguard Registry

Pour créer un secret Secrets Manager pour vos informations d'identification Chainguard ( )AWS Management Console

1. Ouvrez la console Secrets Manager à l'adresse <https://console.aws.amazon.com/secretsmanager/>.
2. Choisissez Store a new secret (Stocker un nouveau secret).
3. Sur la page Choisir un type de secret, procédez comme suit.
  - a. Pour Secret type (Type de secret), choisissez Other type of secret (Autre type de secret).
  - b. Dans les paires clé/valeur, créez deux lignes pour vos informations d'identification Chainguard. Vous pouvez stocker jusqu'à 65 536 octets dans le secret.
    - i. Pour la première key/value paire, spécifiez `username` comme clé et votre nom d'utilisateur Chainguard Registry comme valeur.
    - ii. Pour la deuxième key/value paire, spécifiez `accessToken` comme clé et votre jeton d'accès au registre Chainguard comme valeur. Pour plus d'informations sur la création d'un jeton d'extraction du registre Chainguard, voir [Authentification avec un jeton de traction](#) dans la documentation de Chainguard.
  - c. Pour la clé de chiffrement, conservez la valeur par défaut de AWS KMS key `aws/secretsmanager`, puis choisissez Suivant. L'utilisation de cette clé n'entraîne aucun coût. Pour plus d'informations, consultez la section [Chiffrement et déchiffrement de secret dans Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .

 Important

Vous devez utiliser la clé de chiffrement par défaut `aws/secretsmanager` pour chiffrer votre secret. Amazon ECR ne prend pas en charge l'utilisation d'une clé gérée par le client (CMK) pour cela.

4. Sur la page Configure secret (Configurer le secret), procédez comme suit :
  - a. Saisissez un Secret name (Nom de secret) descriptif et une Description. Les noms des secrets doivent contenir entre 1 et 512 caractères Unicode et être préfixés par `ecr-pullthroughcache/`.

**⚠ Important**

Amazon ECR affiche AWS Management Console uniquement les secrets de Secrets Manager dont les noms utilisent le `ecr-pullthroughcache/` préfixe.

- b. (Facultatif) Dans la section Tags (Balises), vous pouvez ajouter une ou plusieurs balises à votre secret. Pour les stratégies de balisage, consultez la [Balise secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Ne stockez pas les informations sensibles dans des balises car elles ne sont pas chiffrées.
  - c. (Facultatif) Dans Resource permissions (Permission de la ressource), pour ajouter une stratégie de ressources à votre secret, choisissez Edit permissions (Modification des autorisations). Pour plus d'informations, consultez la rubrique [Attacher une politique d'autorisation à un secret Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager .
  - d. (Facultatif) Dans Répliquer le secret, pour répliquer votre secret vers un autre Région AWS, choisissez Répliquer le secret. Vous pouvez reproduire votre secret maintenant ou revenir et le répliquer ultérieurement. Pour plus d'informations, consultez la rubrique [Réplication d'un secret vers d'autres régions](#) dans le Guide de l'utilisateur AWS Secrets Manager .
  - e. Choisissez Suivant.
5. (Facultatif) Dans la page Configure rotation (Configurer la rotation), vous pouvez activer la rotation automatique. Vous pouvez également garder la rotation désactivée pour l'instant, puis l'activer plus tard. Pour plus d'informations, consultez la rubrique [Rotation des secrets Secrets Manager](#) dans le Guide de l'utilisateur AWS Secrets Manager . Choisissez Suivant.
  6. Dans la page Review (Révision), passez en revue vos paramètres, puis choisissez Store (Stocker).

Secrets Manager revient à la liste des secrets. Si votre nouveau secret n'apparaît pas, choisissez le bouton d'actualisation.

## Personnalisation des préfixes de référentiel pour l'ECR vers le cache d'extraction ECR

Les règles de cache extractibles prennent en charge à la fois le préfixe du référentiel `ecr` et le préfixe du référentiel en amont. Le préfixe de référentiel `ecr` est le préfixe d'espace de noms de référentiel dans le registre de cache Amazon ECR associé à la règle. Tous les référentiels utilisant ce préfixe deviennent des référentiels compatibles avec le cache extractible pour le registre en amont défini dans la règle. Par exemple, le préfixe « » `prod` s'applique à tous les référentiels commençant par `prod/`. Pour appliquer un modèle à tous les référentiels de votre registre auxquels aucune règle de cache d'extraction n'est associée, utilisez-le `ROOT` comme préfixe.

### Important

Il y a toujours un `/` supposé/appliqué à la fin du préfixe. Si vous spécifiez `ecr-public` comme préfixe, Amazon ECR le traite comme `ecr-public/`.

Le préfixe du référentiel en amont correspond au nom du référentiel en amont. Par défaut, il est défini sur `ROOT`, ce qui permet de faire correspondre n'importe quel référentiel en amont. Vous pouvez définir le préfixe du référentiel en amont uniquement lorsque le préfixe du référentiel Amazon ECR n'a pas de valeur. `ROOT`

Le tableau suivant montre le mappage entre les noms de référentiels de cache et les noms de référentiels en amont en fonction de leurs configurations de préfixes dans les règles de cache d'extraction.

| Espace de noms du cache | Espace de noms en amont        | Relation de mappage (dépôt de cache → référentiel en amont)                                                                                  |
|-------------------------|--------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ecr-public</code> | <code>ROOT</code> (par défaut) | <code>ecr-public/my-app/image1</code> → <code>my-app/image1</code><br><br><code>ecr-public/my-app/image2</code> → <code>my-app/image2</code> |

| Espace de noms du cache | Espace de noms en amont | Relation de mappage (dépôt de cache → référentiel en amont) |
|-------------------------|-------------------------|-------------------------------------------------------------|
| RACINE                  | RACINE                  | my-app/image1 → my-app/image1                               |
| équipe-a                | équipe-a                | team-a/myapp/image1 → team-a/myapp/image1                   |
| mon-app                 | appli en amont          | my-app/image1 → upstream-app/image1                         |

## Résolution des problèmes liés au cache d'extraction dans Amazon ECR

Voici les erreurs les plus courantes que vous pouvez recevoir lors de l'extraction d'une image en amont à l'aide d'une règle de cache par extraction.

### Le référentiel n'existe pas

Une erreur indiquant que le référentiel n'existe pas résulte le plus souvent du fait que le référentiel n'existe pas dans votre registre privé Amazon ECR ou du fait que l'autorisation `ecr:CreateRepository` n'est pas accordée à l'IAM principal qui extrait l'image en amont. Pour résoudre cette erreur, vous devez vérifier que l'URI du référentiel dans votre commande d'extraction est correct, que les autorisations IAM requises sont accordées à l'IAM principal qui extrait l'image en amont ou que le référentiel de l'image en amont vers laquelle vous souhaitez effectuer le transfert est créé dans votre registre privé Amazon ECR avant d'effectuer l'extraction d'image en amont. Pour plus d'informations sur les autorisations IAM requises, consultez [Autorisations IAM requises pour synchroniser un registre en amont avec un registre privé Amazon ECR](#)

Voici un exemple de cette erreur.

```
Error response from daemon: repository 111122223333.dkr.ecr.us-east-1.amazonaws.com/
ecr-public/amazonlinux/amazonlinux not found: name unknown: The repository with
```

```
name 'ecr-public/amazonlinux/amazonlinux' does not exist in the registry with id '111122223333'
```

## Image demandée introuvable

Une erreur indiquant que l'image est introuvable résulte le plus souvent du fait que l'image n'existe pas dans votre registre privé Amazon ECR en amont ou du fait que l'autorisation `ecr:BatchImportUpstreamImage` n'est pas accordée à l'IAM principal qui extrait l'image en amont mais que le référentiel existe déjà dans votre registre privé Amazon ECR. Pour résoudre cette erreur, vous devez vérifier que le nom de l'image en amont et celui de la balise de l'image sont corrects et qu'ils existent, et que les autorisations IAM requises sont accordées à l'IAM principal qui extrait l'image en amont. Pour plus d'informations sur les autorisations IAM requises, consultez [Autorisations IAM requises pour synchroniser un registre en amont avec un registre privé Amazon ECR](#).

Voici un exemple de cette erreur.

```
Error response from daemon: manifest for 111122223333.dkr.ecr.us-east-1.amazonaws.com/ecr-public/amazonlinux/amazonlinux:latest not found: manifest unknown: Requested image not found
```

## 403 Interdit lors de l'extraction depuis un dépôt Docker Hub

Lorsque vous extrayez un référentiel Docker Hub étiqueté comme image officielle Docker, vous devez inclure `/library/` dans l'URI que vous utilisez. Par exemple, `aws_account_id.dkr.ecr.region.amazonaws.com/docker-hub/library/image_name:tag`. Si vous omettez `/library/` les images officielles de Docker Hub, une 403 Forbidden erreur sera renvoyée lorsque vous tenterez d'extraire l'image à l'aide d'une règle de cache d'extraction. Pour de plus amples informations, veuillez consulter [Extraction d'une image à l'aide d'une règle de cache d'extraction dans Amazon ECR](#).

Voici un exemple de cette erreur.

```
Error response from daemon: failed to resolve reference "111122223333.dkr.ecr.us-west-2.amazonaws.com/docker-hub/amazonlinux:2023": pulling from host 111122223333.dkr.ecr.us-west-2.amazonaws.com failed with status code [manifests 2023]: 403 Forbidden
```

# Réplication d'images privées sur Amazon ECR

Vous pouvez configurer votre registre privé Amazon ECR pour prendre en charge la réplication de vos référentiels. Amazon ECR prend en charge la réplication inter-régions et inter-comptes. Pour que la réplication inter-comptes se produise, le compte de destination doit configurer une politique d'autorisations de registre pour autoriser la réplication à partir du registre source. Pour de plus amples informations, veuillez consulter [Autorisations de registre privé dans Amazon ECR](#).

## Rubriques

- [Exigences relatives à la politique de réplication entre comptes](#)
- [Considérations relatives à la réplication d'images privées](#)
- [Exemples de réplication d'images privées pour Amazon ECR](#)
- [Configuration de la réplication d'images privées dans Amazon ECR](#)
- [Suppression des paramètres de réplication d'images privées dans Amazon ECR](#)

## Exigences relatives à la politique de réplication entre comptes

Pour que la réplication ECR entre comptes fonctionne correctement, vous devez comprendre quel compte a besoin de quelles politiques configurées. Cette section précise les exigences de la politique pour les comptes source et de destination.

## Présentation de la configuration des politiques

La réplication ECR entre comptes nécessite la configuration des politiques sur le compte de destination uniquement. Le compte source ne nécessite aucune politique de dépôt ou de registre particulière.

- **Compte source** : configurez les règles de réplication dans les paramètres du registre. Aucune politique supplémentaire n'est requise pour les référentiels sources.
- **Compte de destination** : configurez une politique d'autorisations de registre pour autoriser le compte source à répliquer des images.

## Exigences relatives à la politique du registre de destination

Le compte de destination doit configurer une politique d'autorisations de registre qui accorde au compte source l'autorisation d'effectuer les actions suivantes :

- `ecr:ReplicateImage`- Permet au compte source de répliquer les images dans le registre de destination
- `ecr:CreateRepository`- Permet à ECR de créer automatiquement des référentiels dans le registre de destination s'ils n'existent pas déjà

### Important

Si vous n'accordez pas l'`ecr:CreateRepository` autorisation, vous devez créer manuellement des référentiels portant le même nom dans le compte de destination pour que la réplication puisse réussir.

Exemple de politique de registre de destination :

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "AllowCrossAccountReplication",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:root"
 },
 "Action": [
 "ecr:ReplicateImage",
 "ecr:CreateRepository"
],
 "Resource": "*"
 }
]
}
```

## Exigences relatives au compte source

Le compte source doit uniquement :

- Configurez les règles de réplication dans les paramètres du registre pour spécifier le compte et les régions de destination
- Assurez-vous que le principal IAM qui configure la réplication dispose des autorisations ECR nécessaires

Aucune politique supplémentaire n'est requise pour les référentiels sources. Les référentiels sources n'ont pas besoin de politiques de dépôt accordant des autorisations de réplication.

## Idées fausses courantes

Voici quelques idées fausses courantes concernant les politiques de réplication entre comptes ECR :

- Idée fausse : le référentiel source a besoin d'une politique permettant au compte de destination de répliquer des images.

Réalité : les référentiels sources n'ont pas besoin de règles spéciales pour la réplication.

- Idée fausse : les comptes source et de destination ont besoin de politiques de registre.

Réalité : Seul le compte de destination a besoin d'une politique d'autorisation de registre.

- Idée fausse : les politiques de dépôt et les politiques de registre sont identiques.

Réalité : les politiques de référentiel contrôlent l'accès aux référentiels individuels, tandis que les politiques de registre contrôlent les opérations au niveau du registre, telles que la réplication.

## Résolution des problèmes de réplication

Si la réplication entre comptes échoue, vérifiez les points suivants :

- Vérifiez que le compte de destination dispose d'une politique d'autorisations de registre configurée
- Assurez-vous que la politique de registre inclut les deux `ecr:ReplicateImage` et les `ecr:CreateRepository` actions
- Vérifiez que l'ID du compte source est correctement spécifié dans la politique de registre de destination

- Vérifiez que les référentiels de destination existent (s'ils ne `ecr:CreateRepository` sont pas autorisés)
- Consultez CloudTrail les journaux pour détecter les échecs `CreateRepository` ou les appels `ReplicateImage` d'API

## Considérations relatives à la réplication d'images privées

Les informations suivantes doivent être prises en compte lors de l'utilisation de la réplication d'images privées.

- Seul le contenu du référentiel transféré ou restauré vers un référentiel après la configuration de la réplication est répliqué. Tout contenu préexistant dans un référentiel n'est pas répliqué. Si une image est restaurée après l'activation de la réplication, elle sera répliquée. S'il est restauré avant que la réplication ne soit activée, il ne sera pas répliqué.
- Le nom du référentiel restera le même pour toutes les régions et tous les comptes une fois la réplication effectuée. Amazon ECR ne prend pas en charge la modification du nom du référentiel pendant la réplication.
- La première fois que vous configurez votre registre privé pour la réplication, Amazon ECR crée un rôle IAM lié à un service en votre nom. Le rôle IAM lié à un service octroie au service de réplication Amazon ECR l'autorisation dont il a besoin pour créer des référentiels et répliquer des images dans votre registre. Pour de plus amples informations, veuillez consulter [Utilisation des rôles liés à un service pour Amazon ECR](#).
- Pour que la réplication inter-comptes se produise, la destination du registre privé doit octroyer au registre source l'autorisation de répliquer ses images. Pour ce faire, définissez une politique d'autorisations de registre privé. Pour de plus amples informations, veuillez consulter [Autorisations de registre privé dans Amazon ECR](#).
- Si la politique d'autorisation d'un registre privé est modifiée pour supprimer une autorisation, toutes les réplications en cours précédemment octroyées peuvent se terminer.
- Pour que la réplication entre régions ait lieu, les comptes source et de destination doivent être intégrés à la région avant toute action de réplication au sein ou vers cette région. Pour plus d'informations, consultez [Gestion des régions AWS](#) dans le Référence générale d'Amazon Web Services.
- La réplication entre régions n'est pas prise en charge entre les AWS partitions. Par exemple, un référentiel dans `us-west-2` ne peut pas être répliqué vers `cn-north-1`. Pour plus

d'informations sur AWS les partitions, consultez la section [Format ARN](#) dans le manuel de référence AWS général.

- La configuration de réplication d'un registre privé peut contenir jusqu'à 25 destinations uniques pour toutes les règles, avec un maximum de 10 règles au total. Chaque règle peut contenir jusqu'à 100 filtres. Cela permet de préciser des règles distinctes pour les référentiels contenant des images utilisées pour la production et le test, par exemple.
- La configuration de réplication prend en charge le filtrage des référentiels dans un registre privé qui sont répliqués en indiquant un préfixe de référentiel. Pour voir un exemple, consultez [Exemple : Configuration de la réplication inter-régions à l'aide d'un filtre de référentiel](#).
- Une action de réplication ne se produit qu'une seule fois par transfert d'image ou par restauration d'image. Par exemple, si vous avez configuré la réplication inter-régions à partir de us-west-2 sur us-east-1 et à partir de us-east-1 sur us-east-2, une image poussée vers us-west-2 répliquera uniquement sur us-east-1, elle ne se répliquera pas à nouveau sur us-east-2. Ce comportement s'applique à la fois à la réplication inter-régions et inter-comptes.
- La majorité des images se répliquent en moins de 30 minutes, mais dans de rares cas, la réplication peut prendre plus de temps.
- La réplication du registre n'effectue aucune action de suppression ou d'archivage. Les images et les référentiels répliqués peuvent être supprimés ou archivés lorsqu'ils ne sont plus utilisés.
- Si l'image à répliquer est archivée dans la destination, elle sera restaurée dans la destination.
- Lorsqu'une image est archivée dans une région source, elle ne sera pas archivée dans une région de destination spécifiée par la configuration de réplication.
- Les politiques de référentiel, notamment les politiques IAM et les politiques de cycle de vie, ne sont pas répliquées et n'ont aucun effet autre que sur le référentiel pour lequel elles sont définies.
- Les paramètres du référentiel ne sont pas répliqués par défaut. Vous pouvez répliquer les paramètres du référentiel à l'aide de modèles de création de référentiel. Ces paramètres incluent la mutabilité des balises, le chiffrement, les autorisations de dépôt et les politiques de cycle de vie. Pour plus d'informations sur les modèles de création de référentiels, consultez [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).
- Si l'immutabilité des étiquettes est activée sur un référentiel et qu'une image qui utilise la même étiquette qu'une image existante est répliquée, l'image sera répliquée, mais elle ne contiendra pas l'étiquette dupliquée. Cela pourrait entraîner le non-étiquetage de l'image.

- Lors de la réplication d'images, si le montage par blob a été configuré, ECR vérifie que toutes les couches du référentiel source existent déjà dans le registre de destination. Si des couches existent déjà dans le registre de destination, ECR les montera.

#### Note

Si le registre source est différent de son registre de destination, le montage par blob devra être activé pour les deux registres pour que l'ECR puisse monter les couches répliquées.

## Exemples de réplication d'images privées pour Amazon ECR

Les exemples suivants montrent des cas d'utilisation courants de la réplication d'images privées. Si vous configurez la réplication à l'aide du AWS CLI, vous pouvez utiliser les exemples JSON comme point de départ lorsque vous créez votre fichier JSON. Si vous configurez la réplication à l'aide du AWS Management Console, vous verrez un JSON similaire lorsque vous examinerez votre règle de réplication sur la page Révision et envoi.

### Exemple : configuration de la réplication inter-régions sur une même région de destination

L'exemple suivant illustre la configuration de la réplication inter-régions dans un registre unique. Cet exemple suppose que votre ID de compte est `111122223333` et que vous indiquez cette configuration de réplication dans une région autre que `us-west-2`.

```
{
 "rules": [
 {
 "destinations": [
 {
 "region": "us-west-2",
 "registryId": "111122223333"
 }
]
 }
]
}
```

## Exemple : Configuration de la réplication inter-régions à l'aide d'un filtre de référentiel

L'exemple suivant illustre la configuration de la réplication inter-régions pour les référentiels qui correspondent à une valeur de nom de préfixe. Cet exemple suppose que votre ID de compte est 111122223333, que vous indiquez cette configuration de réplication dans une région autre que us-west-1 et dispose de référentiels avec un préfixe de prod.

```
{
 "rules": [{
 "destinations": [{
 "region": "us-west-1",
 "registryId": "111122223333"
 }],
 "repositoryFilters": [{
 "filter": "prod",
 "filterType": "PREFIX_MATCH"
 }]
 }]
}
```

## Exemple : Configuration de la réplication inter-régions vers plusieurs régions de destination

L'exemple suivant illustre la configuration de la réplication inter-régions dans un registre unique. Cet exemple suppose que votre identifiant de compte est 111122223333 et que vous spécifiez cette configuration de réplication dans une région autre que us-west-1 ou us-west-2.

```
{
 "rules": [
 {
 "destinations": [
 {
 "region": "us-west-1",
 "registryId": "111122223333"
 },
 {
 "region": "us-west-2",
 "registryId": "111122223333"
 }
]
 }
]
}
```

```

]
 }
]
}

```

## Exemple : Configuration de la réplication inter-comptes

L'exemple suivant illustre la configuration de la réplication inter-comptes pour votre registre. Cet exemple configure la réplication vers le compte 444455556666 et vers la région us-west-2.

### Important

Pour que la réplication inter-comptes se produise, le compte de destination doit configurer une politique d'autorisations de registre pour autoriser la réplication. Pour de plus amples informations, veuillez consulter [Autorisations de registre privé dans Amazon ECR](#).

```

{
 "rules": [
 {
 "destinations": [
 {
 "region": "us-west-2",
 "registryId": "444455556666"
 }
]
 }
]
}

```

## Exemple : Spécification de plusieurs règles dans une configuration

L'image suivante présente un exemple de configuration de plusieurs règles de réplication pour votre registre. Cet exemple configure la réplication pour le compte **111122223333** avec une seule règle qui réplique les référentiels avec un préfixe de prod vers la région us-west-2 et les référentiels avec un préfixe de test vers la région us-east-2. Une configuration de réplication peut contenir jusqu'à 10 règles, chaque règle indiquant jusqu'à 25 destinations.

```

{

```

```
"rules": [{
 "destinations": [{
 "region": "us-west-2",
 "registryId": "111122223333"
 }],
 "repositoryFilters": [{
 "filter": "prod",
 "filterType": "PREFIX_MATCH"
 }]
},
{
 "destinations": [{
 "region": "us-east-2",
 "registryId": "111122223333"
 }],
 "repositoryFilters": [{
 "filter": "test",
 "filterType": "PREFIX_MATCH"
 }]
}
]
```

## Exemple : suppression de tous les paramètres de réplication

Voici un exemple de suppression de tous les paramètres de réplication de votre registre. Pour supprimer les paramètres de réplication, vous devez configurer un tableau de règles vide.

```
{
 "rules": []
}
```

### Important

La suppression des paramètres de réplication ne supprime pas les référentiels ou les images précédemment répliqués. Vous devez supprimer manuellement le contenu répliqué s'il n'est plus nécessaire.

# Configuration de la réplication d'images privées dans Amazon ECR

Configurez la réplication par région pour votre registre privé. Vous pouvez configurer la réplication entre régions ou entre comptes.

Pour obtenir des exemples sur la manière dont la réplication est couramment utilisée, consultez [Exemples de réplication d'images privées pour Amazon ECR](#).

## Configurer les paramètres de réplication du registre (AWS Management Console)

1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région pour laquelle vous souhaitez configurer les paramètres de réplication du registre.
3. Dans le panneau de navigation, choisissez Registre de schémas.
4. Sur la page du registre privé, choisissez Paramètres, puis sélectionnez Modifier sous Configuration de la réplication.
5. Dans la page Réplication, choisissez Ajouter une règle de réplication.
6. Dans la page Types de destination, choisissez si vous souhaitez activer la réplication inter-régions, la réplication inter-comptes ou les deux, puis choisissez Suivant.
7. Si la réplication inter-régions est activée, alors pour Configuration des régions de destination, choisissez une ou plusieurs régions de destination, puis choisissez Suivant.
8. Si la réplication inter-comptes est activée, alors pour Réplication inter-comptes, choisissez le paramètre de réplication inter-comptes pour le registre. Pour Compte de destination, saisissez l'ID de compte pour le compte de destination et une ou plusieurs régions de destination à répliquer. Choisissez Compte de destination + pour configurer des comptes supplémentaires en tant que destinations de réplication.

### Important

Pour que la réplication inter-comptes se produise, le compte de destination doit configurer une politique d'autorisations de registre pour autoriser la réplication. Pour de plus amples informations, veuillez consulter [Autorisations de registre privé dans Amazon ECR](#).

9. (Facultatif) Dans l'onglet Ajouter des filtres, indiquez un ou plusieurs filtres pour la règle de réplication, puis choisissez Ajouter. Répétez cette étape pour chaque filtre que vous souhaitez

associer à l'action de réplication. Un filtre doit être spécifié en tant que préfixe de nom de référentiel. Si aucun filtre n'est ajouté, le contenu de tous les référentiels est répliqué. Choisissez Suivant lorsque tous les filtres auront été ajoutés.

10. Dans la page Examen et soumission, examinez la configuration de la règle de réplication, puis choisissez Soumettre une règle.

## Configurer les paramètres de réplication du registre (AWS CLI)

1. Créez un fichier JSON contenant les règles de réplication à définir pour votre registre. Une configuration de réplication peut contenir jusqu'à 10 règles, avec jusqu'à 25 destinations uniques pour toutes les règles et 100 filtres par règle. Pour configurer la réplication inter-régions dans votre propre compte, indiquez votre propre ID de compte. Pour obtenir plus d'exemples, consultez [Exemples de réplication d'images privées pour Amazon ECR](#).

```
{
 "rules": [{
 "destinations": [{
 "region": "destination_region",
 "registryId": "destination_accountId"
 }],
 "repositoryFilters": [{
 "filter": "repository_prefix_name",
 "filterType": "PREFIX_MATCH"
 }]
 }]
}
```

2. Créez une configuration de réplication pour votre registre.

```
aws ecr put-replication-configuration \
 --replication-configuration file://replication-settings.json \
 --region us-west-2
```

3. Confirmez les paramètres de votre registre.

```
aws ecr describe-registry \
 --region us-west-2
```

# Suppression des paramètres de réplication d'images privées dans Amazon ECR

Pour supprimer ou désactiver les paramètres de réplication de votre registre privé, vous devez configurer une configuration de réplication vide. Il n'existe aucune commande de suppression dédiée dans le AWS CLI.

## Pour supprimer les paramètres de réplication du registre (AWS Management Console)

1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région dont vous souhaitez supprimer les paramètres de réplication du registre.
3. Dans le panneau de navigation, choisissez Registre de schémas.
4. Sur la page du registre privé, choisissez Paramètres, puis sélectionnez Modifier sous Configuration de la réplication.
5. Supprimez toutes les règles de réplication existantes en choisissant l'option de suppression pour chaque règle.
6. Choisissez Enregistrer pour appliquer la configuration de réplication vide.

## Pour supprimer les paramètres de réplication du registre (AWS CLI)

1. Créez un fichier JSON avec un tableau de règles vide pour supprimer tous les paramètres de réplication.

```
{
 "rules": []
}
```

2. Appliquez la configuration de réplication vide à votre registre.

```
aws ecr put-replication-configuration \
 --replication-configuration file://empty-replication-settings.json \
 --region us-west-2
```

3. Vérifiez que les paramètres de réplication ont été supprimés.

```
aws ecr describe-registry \
 --region us-west-2
```

```
--region us-west-2
```

La sortie doit afficher un champ vide `replicationConfiguration` sans règles.

 Important

La suppression des paramètres de réplication ne supprime pas les référentiels ou les images précédemment répliqués. Vous devez supprimer manuellement le contenu répliqué s'il n'est plus nécessaire.

# Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication

Utilisez les modèles de création de référentiels Amazon ECR pour définir les paramètres des référentiels créés par Amazon ECR en votre nom. Les paramètres d'un modèle de création de référentiel ne sont appliqués que lors de la création du référentiel et n'ont aucun effet sur les référentiels existants ou les référentiels créés à l'aide d'une autre méthode. Actuellement, les modèles de création de référentiel peuvent être utilisés pour appliquer des paramètres lors de la création de référentiels pour les fonctionnalités suivantes :

- Extraire le cache
- Créez en mode push
- Réplication

## Fonctionnement des modèles de création de référentiels

Parfois, Amazon ECR doit créer un nouveau référentiel privé en votre nom. Par exemple :

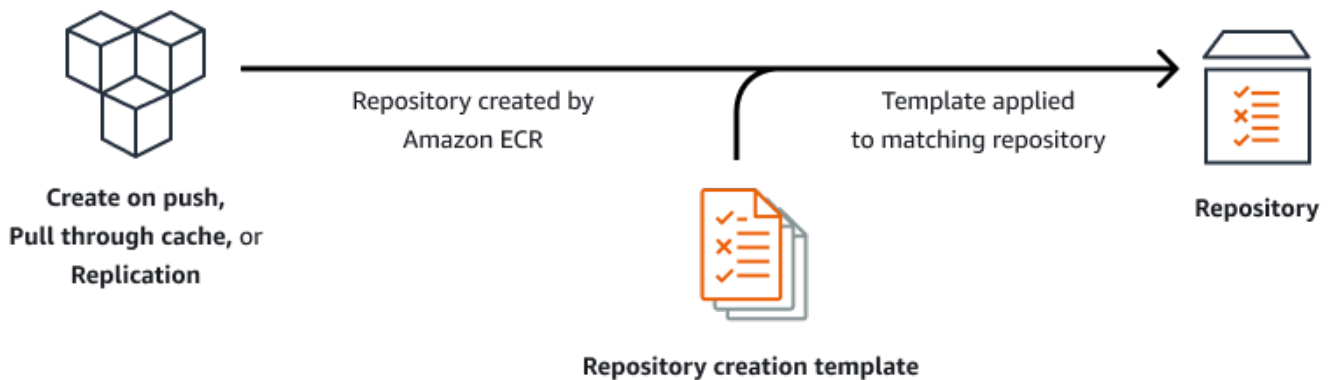
- La première fois que vous utilisez une règle de cache d'extraction pour récupérer le contenu d'un référentiel en amont et le stocker dans votre registre privé Amazon ECR.
- Lorsque vous transférez une image vers un dépôt qui n'existe pas encore.
- Lorsque vous souhaitez qu'Amazon ECR réplique un référentiel vers une autre région ou un autre compte.

Lorsqu'aucun modèle de création de référentiel ne correspond à votre règle de cache d'extraction ou à votre référentiel répliqué, Amazon ECR utilise les paramètres par défaut pour le nouveau référentiel. Ces paramètres par défaut incluent la désactivation de l'immuabilité des balises, l'utilisation du chiffrement AES-256 et l'absence d'application de politiques de référentiel ou de cycle de vie.

Lorsqu'aucun modèle de création de référentiel ne correspond au référentiel cible pour un transfert d'image, Amazon ECR ne crée pas de référentiel avec les paramètres par défaut.

L'utilisation d'un modèle de création de référentiel vous permet de définir les paramètres qu'Amazon ECR applique aux nouveaux référentiels créés via le cache d'extraction, la création en mode push et les actions de réplication. Vous pouvez définir l'immutabilité des balises, la configuration du chiffrement, les autorisations des référentiels, la politique de cycle de vie et les balises de ressource pour les nouveaux référentiels.

Le schéma suivant illustre le flux de travail qu'Amazon ECR utilise lorsqu'un modèle de création de référentiel est utilisé.



Vous trouverez ci-dessous une description détaillée de chaque paramètre d'un modèle de création de référentiel.

## Préfixe

Le préfixe est le préfixe de l'espace de noms du référentiel à associer au modèle. Les paramètres définis dans ce modèle seront appliqués à tous les référentiels créés à l'aide de ce préfixe. Par exemple, le préfixe de `prod` s'appliquerait à tous les référentiels commençant par `prod/`. De la même façon, le préfixe de `prod/team` s'appliquerait à tous les référentiels commençant par `prod/team/`. Dans un registre contenant deux modèles, si l'un des modèles porte le préfixe « `prod` » et l'autre le préfixe « `/prod/team` », the template with the prefix "prod/team" will be applied to all repositories whose names start with "prod/team".

Pour appliquer un modèle à tous les référentiels de votre registre auxquels aucun modèle de création n'est associé, vous pouvez utiliser `ROOT` comme préfixe.

### ⚠ Important

Il y a toujours un `/` supposé/appliqué à la fin du préfixe. Si vous spécifiez `ecr-public` comme préfixe, Amazon ECR le traite comme `ecr-public/`. Lorsque vous utilisez une règle de mise en cache par extraction, le préfixe de référentiel que vous spécifiez lors de

la création de la règle est également celui que vous devez spécifier comme préfixe de modèle de création de référentiel.

## Description

Cette description du modèle est facultative et est utilisée pour décrire l'objectif du modèle de création de référentiel.

## Appliqué pour

Le paramètre appliqué détermine quels référentiels créés par Amazon ECR seront créés avec ce modèle. Les valeurs valides sont `PULL_THROUGH_CACHE`, `CREATE_ON_PUSH` et `REPLICATION`. Par exemple, la première fois que vous utilisez une règle de mise en cache par extraction pour récupérer le contenu d'un référentiel en amont et le stocker dans votre registre privé Amazon ECR. Lorsqu'aucun modèle de création de référentiel ne correspond à votre règle de mise en cache par extraction, Amazon ECR utilise les paramètres par défaut pour le nouveau référentiel.

## Rôle de création de référentiel

Le rôle de création de référentiel est un ARN de rôle IAM qui sera assumé par Amazon ECR lors de la création et de la configuration de référentiels via des modèles de création de référentiels. Ce rôle doit être fourni lors de l'utilisation des balises de référentiel and/or KMS dans le modèle, sinon la création du référentiel échouera.

## Mutabilité d'une étiquette d'image

Le paramètre de mutabilité des balises à utiliser pour les référentiels créés à l'aide du modèle. Si ce paramètre est omis, le paramètre par défaut `MUTABLE` sera utilisé, ce qui permettra le remplacement des balises d'image. Il est recommandé d'utiliser ce paramètre pour les modèles utilisés pour les référentiels créés par des actions de mise en cache par extraction. Cela permet à Amazon ECR de mettre à jour les images mises en cache lorsque les balises sont identiques.

Si on spécifie `IMMUABLE`, toutes les balises d'image du référentiel seront immuables, ce qui les empêchera d'être remplacées.

## Configuration du chiffrement

### Important

Le chiffrement double couche côté serveur avec AWS KMS (DSSE-KMS) n'est disponible que dans les régions. AWS GovCloud (US)

La Configuration de chiffrement à utiliser pour les référentiels créés à l'aide du modèle.

Si vous utilisez le type de chiffrement KMS, le contenu du référentiel sera chiffré à l'aide du chiffrement côté serveur avec une clé AWS Key Management Service stockée dans AWS KMS. Lorsque vous chiffrez vos données, vous pouvez soit utiliser la AWS KMS clé AWS gérée par défaut pour Amazon ECR, soit spécifier votre propre AWS KMS clé, que vous avez déjà créée. AWS KMS Vous pouvez également choisir d'utiliser le chiffrement monocouche ou double couche avec. AWS KMS Pour plus d'informations, consultez [Chiffrement au repos](#). Si vous utilisez le type de chiffrement KMS et que vous l'utilisez pour la réplication entre régions, vous aurez peut-être besoin d'autorisations supplémentaires. Pour plus d'informations, consultez la section [Création d'une politique de clé KMS pour la réplication](#).

Si vous utilisez le type de chiffrement AES256, Amazon ECR utilise le chiffrement côté serveur avec des clés de chiffrement gérées par Amazon S3, ce qui chiffre les images dans le référentiel à l'aide d'un algorithme de chiffrement AES-256. Pour plus d'informations, consultez la section [Protection des données à l'aide du chiffrement côté serveur avec les clés de chiffrement gérés par Amazon S3 \(SSE-S3\)](#) dans le Guide de l'utilisateur Amazon Simple Storage Service.

## Autorisations du référentiel

La Politique de référentiel à appliquer aux référentiels créés à l'aide du modèle. Une politique de référentiel utilise les autorisations basées sur les ressources pour contrôler l'accès à un référentiel. Les autorisations basées sur les ressources vous permettent de spécifier les utilisateurs ou rôles IAM qui ont accès à un référentiel et quelles sont les actions qui peuvent être exécutées. Par défaut, seul le AWS compte qui a créé le référentiel a accès à un référentiel. Vous pouvez appliquer un document de politique pour accorder ou refuser des autorisations supplémentaires à votre référentiel. Pour de plus amples informations, veuillez consulter [Politiques relatives aux référentiels privés dans Amazon ECR](#).

## Politique de cycle de vie de référentiel

La politique de cycle de vie à utiliser pour les référentiels créés à l'aide du modèle. Une politique de cycle de vie vous permette de contrôler la gestion du cycle de vie des images d'un référentiel privé. Une politique de cycle de vie est un ensemble d'une ou plusieurs règles, où chaque règle définit une action pour Amazon ECR. Cela fournit un moyen d'automatiser le nettoyage des images de conteneur en faisant expirer des images selon l'ancienneté ou le décompte. Pour de plus amples informations, veuillez consulter [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).

## Balises de ressources

Les balises de ressource sont des métadonnées à appliquer au référentiel afin de mieux les classer et les organiser. Chaque balise est constituée d'une clé et d'une valeur facultative que vous définissez. Cette autorisation doit être appliquée à la politique de registre de destination si vous utilisez des modèles de création de référentiels avec réplication entre régions.

## Création d'un modèle de création de référentiel dans Amazon ECR

Vous pouvez créer un modèle de création de référentiel pour définir les paramètres à utiliser pour les référentiels créés par Amazon ECR en votre nom lors des actions d'extraction du cache, de création sur push ou de réplication. Une fois le modèle de création de référentiel créé, les paramètres seront appliqués à tous les nouveaux référentiels créés. Cela n'a aucun effet sur les référentiels créés précédemment.

Lorsque vous configurez un référentiel avec des modèles, vous avez la possibilité de spécifier des clés KMS et des balises de ressources. Si vous avez l'intention d'utiliser des clés KMS, des balises de ressources ou une combinaison des deux dans un ou plusieurs modèles, vous devez :

- [Création d'une politique personnalisée pour les modèles de création de référentiels.](#)
- [Création d'un rôle IAM pour les modèles de création de référentiels.](#)

Une fois configuré, vous pouvez associer le rôle personnalisé à des modèles spécifiques de votre registre.

## Autorisations IAM pour créer des modèles de création de référentiels

Les autorisations suivantes sont nécessaires pour qu'un principal IAM puisse gérer les modèles de création de référentiels. Cette autorisation doit être accordée à l'aide d'une politique IAM basée sur l'identité.

- `ecr:CreateRepositoryCreationTemplate` : accorde l'autorisation de créer un modèle de création de référentiel.
- `ecr:UpdateRepositoryCreationTemplate`— Autorise la mise à jour d'un modèle de création de dépôt.
- `ecr:DescribeRepositoryCreationTemplates`— Accorde l'autorisation de répertorier les modèles de création de référentiels dans un registre.

- `ecr:DeleteRepositoryCreationTemplate` : accorde l'autorisation de supprimer un modèle de création de référentiel.
- `ecr:CreateRepository`— Accorde l'autorisation de créer un référentiel Amazon ECR.
- `ecr:PutLifecyclePolicy` : accorde l'autorisation de créer une politique de cycle de vie et de l'appliquer à un référentiel. Cette autorisation n'est requise que si le modèle de création de référentiel inclut une politique de cycle de vie.
- `ecr:SetRepositoryPolicy` : accorde l'autorisation de créer une politique d'autorisation pour un référentiel. Cette autorisation n'est requise que si le modèle de création de référentiel inclut une politique de référentiel.
- `iam:PassRole`— Accorde l'autorisation d'autoriser une entité à transmettre un rôle à un service ou à une application. Cette autorisation est nécessaire pour les services et applications qui doivent assumer un rôle pour effectuer des actions en votre nom.

## Création d'une politique personnalisée pour les modèles de création de référentiels

Vous pouvez utiliser le AWS Management Console pour définir une politique qui sera ensuite associée à un rôle IAM. Ce rôle IAM peut ensuite être utilisé comme rôle de création de référentiel lors de la configuration d'un modèle de création de référentiel.

### AWS Management Console

Utiliser l'éditeur de stratégie JSON afin de créer une politique personnalisée pour les modèles de création de référentiels.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à <https://console.aws.amazon.com/iam/> l'adresse.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Choisissez Create Policy (Créer une politique).
4. Dans la section Éditeur de politiques, choisissez l'option JSON.
5. Entrez la politique suivante dans le champ JSON.

### JSON

```
{
```

```
"Version": "2012-10-17",
"Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "ecr:CreateRepository",
 "ecr:ReplicateImage",
 "ecr:TagResource"
],
 "Resource": "*"
 },
 {
 "Effect": "Allow",
 "Action": [
 "kms:CreateGrant",
 "kms:RetireGrant",
 "kms:DescribeKey"
],
 "Resource": "*"
 }
]
```

6. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés lors de [la validation des politiques](#), puis choisissez Next.
7. Lorsque vous avez fini d'ajouter des autorisations à la politique, choisissez Suivant.
8. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
9. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.
10. Créez un rôle pour attribuer cette politique au modèle de création, voir [Création d'un rôle IAM pour les modèles de création de référentiels](#).

## Création d'un rôle IAM pour les modèles de création de référentiels

Vous pouvez utiliser le AWS Management Console pour créer un rôle qui peut être utilisé par Amazon ECR lorsque vous spécifiez le rôle de création de référentiel dans un modèle de création de référentiel qui utilise des balises de référentiel ou des KMS dans un modèle.

## AWS Management Console

Pour créer un rôle.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à <https://console.aws.amazon.com/iam/> l'adresse.
2. Dans le panneau de navigation de la console, choisissez Rôles, puis Créer un rôle.
3. Choisissez le type de rôle de politique de confiance personnalisée.
4. Dans la section Politique de confiance personnalisée, collez la politique de confiance personnalisée répertoriée ci-dessous :

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Principal": {
 "Service": "ecr.amazonaws.com"
 },
 "Action": "sts:AssumeRole"
 }
]
}
```

5. Choisissez Suivant.
6. Sur la page Ajouter des autorisations, cochez la case à côté de la politique personnalisée que vous avez créée précédemment dans la liste des politiques d'autorisations et choisissez Suivant.
7. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il est sensible à la casse. Lorsque le nom d'un rôle apparaît aux clients dans la console, par exemple lors du processus de connexion, il n'est pas sensible à la casse. Différentes entités peuvent référencer le rôle et il n'est donc pas possible de modifier son nom après sa création.
8. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
9. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

## Création d'un modèle de création de référentiel

Une fois que vous avez rempli les conditions requises pour vos modèles, vous pouvez procéder à la création des modèles de création de référentiels.

### AWS Management Console


Pour créer un modèle de création de référentiel (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/>l'adresse.
2. Dans la barre de navigation, choisissez la région dans laquelle créer le modèle de création de référentiel.
3. Dans le volet de navigation, choisissez Registre privé, Modèles de création de référentiels.
4. Sur la page Modèles de création de référentiels, choisissez Créer un modèle.
5. Sur la page Étape 1 : définir le modèle, pour Détails du modèle, choisissez Un préfixe spécifique pour appliquer le modèle à un préfixe d'espace de noms de référentiel spécifique ou choisissez N'importe quel préfixe dans votre registre ECR pour appliquer le modèle à tous les référentiels qui ne correspondent à aucun autre modèle dans la région.
  - a. Si vous choisissez Un préfixe spécifique pour Préfixe, spécifiez le préfixe de l'espace de noms du référentiel auquel appliquer le modèle. Il y a toujours un / supposé/appliqué à la fin du préfixe. Par exemple, le préfixe de prod s'appliquerait à tous les référentiels commençant par prod/. De la même façon, le préfixe de prod/team s'appliquerait à tous les référentiels commençant par prod/team/.
  - b. Si vous choisissez N'importe quel préfixe dans votre registre ECR, le Préfixe sera défini sur ROOT.
6. Pour Applied for, spécifiez à quels flux de travail Amazon ECR ce modèle s'appliquera. Les options sont PULL\_THROUGH\_CACHE, CREATE\_ON\_PUSH et REPLICATION.
7. Pour Description du modèle, spécifiez une description facultative pour le modèle, puis choisissez Suivant.
8. Sur la page Étape 2 : ajouter une configuration de création de référentiel, spécifiez la configuration des paramètres de référentiel à appliquer aux référentiels créés à l'aide du modèle.
  - a. Pour Mutabilité des balises d'image, choisissez le paramètre mutabilité des balises à utiliser. Pour de plus amples informations, veuillez consulter [Empêcher le remplacement des balises d'image dans Amazon ECR](#).

- **Mutable** : choisissez cette option si vous souhaitez que les balises d'image soient remplacées. Recommandé pour les référentiels utilisant des actions de cache d'extraction afin de garantir qu'Amazon ECR puisse mettre à jour les images mises en cache. En outre, pour désactiver les mises à jour de balises pour quelques balises modifiables, entrez le nom des balises ou utilisez des caractères génériques (\*) pour faire correspondre plusieurs balises similaires dans la zone de texte d'exclusion des balises modifiables.
  - **Immuable** : choisissez cette option si vous souhaitez empêcher le remplacement des balises d'image. Elle s'applique à toutes les balises et exclusions du référentiel lors du transfert d'une image avec une balise existante. Amazon ECR renvoie un `ImageTagAlreadyExistsException` si vous tentez de publier une image avec une balise existante. En outre, pour activer les mises à jour des balises pour quelques balises immuables, entrez les noms des balises ou utilisez des caractères génériques (\*) pour faire correspondre plusieurs balises similaires dans la zone de texte Exclusion de balises immuable.
- b. Pour la configuration du chiffrement, choisissez le paramètre de chiffrement à utiliser. Pour de plus amples informations, veuillez consulter [Chiffrement au repos](#).

Lorsque AES-256 est sélectionné, Amazon ECR utilise le chiffrement côté serveur avec des clés de chiffrement gérées par Amazon Simple Storage Service, ce qui chiffre vos données au repos à l'aide d'un algorithme de chiffrement AES-256 standard du secteur. Ceci est offert sans frais supplémentaires.

Lorsqu'AWS KMS est sélectionné, Amazon ECR utilise le chiffrement côté serveur avec les clés stockées dans AWS Key Management Service (AWS KMS). Lorsque vous chiffrez vos données, vous pouvez soit utiliser la clé AWS gérée par défaut, qui est gérée par Amazon ECR, soit spécifier votre propre AWS KMS clé, appelée clé gérée par le client. AWS KMS

 Note

Les paramètres de chiffrement pour un référentiel ne peuvent pas être modifiés une fois celui-ci créé.

- c. Pour les autorisations de référentiel, spécifiez la politique d'autorisation de référentiel à appliquer aux référentiels créés à l'aide de ce modèle. Vous pouvez éventuellement utiliser le menu déroulant pour sélectionner l'un des exemples JSON pour les cas

d'utilisation les plus fréquents. Pour de plus amples informations, veuillez consulter [Politiques relatives aux référentiels privés dans Amazon ECR](#).

- d. Pour la politique de cycle de vie de référentiel, spécifiez la politique de cycle de vie de référentiel à appliquer aux référentiels créés à l'aide de ce modèle. Vous pouvez éventuellement utiliser le menu déroulant pour sélectionner l'un des exemples JSON pour les cas d'utilisation les plus fréquents. Pour de plus amples informations, veuillez consulter [Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR](#).
  - e. Pour les AWS balises de référentiel, spécifiez les métadonnées, sous forme de paires clé-valeur, à associer aux référentiels créés à l'aide de ce modèle, puis choisissez Next. Pour de plus amples informations, veuillez consulter [Marquage d'un référentiel privé dans Amazon ECR](#).
  - f. Pour le rôle de création de référentiel, sélectionnez un rôle IAM personnalisé dans le menu déroulant à utiliser pour les modèles de création de référentiel lors de l'utilisation de balises de référentiel ou de KMS dans le modèle (voir [Création d'un rôle IAM pour les modèles de création de référentiels](#) pour plus de détails). Choisissez ensuite Next.
9. Sur la page Étape 3 : vérifier et créer, passez en revue les paramètres que vous avez spécifiés pour le modèle de création de référentiel. Choisissez l'option Modifier pour effectuer des changements. Une fois que vous avez terminé, choisissez Créer.

## AWS CLI

La [create-repository-creation-template](#) AWS CLI commande est utilisée pour créer un modèle de création de référentiel pour votre registre privé.

Pour créer un modèle de création de référentiel (AWS CLI)

1. Utilisez le AWS CLI pour générer un squelette pour la [create-repository-creation-template](#) commande.

```
aws ecr create-repository-creation-template \
 --generate-cli-skeleton
```

La sortie de la commande affiche la syntaxe complète du modèle de création de référentiel.

```
{
```

```

"appliedFor":[""], // string array, but valid are PULL_THROUGH_CACHE,
CREATE_ON_PUSH, and REPLICATION
"prefix": "string",
 "description": "string",
 "imageTagMutability":
 "MUTABLE"|"IMMUTABLE"|"IMMUTABLE_WITH_EXCLUSION"|"MUTABLE_WITH_EXCLUSION",
 "imageTagMutabilityExclusionFilters": [
 "filterType": "WILDCARD",
 "filter": "string"
],
 "repositoryPolicy": "string",
 "lifecyclePolicy": "string"
"encryptionConfiguration": {
"encryptionType": "AES256"|"KMS",
 "kmsKey": "string"
},
 "resourceTags": [
 {
"Key": "string",
 "Value": "string"
 }
],
 "customRoleArn": "string", // must be a valid IAM Role ARN
}

```

2. Créez un fichier nommé `repository-creation-template.json` avec le résultat de l'étape précédente. Ce modèle définit une clé de chiffrement KMS pour tout dépôt créé dans le cadre `prod/*` d'une politique de dépôt qui permet de transférer et d'extraire des images vers de futurs référentiels, définit une politique de cycle de vie qui fera expirer les images datant de plus de deux semaines et définit un rôle personnalisé qui permettra à ECR d'accéder à la clé KMS et d'attribuer la balise de ressource aux `examplekey` futurs référentiels.

```

{
"prefix": "prod",
 "description": "For repositories cached from my PTC rule and in my
replication configuration that start with 'prod/'",
 "appliedFor": ["PULL_THROUGH_CACHE", "CREATE_ON_PUSH", "REPLICATION"],
 "encryptionConfiguration": {
"encryptionType": "KMS",
 "kmsKey": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-
cdef-example11111"
}
}

```

```

 },
 "resourceTags": [
 {
 "Key": "examplekey",
 "Value": "examplevalue"
 }
],
 "imageTagMutability": "IMMUTABLE_WITH_EXCLUSION",
 "imageTagMutabilityExclusionFilters": [
 {
 "filterType": "WILDCARD",
 "filter": "latest"
 },
 {
 "filterType": "WILDCARD",
 "filter": "beta*"
 }
]
 "repositoryPolicy": "{ \"Version\": \"2012-10-17\", \"Statement\":
 [{ \"Sid\": \"AllowPushPullIAMRole\", \"Effect\": \"Allow\", \"Principal\": { \"AWS\":
 \"arn:aws:iam::111122223333:user/IAMUsername\" }, \"Action\": [\"ecr:BatchGetImage
 \", \"ecr:BatchCheckLayerAvailability\", \"ecr:CompleteLayerUpload\",
 \"ecr:GetDownloadUrlForLayer\", \"ecr:InitiateLayerUpload\", \"ecr:PutImage\",
 \"ecr:UploadLayerPart\"]] } }",
 "lifecyclePolicy": "{ \"rules\": [{ \"rulePriority\": 1, \"description\": \"Expire
 images older than 14 days\", \"selection\": { \"tagStatus\": \"any\", \"countType
 \": \"sinceImagePushed\", \"countUnit\": \"days\", \"countNumber\": 14 }, \"action\":
 { \"type\": \"expire\" } }] }",
 "customRoleArn": "arn:aws:iam::111122223333:role/myRole"
 }

```

- Utilisez la commande suivante pour créer un modèle de création de référentiel. Assurez-vous de spécifier le nom du fichier de configuration créé à l'étape précédente à la place de celui `repository-creation-template.json` de l'exemple suivant.

```

aws ecr create-repository-creation-template \
 --cli-input-json file://repository-creation-template.json

```



```
--image-tag-mutability="IMMUTABLE_WITH_EXCLUSION" \
--image-tag-mutability-exclusion-filters filterType=WILDCARD, filter=latest
```

La sortie de la commande affiche les détails du modèle de création de référentiel mis à jour.

## Supprimer un modèle de création de référentiel dans Amazon ECR

Vous pouvez supprimer un modèle de création de référentiel une fois que vous avez fini de l'utiliser. Une fois qu'un modèle de création de référentiel est supprimé, tous les référentiels nouvellement créés sous le préfixe associé lors d'une action de récupération du cache ou de réplication hériteront des paramètres par défaut, sauf si un autre modèle correspondant est trouvé, voir. [Fonctionnement des modèles de création de référentiels](#)

### Important

Cela n'a aucun effet sur les référentiels créés précédemment.

### AWS Management Console

Pour supprimer un modèle de création de référentiel (AWS Management Console)

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région dans laquelle se trouve le modèle de création de référentiel à supprimer.
3. Dans le volet de navigation, choisissez Registre privé, Modèles de création de référentiels.
4. Sur la page Modèles de création de référentiels, sélectionnez le modèle de création de référentiel à supprimer.
5. Dans le menu déroulant Actions, choisissez Supprimer.

### AWS CLI

Pour supprimer un modèle de création de référentiel (AWS CLI)

- Utilisez la commande [delete-repository-creation-template.html](#) pour supprimer un modèle de création de référentiel existant. Vous devez spécifier la prefix valeur du modèle. L'exemple suivant supprime un modèle de création de référentiel avec le prod préfixe.

```
aws ecr delete-repository-creation-template \
 --prefix prod
```

La sortie de la commande affiche les détails du modèle de création de référentiel supprimé.

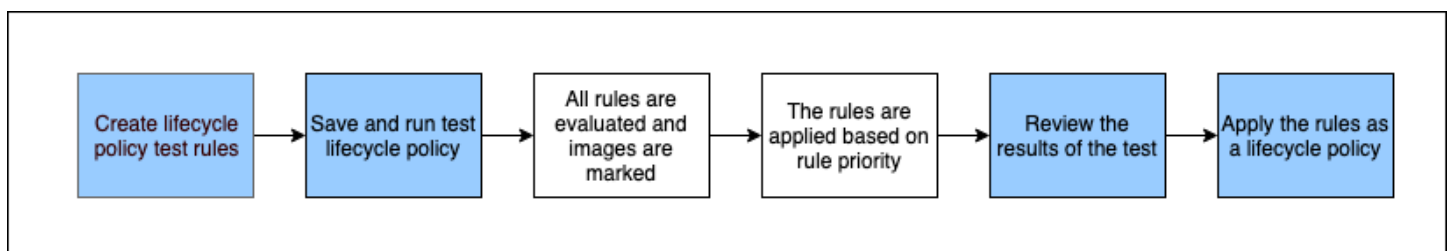
# Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR

Les politiques de cycle de vie Amazon ECR vous permettent de contrôler la gestion du cycle de vie des images d'un référentiel privé. Une politique de cycle de vie contient une ou plusieurs règles, et chaque règle définit une action pour Amazon ECR. Selon les critères d'expiration de la politique de cycle de vie, les images peuvent être archivées ou expirées selon les critères spécifiés dans la politique de cycle de vie dans les 24 heures. Lorsqu'Amazon ECR exécute une action basée sur une politique de cycle de vie, cette action est capturée en tant qu'événement dans AWS CloudTrail. Pour de plus amples informations, veuillez consulter [Journalisation des actions Amazon ECR avec AWS CloudTrail](#).

## Fonctionnement des politiques de cycle de vie

Une politique de cycle de vie se compose d'une ou de plusieurs règles qui déterminent quelles sont images d'un référentiel qui doivent expirer. Lorsque vous envisagez d'utiliser des politiques de cycle de vie, il est important d'afficher l'aperçu de la politique de cycle de vie pour confirmer quelles sont les images que la politique de cycle de vie doit faire expirer avant de l'appliquer à un référentiel. Une fois qu'une politique de cycle de vie est appliquée à un référentiel, vous pouvez vous attendre à ce que les images concernées expireront dans les 24 heures après avoir satisfait aux critères d'expiration. Lorsque Amazon ECR exécute une action basée sur une stratégie de cycle de vie, elle est capturée en tant qu'événement dans AWS CloudTrail. Pour de plus amples informations, veuillez consulter [Journalisation des actions Amazon ECR avec AWS CloudTrail](#).

Le diagramme suivant illustre un flux de travail de la politique du cycle de vie.



1. Créez une ou plusieurs règles de test.
2. Enregistrez les règles de test et exécutez l'aperçu.
3. L'évaluateur de la politique de cycle de vie examine toutes les règles et marque les images auxquelles chaque règle doit s'appliquer.

4. L'évaluateur des politiques de cycle de vie applique ensuite les règles, en fonction de la priorité des règles, et affiche les images du référentiel qui sont configurées pour être expirées ou archivées. Un numéro de priorité de règle inférieur signifie une priorité plus élevée. Par exemple, une règle de priorité 1 a priorité sur une règle de priorité 2.
5. Passez en revue les résultats du test pour vous assurer que les images marquées comme expirées ou archivées correspondent à vos attentes.
6. Appliquez les règles de test en tant que politique de cycle de vie du référentiel.
7. Une fois la politique de cycle de vie créée, vous devez vous attendre à ce que les images soient expirées ou archivées dans les 24 heures suivant le respect des critères d'expiration.

## Règles d'évaluation de la politique de cycle de vie

Cet évaluateur de politique de cycle de vie analyse le code JSON en texte brut de la politique de cycle de vie, évalue toutes les règles, puis applique ces règles aux images en fonction de la priorité des règles dans le référentiel. Ce qui suit explique la logique de l'évaluateur de la politique de cycle de vie plus en détails. Pour obtenir des exemples, consultez [Exemples de politiques de cycle de vie dans Amazon ECR](#).

- Lorsque des artefacts de référence sont présents dans un référentiel, les politiques de cycle de vie d'Amazon ECR expirent ou archivent automatiquement ces artefacts dans les 24 heures suivant la suppression ou l'archivage de l'image d'objet.
- Toutes les règles sont évaluées en même temps, quelle que soit la priorité de la règle. Lorsque l'évaluation de toutes les règles est terminée, les règles sont ensuite appliquées en fonction de leur priorité.
- Une image est expirée ou archivée selon exactement une ou aucune règle.
- Une image qui répond aux exigences de balisage d'une règle ne peut pas être expirée ou archivée par une règle de priorité inférieure.
- Les règles ne peuvent jamais marquer les images marquées par des règles de priorité plus élevée, mais elles peuvent tout de même les identifier comme si elles n'avaient pas été expirées ou archivées.
- L'ensemble de règles sélectionnant une classe de stockage spécifique doit contenir un ensemble unique de préfixes.
- Une seule règle sélectionnant une classe de stockage spécifique est autorisée pour sélectionner des images non étiquetées.

- Si une image est référencée par une liste de manifestes, elle ne peut pas être expirée ou archivée sans que la liste de manifestes soit préalablement supprimée ou archivée.
- L'expiration est toujours ordonnée par `pushed_at_time` ou `transitioned_at_time` et fait toujours expirer les anciennes images avant les plus récentes. Si une image a été archivée puis restaurée par le passé, c'est l'image qui `last_activated_at` est utilisée à la place de `pushed_at_time`.
- Une règle de politique de cycle de vie peut spécifier soit `tagPatternList`, soit `tagPrefixList`, mais pas les deux. Cependant, une politique de cycle de vie peut contenir plusieurs règles dans lesquelles différentes règles utilisent à la fois des listes de modèles et de préfixes. Une image est correctement mise en correspondance si toutes les balises de la `tagPrefixList` valeur `tagPatternList` ou sont mises en correspondance avec l'une des balises de l'image.
- Les paramètres `tagPatternList` ou `tagPrefixList` ne peuvent être utilisés que si `tagStatus` est `tagged`.
- Lors de l'utilisation de `tagPatternList`, une image fait l'objet d'une correspondance si elle correspond au filtre de caractère générique. Par exemple, si un filtre de `prod*` est appliqué, il correspondra aux balises d'image dont le nom commence par `prod` tel que `prodprod1`, ou `production-team1`. De même, si un filtre de `*prod*` est appliqué, il correspondra aux balises d'image dont le nom contient une valeur `prod` telle que `repo-production` ou `prod-team`.

#### Important

Il existe une limite maximale de quatre caractères génériques (\*) par chaîne. Par exemple, `["*test*1*2*3", "test*1*2*3*"]` est valide mais `["test*1*2*3*4*5*6"]` ne l'est pas.

- Lors de l'utilisation `tagPrefixList`, une image est correctement mise en correspondance si tous les filtres génériques de la `tagPrefixList` valeur correspondent à l'une des balises de l'image.
- Le `countUnit` paramètre n'est utilisé que si `countType` est `sinceImagePushed`, `sinceImagePulled`, ou `sinceImageTransitioned`.
- Avec `countType = imageCountMoreThan`, les images sont triées de la plus jeune à la plus ancienne en fonction de, `pushed_at_time` puis toutes les images supérieures au nombre spécifié sont expirées ou archivées.
- Avec `countType = sinceImagePushed`, toutes les images dont le nombre de jours `pushed_at_time` est supérieur au nombre de jours spécifié `countNumber` sont expirées ou archivées.

- Avec `countType = sinceImagePulled`, toutes les images dont le nombre de jours `last_recorded_pulltime` est supérieur au nombre de jours spécifié `countNumber` sont archivées. Si une image n'a jamais été extraite, `pushed_at_time` est utilisée à la place de `last_recorded_pulltime`. Si une image a été archivée puis restaurée par le passé, mais qu'elle n'a jamais été extraite depuis sa restauration, `last_activated_at` est l'image qui est utilisée à la place de `last_recorded_pulltime`.
- Avec `countType = sinceImageTransitioned`, toutes les images archivées `last_archived_at` datant de plus de jours que le nombre de jours spécifié `countNumber` sont expirées.
- L'expiration est toujours ordonnée `pushed_at_time` et fait toujours expirer les anciennes images avant les plus récentes.

## Création d'un aperçu de la politique de cycle de vie dans Amazon ECR

Vous pouvez utiliser un aperçu de la politique de cycle de vie pour voir l'impact d'une politique de cycle de vie sur un référentiel d'images avant de l'appliquer. Il est recommandé d'effectuer un aperçu avant d'appliquer une politique de cycle de vie à un référentiel.


### Note

Si vous utilisez la réplication Amazon ECR pour créer des copies d'un référentiel dans différentes régions ou comptes, notez qu'une politique de cycle de vie ne peut agir que sur les référentiels de la région dans laquelle il a été créé. Par conséquent, si la réplication est activée, vous pouvez envisager de créer une politique de cycle de vie dans chaque région et chaque compte vers lesquels vous répliquez vos référentiels.

### Créer une politique de cycle de vie (AWS Management Console)


1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel sur lequel exécuter un aperçu de la politique de cycle de vie.
3. Dans le volet de navigation, sous **Registre privé**, choisissez **Référentiels**.

4. Sur la page Référentiels privés, sélectionnez un référentiel et utilisez le menu déroulant Actions pour choisir les Politiques de cycle de vie.
5. Sur la page des règles de la politique de cycle de vie du référentiel, choisissez Modifier les règles de test, Créer une règle.
6. Spécifiez les détails suivants pour chaque règle de politique de cycle de vie de test.
  - a. Pour Priorité d'une règle, saisissez un nombre pour la priorité de la règle. La priorité d'une règle détermine l'ordre dans lequel les règles de politique de cycle de vie sont appliquées. Un chiffre inférieur signifie une priorité plus élevée. Par exemple, une règle de priorité 1 a priorité sur une règle de priorité 2.
  - b. Pour Description de la règle, saisissez une description pour la règle de la politique de cycle de vie.
  - c. Pour Statut d'image, choisissez Balisée (correspondance par caractère générique), Balisée (correspondance par préfixe), Non balisée ou Toute.

 Important

Si vous précisez plusieurs étiquettes, seules les images portant toutes les étiquettes précisées seront sélectionnées.

- d. Si vous choisissez Balisée (correspondance par caractère générique) pour Statut d'image, vous pouvez alors spécifier une liste de balises d'image avec un caractère générique (\*) sur lesquelles prendre des mesures conformément à votre politique de cycle de vie pour Spécifier les balises pour la correspondance par caractère générique. Par exemple, si vos images sont balisées comme prod, prod1, prod2, et ainsi de suite, vous devrez spécifier prod\* afin d'appliquer des mesures à toutes les images. Si vous précisez plusieurs étiquettes, seules les images portant toutes les étiquettes précisées seront sélectionnées.

 Important

Il existe une limite maximale de quatre caractères génériques (\*) par chaîne. Par exemple, ["\*test\*1\*2\*3", "test\*1\*2\*3\*"] est valide mais ["test\*1\*2\*3\*4\*5\*6"] ne l'est pas.

- e. Si vous choisissez Balisée (correspondance par préfixe) pour Statut d'image, vous pouvez alors spécifier une liste de balises d'image sur lesquelles prendre des mesures

- conformément à votre politique de cycle de vie pour Spécifier les balises pour la correspondance par préfixe.
- f. Pour les critères de correspondance, choisissez Jours depuis la création de l'image, Jours depuis la dernière date d'extraction enregistrée, Jours depuis l'archivage de l'image ou Nombre d'images, puis spécifiez une valeur.
  - g. Pour Action par règle, choisissez Expirer ou Archiver.
  - h. Choisissez Enregistrer.
7. Créez des règles de politique de cycle de vie de test supplémentaires en répétant les étapes 5 à 7.
  8. Pour exécuter l'aperçu de la politique de cycle de vie, choisissez Enregistrer et exécuter le test.
  9. Sous Correspondances d'images pour les règles de cycle de vie test), vérifiez l'impact de l'aperçu de votre politique de cycle de vie.
  10. Si les résultats de l'aperçu sont satisfaisants, choisissez Appliquer la politique de cycle de vie pour créer une politique de cycle de vie avec les règles indiquées. Après l'application d'une politique de cycle de vie, vous devez vous attendre à ce que les images concernées expirent ou soient archivées dans les 24 heures.
  11. Si vous n'êtes pas satisfait des résultats de l'aperçu, vous pouvez supprimer une ou plusieurs règles de cycle de vie du test et créer une ou plusieurs règles pour les remplacer, puis répéter le test.

## Création d'une politique de cycle de vie pour un référentiel dans Amazon ECR

Utilisez une politique de cycle de vie pour créer un ensemble de règles qui expirent ou archivent les images de référentiel non utilisées. Après avoir créé une politique de cycle de vie, les images concernées expirent ou sont archivées dans les 24 heures.

### Note


Si vous utilisez la réplification Amazon ECR pour créer des copies d'un référentiel dans différentes régions ou comptes, notez qu'une politique de cycle de vie ne peut agir que sur les référentiels de la région dans laquelle il a été créé. Par conséquent, si la réplification est activée, vous pouvez envisager de créer une politique de cycle de vie dans chaque région et chaque compte vers lesquels vous répliquez vos référentiels.

## Prérequis

Bonne pratique : créez un aperçu de la politique de cycle de vie pour vérifier que les images expirées ou archivées conformément à vos règles de politique de cycle de vie correspondent à vos attentes. Pour obtenir des instructions, veuillez consulter [Création d'un aperçu de la politique de cycle de vie dans Amazon ECR](#).

### Créer une politique de cycle de vie (AWS Management Console)

1. Ouvrez la console Amazon ECR dans les <https://console.aws.amazon.com/ecr/référentiels>.
2. Dans la barre de navigation, choisissez la région qui contient le référentiel sur lequel créer une politique de cycle de vie.
3. Dans le volet de navigation, sous Registre privé, choisissez Référentiels.
4. Sur la page Référentiels privés, sélectionnez un référentiel et utilisez le menu déroulant Actions pour choisir les Politiques de cycle de vie.
5. Sur la page des règles de la politique de cycle de vie du référentiel, choisissez Créer une règle.
6. Saisissez les détails suivants pour votre règle de politique de cycle de vie.
  - a. Pour Priorité d'une règle, saisissez un nombre pour la priorité de la règle. La priorité d'une règle détermine l'ordre dans lequel les règles de politique de cycle de vie sont appliquées. Un numéro de priorité de règle inférieur signifie une priorité plus élevée. Par exemple, une règle de priorité 1 a priorité sur une règle de priorité 2.
  - b. Pour Description de la règle, saisissez une description pour la règle de la politique de cycle de vie.
  - c. Pour Statut d'image, choisissez Balisée (correspondance par caractère générique), Balisée (correspondance par préfixe), Non balisée ou Toute.

 **Important**

Si vous précisez plusieurs étiquettes, seules les images portant toutes les étiquettes précisées seront sélectionnées.
  - d. Si vous choisissez Balisée (correspondance par caractère générique) pour Statut d'image, vous pouvez alors spécifier une liste de balises d'image avec un caractère générique (\*) sur lesquelles prendre des mesures conformément à votre politique de cycle de vie pour Spécifier les balises pour la correspondance par caractère générique. Par exemple, si vos

images sont balisées comme `prod`, `prod1`, `prod2`, et ainsi de suite, vous devrez spécifier `prod*` afin d'appliquer des mesures à toutes les images. Si vous précisez plusieurs étiquettes, seules les images portant toutes les étiquettes précisées seront sélectionnées.

**⚠ Important**

Il existe une limite maximale de quatre caractères génériques (\*) par chaîne. Par exemple, `["*test*1*2*3", "test*1*2*3*"]` est valide mais `["test*1*2*3*4*5*6"]` ne l'est pas.

- e. Si vous choisissez Balisée (correspondance par préfixe) pour Statut d'image, vous pouvez alors spécifier une liste de balises d'image sur lesquelles prendre des mesures conformément à votre politique de cycle de vie pour Spécifier les balises pour la correspondance par préfixe.
  - f. Pour les critères de correspondance, choisissez Jours depuis la création de l'image, Jours depuis la dernière date d'extraction enregistrée, Jours depuis l'archivage de l'image ou Nombre d'images, puis spécifiez une valeur.
  - g. Pour Action par règle, choisissez Expirer ou Archiver.
  - h. Choisissez Enregistrer.
7. Créez des règles de politique de cycle de vie supplémentaires en répétant les étapes 5 à 7.

## Créer une politique de cycle de vie (AWS CLI)

1. Obtenez le nom du référentiel pour lequel créer la politique de cycle de vie.

```
aws ecr describe-repositories
```

2. Créez un fichier local nommé `policy.json` avec le contenu de la politique de cycle de vie. Pour obtenir des exemples de politiques de cycle de vie, consultez [Exemples de politiques de cycle de vie dans Amazon ECR](#).
3. Créez une politique de cycle de vie en indiquant le nom du référentiel, puis référencez le fichier JSON de la politique de cycle de vie que vous avez créé.

```
aws ecr put-lifecycle-policy \
 --repository-name repository-name \
 --lifecycle-policy-text file://policy.json
```

# Exemples de politiques de cycle de vie dans Amazon ECR

Voici des exemples de politiques de cycle de vie illustrant la syntaxe.

Pour plus d'informations sur les propriétés des politiques, consultez [Propriétés de la politique de cycle de vie dans Amazon ECR](#). Pour obtenir des instructions sur la création d'une politique de cycle de vie à l'aide du AWS CLI, consultez [Créer une politique de cycle de vie \(AWS CLI\)](#).

## Modèle de politique de cycle de vie

Le contenu de votre politique de cycle de vie est évalué avant d'être associé à un référentiel. Voici le modèle de syntaxe JSON pour la politique de cycle de vie.

```
{
 "rules": [
 {
 "rulePriority": integer,
 "description": "string",
 "selection": {
 "tagStatus": "tagged"|"untagged"|"any",
 "tagPatternList": list<string>,
 "tagPrefixList": list<string>,
 "storageClass": "standard"|"archive",
 "countType":
"imageCountMoreThan"|"sinceImagePushed"|"sinceImagePulled"|"sinceImageTransitioned",
 "countUnit": "string",
 "countNumber": integer
 },
 "action": {
 "type": "expire"|"transition",
 "targetStorageClass": "archive"
 }
 }
]
}
```

## Filtrer sur l'ancienneté des images

Voici un exemple de syntaxe d'une politique de cycle de vie qui fait expirer les images qui ont une balise commençant par prod à l'aide d'une tagPatternList de prod\* qui ont également plus de 14 jours d'ancienneté.

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images older than 14 days",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["prod*"],
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 14
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

## Filtrage en fonction de l'heure de la dernière extraction

L'exemple suivant montre la syntaxe de la politique de cycle de vie d'une politique qui transfère les images vers le stockage d'archives qui n'ont pas été extraites depuis des 90 jours.

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Archive images not pulled in 90 days",
 "selection": {
 "tagStatus": "any",
 "countType": "sinceImagePulled",
 "countUnit": "days",
 "countNumber": 90
 },
 "action": {
 "type": "transition",
 "targetStorageClass": "archive"
 }
 }
]
}
```

**⚠ Important**

Le type de `sinceImagePulled` comptage doit être utilisé avec l'`transitionaction`. Il ne peut pas être utilisé avec l'`expireaction`. Pour supprimer des images en fonction de l'activité d'extraction, transportez-les d'abord vers le stockage d'archives à l'`sinceImageTransitionedaide sinceImagePulled` d'une `expire action` pour les supprimer. Les images doivent être stockées dans des archives pendant au moins 90 jours avant d'être supprimées.

## Filtrage sur le temps de transition des archives

L'exemple suivant montre la syntaxe de la politique de cycle de vie d'une politique qui fait expirer les images archivées stockées depuis plus de 365 jours.

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images archived for more than 365 days",
 "selection": {
 "tagStatus": "any",
 "storageClass": "archive",
 "countType": "sinceImageTransitioned",
 "countUnit": "days",
 "countNumber": 365
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

**⚠ Important**

Le type de `sinceImageTransitioned` compte doit être utilisé avec l'`expireaction` et la classe `archive` de stockage. Les images doivent être stockées dans des archives pendant au moins 90 jours avant d'être supprimées.

## Filtrer sur le décompte d'images

Voici un exemple de syntaxe d'une politique de cycle de vie qui conserve uniquement une image non balisée et fait expirer toutes les autres.

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Keep only one untagged image, expire all others",
 "selection": {
 "tagStatus": "untagged",
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

## Filtrer sur plusieurs règles

Les exemples suivants utilisent plusieurs règles dans une politique de cycle de vie. Des exemples de référentiel et de politique de cycle de vie sont fournis avec une explication du résultat.

### Exemple A

Contenu du référentiel :

- Image A, Taglist: ["beta-1", "prod-1"], Transmise : il y a 10 jours
- Image B, Taglist: ["beta-2", "prod-2"], Transmise : il y a 9 jours
- Image C, Taglist: ["beta-3"], Transmise : il y a 8 jours

Texte de la politique de cycle de vie :

```
{
 "rules": [
```

```
{
 "rulePriority": 1,
 "description": "Rule 1",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["prod*"],
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
},
{
 "rulePriority": 2,
 "description": "Rule 2",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["beta*"],
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
}
]
```

La logique de cette politique de cycle de vie serait :

- La règle 1 identifie les images étiquetées avec le préfixe prod. Elle doit marquer les images, en commençant par la plus ancienne, jusqu'à ce qu'il reste une image qui corresponde, ou moins. Elle marque l'image A pour expiration.
- La règle 2 identifie les images étiquetées avec le préfixe beta. Elle doit marquer les images, en commençant par la plus ancienne, jusqu'à ce qu'il reste une image qui corresponde, ou moins. Elle marque l'image A et l'image B pour expiration. Cependant, l'image A a déjà été examinée par la règle 1 et, si l'image B expirait, cela violerait la règle 1. L'image B est donc ignorée.
- Résultat : L'image A est expirée.

## Exemple B

Il s'agit du même référentiel que dans l'exemple précédent, mais l'ordre de priorité des règles est modifié pour illustrer le résultat.

Contenu du référentiel :

- Image A, Taglist: ["beta-1", "prod-1"], Transmise : il y a 10 jours
- Image B, Taglist: ["beta-2", "prod-2"], Transmise : il y a 9 jours
- Image C, Taglist: ["beta-3"], Transmise : il y a 8 jours

Texte de la politique de cycle de vie :

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Rule 1",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["beta*"],
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
 },
 {
 "rulePriority": 2,
 "description": "Rule 2",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["prod*"],
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

```
}
```

La logique de cette politique de cycle de vie serait :

- La règle 1 identifie les images étiquetées avec le préfixe `beta`. Elle doit marquer les images, en commençant par la plus ancienne, jusqu'à ce qu'il reste une image qui corresponde, ou moins. Elle examine les trois images, et marque l'image A et l'image B pour expiration.
- La règle 2 identifie les images étiquetées avec le préfixe `prod`. Elle doit marquer les images, en commençant par la plus ancienne, jusqu'à ce qu'il reste une image qui corresponde, ou moins. Elle n'examine aucune image, car toutes les images disponibles ont déjà été examinées par la règle 1. Elle ne marque donc aucune image supplémentaire.
- Résultat : Les images A et B sont expirées.

## Filtrer sur plusieurs étiquettes dans une seule règle

Les exemples suivants indiquent la syntaxe de la politique de cycle de vie pour plusieurs modèles de balises dans une seule règle. Des exemples de référentiel et de politique de cycle de vie sont fournis avec une explication du résultat.

### Exemple A

Lorsque plusieurs modèles de balises sont indiqués dans une seule règle, les images doivent correspondre à tous les modèles de balises répertoriés.

Contenu du référentiel :

- Image A, Taglist: ["alpha-1"], Transmise : il y a 12 jours
- Image B, Taglist: ["beta-1"], Transmise : il y a 11 jours
- Image C, Taglist: ["alpha-2", "beta-2"], Transmise : il y a 10 jours
- Image D, Taglist: ["alpha-3"], Transmise : il y a 4 jours
- Image E, Taglist: ["beta-3"], Transmise : il y a 3 jours
- Image F, Taglist: ["alpha-4", "beta-4"], Transmise : il y a 2 jours

```
{
 "rules": [
 {
```

```
 "rulePriority": 1,
 "description": "Rule 1",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["alpha*", "beta*"],
 "countType": "sinceImagePushed",
 "countNumber": 5,
 "countUnit": "days"
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

La logique de cette politique de cycle de vie serait :

- La règle 1 identifie les images balisées par les préfixes alpha et beta. Elle examine les images C et F. Elle doit marquer les images de plus de cinq jours, ce qui est le cas de l'image C.
- Résultat : L'image C est expirée.

## Exemple B

L'exemple suivant montre que des étiquettes ne sont pas exclusives.

Contenu du référentiel :

- Image A, Taglist: ["alpha-1", "beta-1", "gamma-1"], Transmise : il y a 10 jours
- Image B, Taglist: ["alpha-2", "beta-2"], Transmise : il y a 9 jours
- Image C, Taglist: ["alpha-3", "beta-3", "gamma-2"], Transmise : il y a 8 jours

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Rule 1",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["alpha*", "beta*"],
```

```
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
}
]
```

La logique de cette politique de cycle de vie serait :

- La règle 1 identifie les images balisées par les préfixes alpha et beta. Elle examine toutes les images. Elle doit marquer les images, en commençant par la plus ancienne, jusqu'à ce qu'il reste une image qui corresponde, ou moins. Elle marque les images A et B pour expiration.
- Résultat : Les images A et B sont expirées.

## Filtrer sur toutes les images

Les exemples de politique de cycle de vie suivants indiquent toutes les images avec différents filtres. Des exemples de référentiel et de politique de cycle de vie sont fournis avec une explication du résultat.

### Exemple A

Voici la syntaxe d'une politique de cycle de vie qui applique toutes les règles, mais conserve uniquement une image et fait expirer toutes les autres.

Contenu du référentiel :

- Image A, Taglist: ["alpha-1"], Transmise : il y a 4 jours
- Image B, Taglist: ["beta-1"], Transmise : il y a 3 jours
- Image C, Taglist: [], Transmise : il y a 2 jours
- Image D, Taglist: ["alpha-2"], Transmise : il y a 1 jour

```
{
 "rules": [
 {
```

```
 "rulePriority": 1,
 "description": "Rule 1",
 "selection": {
 "tagStatus": "any",
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

La logique de cette politique de cycle de vie serait :

- La règle 1 identifie toutes les images. Elle examine les images A, B, C et D. Elle doit faire expirer toutes les images, à l'exception de la plus récente. Elle marque les images A, B et C pour expiration.
- Résultat : Les images A, B et C expirent.

## Exemple B

L'exemple suivant illustre une politique de cycle de vie qui combine tous les types de règles dans une seule politique.

Contenu du référentiel :

- Image A, Taglist: ["alpha-1", "beta-1"], Transmise : il y a 4 jours
- Image B, Taglist: [], Transmise : il y a 3 jours
- Image C, Taglist: ["alpha-2"], Transmise : il y a 2 jours
- Image D, Taglist: ["git hash"], Transmise : il y a 1 jour
- Image E, Taglist: [], Transmise : il y a 1 jour

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Rule 1",
```

```
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["alpha*"],
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
 },
 {
 "rulePriority": 2,
 "description": "Rule 2",
 "selection": {
 "tagStatus": "untagged",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
 },
 {
 "rulePriority": 3,
 "description": "Rule 3",
 "selection": {
 "tagStatus": "any",
 "countType": "imageCountMoreThan",
 "countNumber": 1
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

La logique de cette politique de cycle de vie serait :

- La règle 1 identifie les images étiquetées avec le préfixe `alpha`. Elle identifie les images A et C. Elle doit conserver l'image la plus récente et marquer les autres pour expiration. Elle marque l'image A pour expiration.

- La règle 2 identifie les images non étiquetées. Elle identifie les images B et E. Elle doit marquer toutes les images datant de plus d'un jour pour expiration. Elle marque l'image B pour expiration.
- La règle 3 identifie toutes les images. Elle identifie les images A, B, C, D et E. Elle doit conserver l'image la plus récente et marquer les autres pour expiration. Cependant, elle ne peut pas marquer les images A, B, C ou E, car elles ont été identifiées par des règles de priorité plus haute. Elle marque l'image D pour expiration.
- Résultat : Les images A, B et D expirent.

## Exemples d'Archive

Les exemples suivants présentent des politiques de cycle de vie qui archivent les images au lieu de les supprimer.

### Archivage d'images datant de plus d'un certain nombre de jours

L'exemple suivant illustre une politique de cycle de vie qui archive les images dont les balises commencent par et prod datent de plus de 30 jours :

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Archive production images older than 30 days",
 "selection": {
 "tagStatus": "tagged",
 "tagPatternList": ["prod*"],
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 30
 },
 "action": {
 "type": "transition",
 "targetStorageClass": "archive"
 }
 }
]
}
```

## Archivage des images non extraites pendant un certain nombre de jours

L'exemple suivant illustre une politique de cycle de vie qui archive les images qui n'ont pas été extraites depuis 90 jours :

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Archive images not pulled in 90 days",
 "selection": {
 "tagStatus": "any",
 "countType": "sinceImagePulled",
 "countUnit": "days",
 "countNumber": 90
 },
 "action": {
 "type": "transition",
 "targetStorageClass": "archive"
 }
 }
]
}
```

## Combinaison des règles d'archivage et d'expiration

L'exemple suivant illustre une politique de cycle de vie qui archive les images datant de plus de 30 jours, puis fait expirer définitivement les images archivées depuis plus de 365 jours :

### Note

Les images archivées ont une durée de stockage minimale de 90 jours. Vous ne pouvez pas configurer de politiques de cycle de vie qui suppriment les images archivées depuis moins de 90 jours. Si vous devez supprimer des images archivées depuis moins de 90 jours, vous devez utiliser l'`batch-delete-imageAPI`, mais la durée de stockage minimale de 90 jours vous sera facturée.

```
{
 "rules": [
```

```
{
 "rulePriority": 1,
 "description": "Archive images older than 30 days",
 "selection": {
 "tagStatus": "any",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 30
 },
 "action": {
 "type": "transition",
 "targetStorageClass": "archive"
 }
},
{
 "rulePriority": 2,
 "description": "Expire images archived for more than 365 days",
 "selection": {
 "tagStatus": "any",
 "storageClass": "archive",
 "countType": "sinceImageTransitioned",
 "countUnit": "days",
 "countNumber": 365
 },
 "action": {
 "type": "expire"
 }
}
]
```

## Propriétés de la politique de cycle de vie dans Amazon ECR

Les politiques de cycle de vie présentent les propriétés suivantes.

Pour consulter des exemples de politiques relatives au cycle de vie, consultez [Exemples de politiques de cycle de vie dans Amazon ECR](#). Pour obtenir des instructions sur la création d'une politique de cycle de vie à l'aide du AWS CLI, consultez [Créer une politique de cycle de vie \(AWS CLI\)](#).

## Priorité de la règle

### rulePriority

Type : entier

Obligatoire : oui

Définit l'ordre dans lequel les règles sont évaluées, de la priorité la plus basse à la plus haute. Une règle de politique de cycle de vie avec une priorité de 1 est appliquée en premier, une règle avec une priorité de 2 est appliquée ensuite, et ainsi de suite. Lorsque vous ajoutez des règles à une politique de cycle de vie, vous devez attribuer à chacune une valeur unique de `rulePriority`. Les valeurs n'ont pas besoin d'être séquentielles entre les règles d'une politique. Une règle avec une valeur `tagStatus` de `any` doit avoir la valeur la plus élevée pour `rulePriority` et être évaluée en dernier.

## Description

### description

Type : chaîne

Obligatoire : non

(Facultatif) Décrit l'objectif d'une règle dans une politique de cycle de vie.

## État de l'étiquetage

### tagStatus

Type : chaîne

Obligatoire : oui

Détermine si la règle de la politique de cycle de vie que vous ajoutez précise une étiquette pour une image. Les options acceptables sont `tagged`, `untagged` ou `any`. Si vous précisez `any`, la règle s'appliquera à toutes les images évaluées par la règle. Si vous spécifiez `tagged`, vous devez également spécifier une `tagPrefixList` valeur ou une `tagPatternList` valeur. Si vous le spécifiez `untagged`, vous devez omettre à la fois `tagPrefixList` et `tagPatternList`.

## Liste des modèles de balises

### tagPatternList

Type : list[string]

Obligatoire : oui, si tagStatus est défini sur balisé et tagPrefixList n'est pas spécifiée

Lors de la création d'une politique de cycle de vie pour les images balisées, il est recommandé d'utiliser une tagPatternList pour spécifier les balises à expirer. Précisez une liste séparée par des virgules de modèles de balises d'image pouvant contenir des caractères génériques (\*) sur lesquels exécuter une action avec votre politique de cycle de vie. Par exemple, si vos images sont balisées comme prod, prod1, prod2, et ainsi de suite, vous devrez utiliser le modèle de balise prod\* pour les spécifier toutes. Si vous précisez plusieurs étiquettes, seules les images portant toutes les étiquettes précisées seront sélectionnées.

#### Important

Il existe une limite maximale de quatre caractères génériques (\*) par chaîne. Par exemple, ["\*test\*1\*2\*3", "test\*1\*2\*3\*"] est valide mais ["test\*1\*2\*3\*4\*5\*6"] ne l'est pas.

## Liste des préfixes d'étiquette

### tagPrefixList

Type : list[string]

Obligatoire : oui, si tagStatus est défini sur balisé et tagPatternList n'est pas spécifiée

Uniquement utilisé si vous avez spécifié "tagStatus": "tagged" et que vous ne spécifiez pas une tagPatternList. Vous devez préciser une liste séparée par des virgules de préfixes d'étiquette d'image sur lesquels exécuter une action avec votre politique de cycle de vie. Par exemple, si vos images sont étiquetées comme prod, prod1, prod2, et ainsi de suite, vous devrez utiliser le préfixe d'étiquette prod pour toutes les préciser. Si vous précisez plusieurs étiquettes, seules les images portant toutes les étiquettes précisées seront sélectionnées.

## Classe de stockage

### storageClass

Type : chaîne

Obligatoire : oui, si `countType` c'est le cas `sinceImageTransitioned`

La règle ne sélectionnera que les images de cette classe de stockage. Lorsque vous utilisez un `countType` of `imageCountMoreThansinceImagePushed`, ou `sinceImagePulled`, la seule valeur prise en charge est `standard`. Lorsque vous utilisez un type de comptage de `sinceImageTransitioned`, cela est obligatoire et la seule valeur prise en charge est `archive`. Si vous l'omettez, la valeur de `standard` sera utilisée.

## Type de décompte

### countType

Type : chaîne

Obligatoire : oui

Indiquez un type de décompte à appliquer aux images.

Si `countType` est défini sur `imageCountMoreThan`, vous précisez également `countNumber` pour créer une règle qui définit une limite sur le nombre d'images existant dans votre référentiel. `countType` est défini sur `sinceImagePushed`, ou `sinceImagePulled` `sinceImageTransitioned`, vous spécifiez également `countUnit` et `countNumber` pour spécifier une limite de temps pour les images qui existent dans votre référentiel.

## Unité de décompte

### countUnit

Type : chaîne

Obligatoire : oui, uniquement si `countType` est défini sur `sinceImagePushed` `sinceImagePulled`, ou `sinceImageTransitioned`

Précisez une unité de décompte days pour indiquer celle-ci comme unité de temps, en plus de `countNumber`, qui est le nombre de jours.

Cela ne doit être spécifié que lorsque `countType` est `sinceImagePushed`, `sinceImagePulled`, ou `sinceImageTransitioned` ; une erreur se produira si vous spécifiez une unité de comptage alors qu'il `countType` s'agit d'une autre valeur.

## Chiffre du décompte

`countNumber`

Type : entier

Obligatoire : oui

Précisez un chiffre de décompte. Les valeurs acceptables sont des entiers positifs (0 n'est pas une valeur acceptée).

Si le paramètre `countType` utilisé est `imageCountMoreThan`, la valeur sera le nombre maximal d'images que vous souhaitez conserver dans votre référentiel. Si le paramètre `countType` utilisé est `sinceImagePushed`, la valeur sera la limite d'ancienneté maximale pour vos images. Si `countType` est le cas `sinceImagePulled`, la valeur est le nombre maximum de jours écoulés depuis la dernière extraction de l'image. Si `countType` est le cas `sinceImageTransitioned`, la valeur est le nombre maximal de jours écoulés depuis l'archivage de l'image.

## Action

`type`

Type : chaîne

Obligatoire : oui

Précisez un type d'action. Les valeurs prises en charge sont `expire` (pour supprimer des images) et `transition` (pour déplacer les images vers le stockage d'archives).

`targetStorageClass`

Type : chaîne

Obligatoire : oui, si `type` est le cas `transition`

Classe de stockage vers laquelle vous souhaitez que la politique de cycle de vie transfère l'image. `archive` est la seule valeur prise en charge.

# Exclusions liées aux mises à jour instantanées

Amazon ECR met à jour l'`LastRecordedPullTime` à chaque extraction, à l'exception des extractions effectuées par Inspector. AWS Les exclusions de mise à jour au moment de l'extraction vous permettent de spécifier le rôle IAM ARNs qui ne doit pas mettre à jour les temps d'extraction des images lorsqu'ils extraient des images, tels que les extractions effectuées par des scanners tiers (tels que Crowdstrike, Snyk et Trivy). Cela est utile pour les images utilisées à des CI/CD fins de test ou pour lesquelles vous ne souhaitez pas que le temps d'extraction ait une incidence sur les décisions relatives au cycle de vie.

Lorsqu'un rôle de la liste d'exclusion extrait une image, le temps d'extraction reste inchangé. Tout autre rôle continue de mettre à jour le temps d'appel (comportement actuel). Vous pouvez configurer jusqu'à 100 exclusions par compte.

## Gestion des exclusions liées aux mises à jour effectuées au moment du téléchargement

Pour gérer les exclusions liées aux mises à jour automatiques, vous devez disposer des autorisations IAM suivantes :

- `ecr:CreatePullTimeUpdateExclusion`— Accorde l'autorisation d'ajouter un ARN de rôle à la liste d'exclusion.
- `ecr>DeletePullTimeUpdateExclusion`— Accorde l'autorisation de supprimer un ARN de rôle de la liste d'exclusion.
- `ecr:ListPullTimeUpdateExclusions`— Accorde l'autorisation de répertorier tous les rôles ARNs dans la liste d'exclusion.

### Note

Tu n'as pas besoin d'`iam:PassRole` autorisation. Amazon ECR n'assume pas le rôle d'effectuer une action ; il utilise uniquement la configuration d'exclusion ARNs pour déterminer si le temps d'extraction de l'image doit être mis à jour.

Vous pouvez gérer les exclusions de mise à jour au moment de l'appel à l'aide de la console Amazon ECR ou de la CLI. AWS

## AWS Management Console

Pour gérer les exclusions liées aux mises à jour automatiques ( )AWS Management Console

1. [Ouvrez la console Amazon ECR sur private-registry/repositories https://console.aws.amazon.com/ecr/](https://console.aws.amazon.com/ecr/)
2. Dans la barre de navigation, choisissez la région.
3. Dans le volet de navigation, choisissez Registre privé, Fonctionnalités et paramètres, puis choisissez Exclusions de mise à jour par extraction.
4. Pour ajouter une exclusion, choisissez Ajouter une exclusion, entrez l'ARN du rôle, puis choisissez Ajouter.
5. Pour supprimer une exclusion, sélectionnez l'ARN du rôle dans la liste et choisissez Supprimer.
6. Pour afficher toutes les exclusions, la liste affiche tous les rôles configurés ARNs.

## AWS CLI

Pour créer une exclusion de mise à jour automatique

- Utilisez la `create-pull-time-update-exclusion` commande pour ajouter un ARN de rôle à la liste d'exclusion :

```
aws ecr create-pull-time-update-exclusion \
 --role-arn arn:aws:iam::123456789012:role/scanner-role
```

La commande renvoie l'ARN du rôle et l'horodatage de création :

```
{
 "roleArn": "arn:aws:iam::123456789012:role/scanner-role",
 "createdAt": 1745531331.0
}
```

Pour supprimer une exclusion de mise à jour automatique

- Utilisez la `delete-pull-time-update-exclusion` commande pour supprimer un ARN de rôle de la liste d'exclusion :

```
aws ecr delete-pull-time-update-exclusion \
 --role-arn arn:aws:iam::123456789012:role/scanner-role
```

La commande renvoie l'ARN du rôle qui a été supprimé :

```
{
 "roleArn": "arn:aws:iam::123456789012:role/scanner-role"
}
```

Pour répertorier les exclusions liées aux mises à jour par pull-time

1. Utilisez la `list-pull-time-update-exclusions` commande pour répertorier tous les rôles ARNs de la liste d'exclusion :

```
aws ecr list-pull-time-update-exclusions
```

Si aucune exclusion n'est configurée, la commande renvoie une liste vide :

```
{
 "pullTimeUpdateExclusions": []
}
```

Si des exclusions sont configurées, la commande renvoie la liste des rôles ARNs :

```
{
 "pullTimeUpdateExclusions": [
 "arn:aws:iam::123456789012:role/security-role"
]
}
```

2. Pour paginer les résultats, utilisez les `--next-token` paramètres `--max-results` et :

```
aws ecr list-pull-time-update-exclusions \
 --max-results 4
```

La commande renvoie le nombre de résultats spécifié et un `nextToken` si d'autres résultats sont disponibles :

```
{
 "pullTimeUpdateExclusions": [
 "arn:aws:iam::123456789012:role/security-role1",
 "arn:aws:iam::123456789012:role/security-role2",
 "arn:aws:iam::123456789012:role/security-role3",
 "arn:aws:iam::123456789012:role/security-role4"
],
 "nextToken": "ukD72mdD/mC8b5xV3susmJzzaTgp3hKwR9nRUW1yZZ79..."
}
```

Pour récupérer la page de résultats suivante, utilisez `nextToken` la réponse précédente :

```
aws ecr list-pull-time-update-exclusions \
 --max-results 4 \
 --next-token ukD72mdD/mC8b5xV3susmJzzaTgp3hKwR9nRUW1yZZ79...
```

## Considérations relatives aux exclusions liées aux mises à jour automatiques

Tenez compte des points suivants lorsque vous utilisez des exclusions de mise à jour par extraction :

- La taille de page par défaut pour les exclusions d'annonces est de 100. Vous pouvez utiliser la pagination avec `maxResults` et `nextToken` paramètres.
- Seuls les rôles IAM valides ARNs au format ARN correct sont acceptés.
- Si vous essayez de créer une exclusion qui existe déjà, vous recevrez un `ExclusionAlreadyExistsException` message d'erreur. Si vous essayez de supprimer une exclusion qui n'existe pas, vous recevrez un `ExclusionNotFoundException` message d'erreur.

# Sécurité dans le registre de conteneur Amazon Elastic

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le [modèle de responsabilité partagée](#) décrit cette notion par les termes sécurité du cloud et sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Third-partyles auditeurs testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des [programmes de AWS conformité](#). Pour en savoir plus sur les programmes de conformité qui s'appliquent à Amazon ECR, consultez [Services AWS concernés par le programme de conformité](#).
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lorsque vous utiliser Amazon ECR. Les rubriques suivantes vous expliquent comment configurer Amazon ECR afin de répondre à vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources Amazon ECR.

## Rubriques

- [Gestion des identités et des accès au registre de conteneur Amazon Elastic](#)
- [Protection des données dans Amazon ECR](#)
- [Validation de conformité pour le registre de conteneur Amazon Elastic](#)
- [Sécurité de l'infrastructure dans le registre de conteneur Amazon Elastic](#)
- [Cross-service prévention confuse des adjoints](#)

# Gestion des identités et des accès au registre de conteneur Amazon Elastic

Gestion des identités et des accès AWS (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Des administrateurs IAM contrôlent les personnes qui peuvent être authentifiées (connectées) et autorisées (disposant d'autorisations) à utiliser des ressources Amazon ECR. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

## Rubriques

- [Public ciblé](#)
- [Authentification par des identités](#)
- [Gestion de l'accès à l'aide de politiques](#)
- [Fonctionnement du registre de conteneur Amazon Elastic avec IAM](#)
- [Exemples de Identity-based politiques Amazon Elastic Container Registry](#)
- [Utilisation du contrôle Tag-Based d'accès](#)
- [AWS politiques gérées pour Amazon Elastic Container Registry](#)
- [Utilisation des rôles liés à un service pour Amazon ECR](#)
- [Dépannage de l'identité et de l'accès au registre de conteneur Amazon Elastic](#)

## 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 [Dépannage de l'identité et de l'accès au registre de conteneur Amazon Elastic](#))
- Administrateur du service : déterminez l'accès des utilisateurs et soumettez les demandes d'autorisation (voir [Fonctionnement du registre de conteneur Amazon Elastic avec IAM](#))
- Administrateur IAM : rédigez des politiques pour gérer l'accès (voir [Exemples de Identity-based politiques Amazon Elastic Container Registry](#))

## Authentification par des identités

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en tant qu'identité fédérée à l'aide d'informations d'identification provenant d'une source d'identité telle que AWS IAM Identity Center (IAM Identity Center), d'une authentification unique ou d'informations d'identification. Google/Facebook Pour plus d'informations sur la connexion, consultez [Connexion à votre Compte AWS](#) dans le Guide de l'utilisateur Connexion à AWS .

Pour l'accès par programmation, AWS fournit un SDK et une CLI pour signer les demandes de manière cryptographique. Pour plus d'informations, consultez [Signature AWS Version 4 pour les demandes d'API](#) dans le Guide de l'utilisateur IAM.

### Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une seule identité de connexion appelée utilisateur Compte AWS root qui dispose d'un accès complet à toutes Services AWS les ressources. Il est vivement déconseillé d'utiliser l'utilisateur racine pour vos tâches quotidiennes. Pour les tâches qui requièrent des informations d'identification de l'utilisateur racine, consultez [Tâches qui requièrent les informations d'identification de l'utilisateur racine](#) dans le Guide de l'utilisateur IAM.

### Utilisateurs et groupes IAM

Un [utilisateur IAM](#) est une identité qui dispose d'autorisations spécifiques pour une seule personne ou application. Nous vous recommandons d'utiliser ces informations d'identification temporaires au lieu des utilisateurs IAM avec des informations d'identification à long terme. Pour plus d'informations, voir [Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires](#) dans le guide de l'utilisateur IAM.

[Les groupes IAM](#) spécifient une collection d'utilisateurs IAM et permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Pour plus d'informations, consultez [Cas d'utilisation pour les utilisateurs IAM](#) dans le Guide de l'utilisateur IAM.

### Rôles IAM

Un [rôle IAM](#) est une identité dotée d'autorisations spécifiques qui fournit des informations d'identification temporaires. Vous pouvez assumer un rôle en [passant d'un rôle d'utilisateur à un rôle](#)

[IAM \(console\)](#) ou en appelant une opération d' AWS API AWS CLI ou d'API. Pour plus d'informations, consultez [Méthodes pour endosser un rôle](#) dans le Guide de l'utilisateur IAM.

Les rôles IAM sont utiles pour l'accès des utilisateurs fédérés, les autorisations temporaires des utilisateurs IAM, les accès intercompte, les accès entre services et les applications exécutées sur Amazon EC2. Pour plus d'informations, consultez [Accès intercompte aux ressources dans IAM](#) dans le Guide de l'utilisateur IAM.

## Gestion de l'accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique définit les autorisations lorsqu'elles sont associées à une identité ou à une ressource. AWS évalue ces politiques lorsqu'un directeur fait une demande. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations les documents de politique JSON, consultez [Vue d'ensemble des politiques JSON](#) dans le Guide de l'utilisateur IAM.

À l'aide de politiques, les administrateurs précisent qui a accès à quoi en définissant quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Un administrateur IAM crée des politiques IAM et les ajoute aux rôles, que les utilisateurs peuvent ensuite assumer. Les politiques IAM définissent les autorisations quelle que soit la méthode que vous utilisez pour exécuter l'opération.

### Identity-based politiques

Identity-based les politiques sont des documents de politique d'autorisation JSON que vous attachez à une identité (utilisateur, groupe ou rôle). Ces politiques contrôlent les actions que peuvent exécuter ces identités, 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.

Identity-based les politiques peuvent être des politiques intégrées (intégrées directement dans une seule identité) ou des politiques gérées (politiques autonomes associées à plusieurs identités). Pour découvrir comment choisir entre des politiques gérées et en ligne, consultez [Choix entre les politiques gérées et les politiques en ligne](#) dans le Guide de l'utilisateur IAM.

## Resource-based politiques

Resource-based les politiques sont des documents de politique JSON que vous attachez à une ressource. Les exemples incluent les politiques de confiance de rôle IAM et les stratégies de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Vous devez [spécifier un principal](#) dans une politique basée sur les ressources.

Resource-based les politiques sont des politiques intégrées qui se trouvent dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

## Autres types de politique

AWS prend en charge des types de politiques supplémentaires qui peuvent définir les autorisations maximales accordées par les types de politiques les plus courants :

- Limites d'autorisations : une limite des autorisations définit le nombre maximum d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM. Pour plus d'informations, consultez [Limites d'autorisations pour des entités IAM](#) dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCP) : spécifient les autorisations maximales pour une organisation ou une unité organisationnelle dans AWS Organizations. Pour plus d'informations, consultez [Politiques de contrôle de service](#) dans le Guide de l'utilisateur AWS Organizations .
- Politiques de contrôle des ressources (RCP) : définissent les autorisations maximales disponibles pour les ressources de votre organisation. Pour plus d'informations, consultez [Politiques de contrôle des ressources \(RCP\)](#) dans le Guide de l'utilisateur AWS Organizations .
- Politiques de session : politiques avancées que vous passez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. Pour plus d'informations, consultez [Politiques de session](#) dans le Guide de l'utilisateur IAM.

## Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section [Logique d'évaluation des politiques](#) dans le guide de l'utilisateur IAM.

## Fonctionnement du registre de conteneur Amazon Elastic avec IAM

Avant d'utiliser IAM pour gérer les accès à Amazon ECR, vous devez comprendre quelles sont les fonctions IAM qui peuvent être utilisées avec Amazon ECR. Pour obtenir une vue d'ensemble de la manière dont Amazon ECR et les autres AWS services fonctionnent avec IAM, consultez la section [AWS Services That Work with IAM dans le guide de l'utilisateur IAM](#).

### Rubriques

- [Politiques Amazon ECR Identity-based](#)
- [Politiques basées sur les ressources Amazon ECR](#)
- [Autorisation basée sur les balises Amazon ECR](#)
- [Rôles IAM Amazon ECR](#)

### Politiques Amazon ECR Identity-based

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. Amazon ECR est compatible avec des actions, des ressources et des clés de condition spécifiques. Pour en savoir plus sur 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.

### Actions

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.

Les actions de politique dans Amazon ECR utilisent le préfixe suivant avant l'action : `ecr:`. Par exemple, pour accorder à une personne l'autorisation de créer un référentiel Amazon ECR à l'aide de l'opération d'API `CreateRepository` Amazon ECR, vous devez inclure l'action `ecr:CreateRepository` dans sa politique. Les déclarations de politique doivent inclure un élément `Action` ou `NotAction`. Amazon ECR définit son propre ensemble d'actions qui décrivent 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": [
 "ecr:action1",
 "ecr:action2"
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (\*). Par exemple, pour spécifier toutes les actions qui commencent par le mot Describe, incluez l'action suivante :

```
"Action": "ecr:Describe*"
```

Pour afficher la liste des actions Amazon ECR, consultez [Actions, ressources et clés de condition pour le registre de conteneur Amazon Elastic](#) dans le guide de l'utilisateur IAM.

## 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": "*"
```

Une ressource de référentiel Amazon ECR possède l'ARN suivant :

```
arn:${Partition}:ecr:${Region}:${Account}:repository/${Repository-name}
```

Pour plus d'informations sur le format des ARN, consultez [Amazon Resource Names \(ARN\) et AWS Service Namespaces](#).

Par exemple, pour spécifier le référentiel `my-repo` dans la région `us-east-1` dans votre instruction, utilisez l'ARN suivant :

```
"Resource": "arn:aws:ecr:us-east-1:123456789012:repository/my-repo"
```

Pour spécifier tous les référentiels qui appartiennent à un compte spécifique, utilisez le caractère générique (\*) :

```
"Resource": "arn:aws:ecr:us-east-1:123456789012:repository/*"
```

Pour spécifier plusieurs ressources dans une seule instruction, séparez leurs ARN par des virgules.

```
"Resource": [
 "resource1",
 "resource2"
```

Pour afficher une liste des types de ressources Amazon ECR et de leurs ARN, consultez [Ressources définies par le registre de conteneur Amazon Elastic](#) dans le guide de l'utilisateur IAM. Pour connaître les actions avec lesquelles vous pouvez spécifier l'ARN de chaque ressource, consultez [Actions définies par le registre de conteneur Amazon Elastic](#).

## Clés de condition

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.

Amazon ECR définit son propre ensemble de clés de condition et est également compatible avec 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.

La plupart des actions Amazon ECR prennent en charge les clés de condition `aws:ResourceTag` et `ecr:ResourceTag`. Pour de plus amples informations, veuillez consulter [Utilisation du contrôle Tag-Based d'accès](#).

Pour afficher la liste des clés de condition Amazon ECR, consultez [Clés de condition définies par le registre de conteneur Amazon Elastic](#) dans le guide de l'utilisateur IAM. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez [Actions définies par le registre de conteneur Amazon Elastic](#).

## Exemples

Pour voir des exemples de politiques Amazon ECR basées sur l'identité, consultez [Exemples de Identity-based politiques Amazon Elastic Container Registry](#).

## Politiques basées sur les ressources Amazon ECR

Resource-based les politiques sont des documents de politique JSON qui spécifient les actions qu'un principal spécifié peut effectuer sur une ressource Amazon ECR et dans quelles conditions. Amazon ECR prend en charge les politiques d'autorisation basées sur les ressources pour les référentiels Amazon ECR. Resource-based les politiques vous permettent d'accorder des autorisations d'utilisation à d'autres comptes pour chaque ressource. Vous pouvez également utiliser une politique basée sur une ressource pour autoriser un service AWS à accéder à vos référentiels Amazon ECR.

Pour permettre un accès comptes multiples, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que [principal dans une politique basée sur les ressources](#). L'ajout d'un principal intercompte à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource se trouvent dans des AWS comptes différents, vous devez également accorder à l'entité principale l'autorisation d'accéder à la ressource. Accordez l'autorisation en attachant une stratégie basée sur les identités à l'entité. Toutefois, si une politique basée sur les ressources accorde l'accès à un principal sur le même compte, vous n'avez pas besoin d'autorisations supplémentaires sur le référentiel Amazon ECR dans la politique basée sur l'identité. Pour plus d'informations, consultez la section En [quoi les rôles IAM diffèrent des Resource-based politiques](#) dans le Guide de l'utilisateur IAM.

Le service Amazon ECR prend en charge un seul type de politique basée sur une ressource, nommée politique de référentiel, qui est attachée à un référentiel Cette politique définit les entités principales (comptes, utilisateurs, rôles et utilisateurs fédérés) qui peuvent effectuer des actions sur le référentiel. Pour savoir comment attacher une politique basée sur les ressources à un référentiel, consultez [Politiques relatives aux référentiels privés dans Amazon ECR](#).

**Note**

Dans une politique de référentiel Amazon ECR, l'élément de stratégie `Sid` prend en charge les caractères et les espaces supplémentaires qui ne sont pas pris en charge dans les politiques IAM.

## Exemples

Pour consulter des exemples de politiques basées sur les ressources Amazon ECR, consultez [Exemples de politiques relatives aux référentiels privés dans Amazon ECR](#).

## Autorisation basée sur les balises Amazon ECR

Vous pouvez rattacher des balises aux ressources Amazon ECR ou transmettre des balises dans une demande à Amazon ECR. Pour contrôler l'accès basé sur des balises, vous devez fournir les informations de balises dans l'[élément de condition](#) d'une politique utilisant les clés de condition `ecr:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`. Pour en savoir plus sur le balisage des ressources Amazon ECR, consultez [Marquage d'un référentiel privé dans Amazon ECR](#).

Pour visualiser un exemple de politique basée sur l'identité permettant de limiter l'accès à une ressource en fonction des balises de cette ressource, consultez [Utilisation du contrôle Tag-Based d'accès](#).

## Rôles IAM Amazon ECR

Un [rôle IAM](#) est une entité de votre AWS compte qui possède des autorisations spécifiques.

### Utiliser des informations d'identification temporaires avec Amazon ECR

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 [AssumeRole](#) ou [GetFederationToken](#).

Amazon ECR est compatible avec l'utilisation des informations d'identification temporaires.

## Service-linked rôles

[Service-linked les rôles](#) permettent aux AWS services d'accéder aux ressources d'autres services pour effectuer une action en votre nom. Service-linked les rôles apparaissent dans votre compte IAM et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Amazon ECR prend en charge les rôles liés à un service. Pour de plus amples informations, veuillez consulter [Utilisation des rôles liés à un service pour Amazon ECR](#).

## Exemples de Identity-based politiques Amazon Elastic Container Registry

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier les ressources Amazon ECR. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez [Création de politiques IAM \(console\)](#) dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par Amazon ECR, y compris le format des ARN pour chacun des types de ressources, consultez [Actions, ressources et clés de condition pour Amazon Elastic Container Registry](#) dans la Référence de l'autorisation de service.

Pour savoir comment créer une politique IAM basée sur l'identité à l'aide de ces exemples de documents de politique JSON, consultez [Création de politiques dans l'onglet JSON](#) dans le Guide de l'utilisateur IAM.

### Rubriques

- [Bonnes pratiques en matière de politiques](#)
- [Utiliser la console Amazon ECR](#)
- [Autoriser les utilisateurs à afficher leurs propres autorisations](#)
- [Accéder à un seul référentiel Amazon ECR](#)

## Bonnes pratiques en matière de politiques

Identity-based les politiques déterminent si quelqu'un peut créer, accéder ou supprimer des ressources Amazon ECR 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.

## Utiliser la console Amazon ECR

Pour accéder à la console du registre de conteneur Amazon Elastic, vous devez disposer d'un jeu minimum d'autorisations. Ces autorisations doivent vous permettre de répertorier et de consulter les informations relatives aux ressources Amazon ECR de votre AWS compte. Si vous créez une politique basée sur l'identité qui est plus restrictive que l'ensemble minimum d'autorisations requis, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs ou rôles) tributaires de cette politique.

Pour garantir que ces entités peuvent toujours utiliser la console Amazon ECR, ajoutez la politique `AmazonEC2ContainerRegistryReadOnly` AWS gérée aux entités. Pour en savoir plus, consultez [Ajouter des autorisations à un utilisateur](#) dans le guide de l'utilisateur IAM.

Pour voir les autorisations de cette stratégie, consultez [AmazonElasticContainerRegistryPublicReadOnly](#) dans le AWS Guide de référence des stratégies gérées par.

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.

### Autoriser les utilisateurs à afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ViewOwnUserInfo",
 "Effect": "Allow",
 "Action": [
 "iam:GetUserPolicy",
 "iam:ListGroupsWithUser",
 "iam:ListAttachedUserPolicies",
 "iam:ListUserPolicies",
 "iam:GetUser"
],
 },
],
}
```

```
 "Resource": ["arn:aws:iam::*:user/${aws:username}"]
 },
 {
 "Sid": "NavigateInConsole",
 "Effect": "Allow",
 "Action": [
 "iam:GetGroupPolicy",
 "iam:GetPolicyVersion",
 "iam:GetPolicy",
 "iam:ListAttachedGroupPolicies",
 "iam:ListGroupPolicies",
 "iam:ListPolicyVersions",
 "iam:ListPolicies",
 "iam:ListUsers"
],
 "Resource": "*"
 }
]
}
```

## Accéder à un seul référentiel Amazon ECR

Dans cet exemple, vous souhaitez accorder à un utilisateur de votre AWS compte l'accès à l'un de vos référentiels Amazon ECR. `my-repo` Vous souhaitez également autoriser l'utilisateur à transférer, tirer et lister des images.

### JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "GetAuthorizationToken",
 "Effect": "Allow",
 "Action": [
 "ecr:GetAuthorizationToken"
],
 "Resource": "*"
 },
 {
 "Sid": "ManageRepositoryContents",
 "Effect": "Allow",

```

```
 "Action": [
 "ecr:BatchCheckLayerAvailability",
 "ecr:GetDownloadUrlForLayer",
 "ecr:GetRepositoryPolicy",
 "ecr:DescribeRepositories",
 "ecr:ListImages",
 "ecr:DescribeImages",
 "ecr:BatchGetImage",
 "ecr:InitiateLayerUpload",
 "ecr:UploadLayerPart",
 "ecr:CompleteLayerUpload",
 "ecr:PutImage"
],
 "Resource": "arn:aws:ecr:us-east-1:123456789012:repository/my-repo"
 }
]
```

## Utilisation du contrôle Tag-Based d'accès

L'action d>CreateRepositoryAPI Amazon ECR vous permet de spécifier des balises lorsque vous créez le référentiel. Pour de plus amples informations, veuillez consulter [Marquage d'un référentiel privé dans Amazon ECR](#).

Pour permettre aux utilisateurs d'attribuer des balises aux référentiels au moment de la création, ils doivent avoir les autorisations d'utiliser l'action qui crée la ressource (par exemple, `ecr:CreateRepository`). Si les balises sont spécifiées dans l'action de création de ressources, Amazon effectue une autorisation supplémentaire sur l'action `ecr:CreateRepository` pour vérifier si les utilisateurs sont autorisés à créer des balises.

Vous pouvez utiliser le contrôle d'accès basé sur des balises via des politiques IAM. Voici quelques exemples.

La politique suivante autorise uniquement un utilisateur à créer ou baliser un référentiel en tant que `key=environment, value=dev`.

### JSON

```
{
 "Version": "2012-10-17",
```

```

"Statement": [
 {
 "Sid": "AllowCreateTaggedRepository",
 "Effect": "Allow",
 "Action": [
 "ecr:CreateRepository"
],
 "Resource": "*",
 "Condition": {
 "StringEquals": {
 "aws:RequestTag/environment": "dev"
 }
 }
 },
 {
 "Sid": "AllowTagRepository",
 "Effect": "Allow",
 "Action": [
 "ecr:TagResource"
],
 "Resource": "*",
 "Condition": {
 "StringEquals": {
 "aws:RequestTag/environment": "dev"
 }
 }
 }
]
}

```

La politique suivante autoriserait un utilisateur à extraire des images de tous les référentiels, sauf si elles sont étiquetées comme `key=environment, value=prod`.

JSON

```

{
 "Version": "2012-10-17",
 "Statement": [{
 "Effect": "Allow",
 "Action": [
 "ecr:BatchGetImage",

```

```

 "ecr:GetDownloadUrlForLayer"
],
 "Resource": "*"
 },
 {
 "Effect": "Deny",
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer"
],
 "Resource": "*",
 "Condition": {
 "StringEquals": {
 "ecr:ResourceTag/environment": "prod"
 }
 }
 }
]
}

```

## AWS politiques gérées pour Amazon Elastic Container Registry

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.

Amazon ECR fournit plusieurs politiques gérées que vous pouvez associer aux identités IAM ou aux instances Amazon EC2. Ces politiques gérées permettent différents niveaux de contrôle sur l'accès aux ressources Amazon ECR et aux opérations d'API. Pour en savoir plus sur chaque opération d'API mentionnée dans ces politiques, consultez [Actions](#) dans la Référence de l'API du registre de conteneur Amazon Elastic.

## Rubriques

- [AmazonEC2ContainerRegistryFullAccess](#)
- [AmazonEC2ContainerRegistryPowerUser](#)
- [AmazonEC2ContainerRegistryPullOnly](#)
- [AmazonEC2ContainerRegistryReadOnly](#)
- [AWSECRPullThroughCache\\_ServiceRolePolicy](#)
- [ECRReplicationServiceRolePolicy](#)
- [ECRTemplateServiceRolePolicy](#)
- [Mises à jour d'Amazon ECR pour AWS stratégies gérées](#)

## AmazonEC2ContainerRegistryFullAccess

Vous pouvez associer la politique `AmazonEC2ContainerRegistryFullAccess` à vos identités IAM. Cette politique accorde un accès administratif aux ressources Amazon ECR et accorde à une identité IAM (telle qu'un utilisateur, un groupe ou un rôle) un accès aux AWS services auxquels Amazon ECR est intégré afin d'utiliser toutes les fonctionnalités d'Amazon ECR. L'utilisation de cette politique permet d'accéder à toutes les fonctionnalités d'Amazon ECR disponibles dans le AWS Management Console.

Pour voir les autorisations de cette stratégie, consultez [AmazonEC2ContainerRegistryFullAccess](#) dans le AWS Guide de référence des stratégies gérées par.

## AmazonEC2ContainerRegistryPowerUser

Vous pouvez attacher la politique `AmazonEC2ContainerRegistryPowerUser` à vos identités IAM. Cette politique accorde des autorisations d'administration qui permettent aux utilisateurs IAM de lire et d'écrire dans des référentiels, mais elle ne leur permet pas de supprimer des référentiels ni de modifier les documents de la politique qui leur sont appliqués.

Pour voir les autorisations de cette stratégie, consultez [AmazonEC2ContainerRegistryPowerUser](#) dans le AWS Guide de référence des stratégies gérées par.

## AmazonEC2ContainerRegistryPullOnly

Vous pouvez associer la politique AmazonEC2ContainerRegistryPullOnly à vos identités IAM. Cette politique autorise l'extraction d'images de conteneurs depuis Amazon ECR. Si le registre est activé pour le cache d'extraction, il autorisera également les extractions pour importer une image depuis un registre en amont.

Pour voir les autorisations de cette stratégie, consultez [AmazonEC2ContainerRegistryPullOnly](#) dans le AWS Guide de référence des stratégies gérées par.

## AmazonEC2ContainerRegistryReadOnly

Vous pouvez attacher la politique AmazonEC2ContainerRegistryReadOnly à vos identités IAM. Cette politique accorde des autorisations qui permettent un accès en lecture seule à Amazon ECR. Cela inclut la possibilité de répertorier les référentiels et les images dans les référentiels. Elle offre également la possibilité d'extraire des images depuis Amazon ECR avec la CLI Docker.

Pour voir les autorisations de cette stratégie, consultez [AmazonEC2ContainerRegistryReadOnly](#) dans le AWS Guide de référence des stratégies gérées par.

## AWSECRPullThroughCache\_ServiceRolePolicy

Vous ne pouvez pas attacher la politique IAM gérée AWSECRPullThroughCache\_ServiceRolePolicy à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à Amazon ECR de transférer des images vers vos référentiels via le flux de travail de mise en cache par extraction. Pour de plus amples informations, veuillez consulter [Rôle lié à un service Amazon ECR pour la mise en cache par extraction](#).

## ECRReplicationServiceRolePolicy

Vous ne pouvez pas attacher la politique IAM gérée ECRReplicationServiceRolePolicy à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à Amazon ECR d'effectuer des actions en votre nom. Pour de plus amples informations, veuillez consulter [Utilisation des rôles liés à un service pour Amazon ECR](#).

## ECRTemplateServiceRolePolicy

Vous ne pouvez pas attacher la politique IAM gérée ECRTemplateServiceRolePolicy à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à Amazon ECR d'effectuer des actions en votre nom. Pour de plus amples informations, veuillez consulter [Utilisation des rôles liés à un service pour Amazon ECR](#).

## Mises à jour d'Amazon ECR pour AWS stratégies gérées

Consultez les informations relatives aux mises à jour apportées aux politiques AWS gérées pour Amazon ECR depuis le moment où ce service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS sur la page de l'historique des documents Amazon ECR.

| Modifier                                                                                                                      | Description                                                                                                                                                                                                                                                                                                                                                            | Date            |
|-------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| <a href="#">Rôle lié à un service Amazon ECR pour la mise en cache par extraction</a> – Mise à jour d'une stratégie existante | Amazon ECR a ajouté de nouvelles autorisations à la politique <code>AWSECRPullThroughCache_ServiceRolePolicy</code> . Ces autorisations permettent à Amazon ECR d'extraire des images du registre privé ECR. Cela est nécessaire lorsque vous utilisez une règle de cache d'extraction pour mettre en cache des images provenant d'un autre registre privé Amazon ECR. | 12 mars 2025    |
| <a href="#">AmazonEC2ContainerRegistryPullOnly</a> : nouvelle politique                                                       | Amazon ECR a ajouté une nouvelle politique qui accorde des autorisations d'extraction uniquement à Amazon ECR.                                                                                                                                                                                                                                                         | 10 octobre 2024 |
| <a href="#">ECRTemplateServiceRolePolicy</a> : nouvelle politique                                                             | Amazon ECR a ajouté une nouvelle politique. Cette politique est associée au rôle <code>ECRTemplateServiceRolePolicy</code> lié au service pour la fonctionnalité de modèle de création de référentiel.                                                                                                                                                                 | 20 juin 2024    |

| Modifier                                                                                         | Description                                                                                                                                                                                                                                                                                                                                                                                                                      | Date             |
|--------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| <a href="#">AWSECRPullThroughCache_ServiceRolePolicy</a> — Mise à jour d'une politique existante | <p>Amazon ECR a ajouté de nouvelles autorisations à la politique <code>AWSECRPullThroughCache_ServiceRolePolicy</code>. Ces autorisations permettent à Amazon ECR de récupérer le contenu chiffré d'un secret de Secrets Manager. Cela est nécessaire lors de l'utilisation d'une règle de mise en cache par extraction pour mettre en cache des images provenant d'un registre en amont qui nécessite une authentification.</p> | 15 novembre 2023 |
| <a href="#">AWSECRPullThroughCache_ServiceRolePolicy</a> — Nouvelle politique                    | <p>Amazon ECR a ajouté une nouvelle politique. Cette politique est associée au rôle lié à un service <code>AWSServiceRoleForECRPullThroughCache</code> pour la fonction de mise en cache par extraction.</p>                                                                                                                                                                                                                     | 29 novembre 2021 |
| <a href="#">ECRReplicationServiceRolePolicy</a> : nouvelle politique                             | <p>Amazon ECR a ajouté une nouvelle politique. Cette politique est associée au rôle lié à un service <code>AWSServiceRoleForECRReplication</code> pour la fonction de réplication.</p>                                                                                                                                                                                                                                           | 4 décembre 2020  |

| Modifier                                                                                     | Description                                                                                                                                                                                                                                                  | Date             |
|----------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| <a href="#">AmazonEC2ContainerRegistryFullAccess</a> : mise à jour d'une politique existante | Amazon ECR a ajouté de nouvelles autorisations à la politique AmazonEC2ContainerRegistryFullAccess . Ces autorisations permettent aux mandataires de créer un rôle lié au service Amazon ECR.                                                                | 4 décembre 2020  |
| <a href="#">AmazonEC2ContainerRegistryReadOnly</a> : mise à jour d'une politique existante   | Amazon ECR a ajouté de nouvelles autorisations à la politique AmazonEC2ContainerRegistryReadOnly qui permettent aux mandataires de lire les politiques de cycle de vie, de répertorier les balises et de décrire les résultats de l'analyse des images.      | 10 décembre 2019 |
| <a href="#">AmazonEC2ContainerRegistryPowerUser</a> : mise à jour d'une politique existante  | Amazon ECR a ajouté de nouvelles autorisations à la politique AmazonEC2ContainerRegistryPowerUser . Elles permettent aux mandataires de lire les politiques de cycle de vie, de répertorier les balises et de décrire les résultats de l'analyse des images. | 10 décembre 2019 |

| Modifier                                                                                     | Description                                                                                                                                                                                                                                   | Date                |
|----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|
| <a href="#">AmazonEC2ContainerRegistryFullAccess</a> : mise à jour d'une politique existante | Amazon ECR a ajouté de nouvelles autorisations à la politique AmazonEC2ContainerRegistryFullAccess . Ils permettent aux directeurs de rechercher les événements de gestion ou les événements AWS CloudTrail Insights capturés par CloudTrail. | le 10 novembre 2017 |
| <a href="#">AmazonEC2ContainerRegistryReadOnly</a> : mise à jour d'une politique existante   | Amazon ECR a ajouté de nouvelles autorisations à la politique AmazonEC2ContainerRegistryReadOnly . Elles permettent aux mandataires de décrire les images Amazon ECR.                                                                         | 11 octobre 2016     |
| <a href="#">AmazonEC2ContainerRegistryPowerUser</a> : mise à jour d'une politique existante  | Amazon ECR a ajouté de nouvelles autorisations à la politique AmazonEC2ContainerRegistryPowerUser . Elles permettent aux mandataires de décrire les images Amazon ECR.                                                                        | 11 octobre 2016     |

| Modifier                                                                  | Description                                                                                                                                                                                                                                                                                                                            | Date             |
|---------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| <a href="#">AmazonEC2ContainerRegistryReadOnly</a> : nouvelle politique   | <p>Amazon ECR a ajouté une nouvelle politique qui accorde des autorisations de lecture seule à Amazon ECR. Ces autorisations offrent la possibilité de répertorier les référentiels et les images dans les référentiels. Elles offrent également la possibilité d'extraire des images depuis Amazon ECR à l'aide de la CLI Docker.</p> | 21 décembre 2015 |
| <a href="#">AmazonEC2ContainerRegistryPowerUser</a> : nouvelle politique  | <p>Amazon ECR a ajouté une nouvelle politique qui accorde des autorisations administratives permettant aux utilisateurs de lire et d'écrire dans des référentiels, mais ne leur permet pas de supprimer des référentiels ou de modifier les documents de politique qui leur sont appliqués.</p>                                        | 21 décembre 2015 |
| <a href="#">AmazonEC2ContainerRegistryFullAccess</a> : nouvelle politique | <p>Amazon ECR a ajouté une nouvelle politique. Cette politique accorde à un accès total à Amazon ECR.</p>                                                                                                                                                                                                                              | 21 décembre 2015 |
| <p>Amazon ECR a commencé à assurer le suivi des modifications</p>         | <p>Amazon ECR a commencé à suivre les modifications apportées aux politiques AWS gérées.</p>                                                                                                                                                                                                                                           | 24 juin 2021     |

## Utilisation des rôles liés à un service pour Amazon ECR

Amazon Elastic Container Registry (Amazon ECR) Gestion des identités et des accès AWS utilise des rôles [liés à un service \(IAM\)](#) pour fournir les autorisations nécessaires à l'utilisation des fonctionnalités de réplication et d'extraction du cache. Un rôle lié à un service est un type unique de rôle IAM directement lié à Amazon ECR. Le rôle lié au service est prédéfini par Amazon ECR. Il comprend toutes les autorisations dont le service a besoin pour prendre en charge les fonctions de réplication et de mise en cache par extraction pour votre registre privé. Après avoir configuré la réplication ou la mise en cache par extraction pour votre registre, un rôle lié à un service est créé automatiquement en votre nom. Pour de plus amples informations, veuillez consulter [Paramètres du registre privé dans Amazon ECR](#).

Un rôle lié à un service simplifie la configuration de la réplication et de la mise en cache par extraction avec Amazon ECR. En effet, son utilisation vous évite de devoir ajouter manuellement toutes les autorisations requises. Amazon ECR définit les autorisations de ses rôles liés à un service et, sauf indication contraire, seul Amazon ECR peut assumer ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisations. La politique d'autorisations ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer le rôle lié à un service correspondant uniquement après avoir désactivé la réplication ou la mise en cache par extraction dans votre registre. Cela vous permet de ne pas supprimer par inadvertance les autorisations dont Amazon ECR a besoin pour ces fonctions.

Pour obtenir des informations sur les autres services qui prennent en charge les rôles liés à un service, consultez [Services AWS qui fonctionnent avec IAM](#). Sur cette page liée, recherchez les services dont la valeur est Oui dans la colonne des rôles. Service-linked Choisissez un Oui ayant un lien permettant de consulter la documentation du rôle lié à un service, pour ce service.

### Rubriques

- [Régions prises en charge pour les rôles liés à un service Amazon ECR](#)
- [Rôle lié à un service Amazon ECR pour la réplication](#)
- [Rôle lié à un service Amazon ECR pour la mise en cache par extraction](#)
- [Rôle lié au service Amazon ECR pour les modèles de création de référentiels](#)

## Régions prises en charge pour les rôles liés à un service Amazon ECR

Amazon ECR prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service Amazon ECR est disponible. Pour en savoir plus sur la disponibilité de la région Amazon ECR, consultez [Régions et Points de terminaison AWS](#).

## Rôle lié à un service Amazon ECR pour la réplication

Amazon ECR utilise un rôle lié à un service nommé `AWSServiceRoleForECRReplication` qui permet à Amazon ECR de répliquer des images sur plusieurs comptes.

### Service-linked autorisations de rôle pour Amazon ECR

Le rôle `AWSServiceRoleForECRReplication` lié à un service fait confiance aux services suivants pour assumer le rôle :

- `replication.ecr.amazonaws.com`

La politique d'autorisations liée au rôle `ECRReplicationServiceRolePolicy` permet à Amazon ECR d'utiliser les actions suivantes sur les ressources :

### JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "ecr:CreateRepository",
 "ecr:ReplicateImage"
],
 "Resource": "*"
 }
]
}
```

**Note**

Le `ReplicateImage` est une API interne qu'Amazon ECR utilise pour la réplication et qui ne peut pas être appelée directement.

Vous devez configurer les autorisations de manière à autoriser une entité IAM (comme un utilisateur, un groupe ou un rôle) à créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez la section [Autorisations des Service-Linked rôles](#) dans le guide de l'utilisateur IAM.

### Création d'un rôle lié à un service pour Amazon ECR

Vous n'avez pas besoin de créer manuellement un rôle lié au service Amazon ECR. Lorsque vous configurez les paramètres de réplication pour votre registre dans le AWS Management Console, le AWS CLI, ou l' AWS API, Amazon ECR crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous devez le recréer, vous pourrez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous configurez les paramètres de réplication pour votre registre, Amazon ECR recrée automatiquement le rôle lié à un service.

### Modification d'un rôle lié à un service pour Amazon ECR

Amazon ECR n'autorise pas la modification manuelle du rôle lié au `AWSServiceRoleForECRReplication` service. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence au rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez [Modification d'un rôle lié à un service](#) dans le guide de l'utilisateur IAM.

### Suppression du rôle lié à un service pour Amazon ECR

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, aucune entité inutilisée n'est surveillée ou gérée activement. Toutefois, vous devez supprimer la configuration de réplication de votre registre dans chaque région avant de pouvoir supprimer manuellement le rôle lié à un service.

**Note**

Si vous essayez de supprimer des ressources alors que le service Amazon ECR utilise toujours les rôles, votre action de suppression pourrait échouer. Si cela se produit, attendez quelques minutes et réessayez.

Pour supprimer les ressources Amazon ECR utilisées par `AWSServiceRoleForECRReplication`

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, sélectionnez la région dans laquelle votre configuration de réplication est définie.
3. Dans le panneau de navigation, choisissez Registre de schémas.
4. Dans la page Registre privé, dans la section Configuration de réplication, choisissez Modifier.
5. Pour supprimer toutes vos règles de réplication, choisissez Tout supprimer. Cette étape nécessite une confirmation.

Pour supprimer manuellement le rôle lié au service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au `AWSServiceRoleForECRReplicationservice`. Pour plus d'informations, consultez [la section Suppression d'un Service-Linked rôle](#) dans le guide de l'utilisateur IAM.

Rôle lié à un service Amazon ECR pour la mise en cache par extraction

Amazon ECR utilise un rôle lié à un service nommé `AWSServiceRoleForECRPullThroughCache` qui autorise Amazon ECR à effectuer des actions en votre nom pour effectuer des actions d'extraction dans le cache. Pour plus d'informations sur la mise en cache par extraction, consultez [Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication](#).

Service-linked autorisations de rôle pour Amazon ECR

Le rôle `AWSServiceRoleForECRPullThroughCache` lié à un service fait confiance au service suivant pour assumer le rôle.

- `pullthroughcache.ecr.amazonaws.com`

Détails de l'autorisation

La politique d'autorisations `AWSECRPullThroughCache_ServiceRolePolicy` est attachée au rôle lié à un service. Cette politique gérée accorde à Amazon ECR l'autorisation d'effectuer les actions suivantes. Pour de plus amples informations, veuillez consulter [AWSECRPullThroughCache\\_ServiceRolePolicy](#).

- `ecr`— Permet au service Amazon ECR d'extraire et de transférer des images vers un référentiel privé.
- `secretsmanager:GetSecretValue`— Permet au service Amazon ECR de récupérer le contenu chiffré d'un AWS Secrets Manager secret. Cela est nécessaire lors de l'utilisation d'une règle de mise en cache par extraction pour mettre en cache des images provenant d'un registre en amont qui nécessite une authentification dans votre registre privé. Cette autorisation s'applique uniquement aux secrets portant le préfixe de nom `ecr-pullthroughcache/`.

La politique `AWSECRPullThroughCache_ServiceRolePolicy` contient le JSON suivant.

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ECR",
 "Effect": "Allow",
 "Action": [
 "ecr:GetAuthorizationToken",
 "ecr:BatchCheckLayerAvailability",
 "ecr:InitiateLayerUpload",
 "ecr:UploadLayerPart",
 "ecr:CompleteLayerUpload",
 "ecr:PutImage",
 "ecr:BatchGetImage",
 "ecr:BatchImportUpstreamImage",
 "ecr:GetDownloadUrlForLayer",
 "ecr:GetImageCopyStatus"
],
 "Resource": "*"
 },
 {
 "Sid": "SecretsManager",
 "Effect": "Allow",
 "Action": [
 "secretsmanager:GetSecretValue"
],
 "Resource": "arn:aws:secretsmanager:*:*:secret:ecr-pullthroughcache/"
 }
],
 "*"
}
```

```
 "Condition": {
 "StringEquals": {
 "aws:ResourceAccount": "${aws:PrincipalAccount}"
 }
 }
]
}
```

Vous devez configurer les autorisations de manière à autoriser une entité IAM (comme un utilisateur, un groupe ou un rôle) à créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez les [autorisations relatives aux Service-linked rôles](#) dans le guide de l'utilisateur IAM.

### Création d'un rôle lié à un service pour Amazon ECR

Vous n'avez pas besoin de créer manuellement le rôle lié à un service Amazon ECR pour la mise en cache par extraction. Lorsque vous créez une règle de cache d'extraction pour votre registre privé dans le AWS Management Console, le AWS CLI ou l' AWS API, Amazon ECR crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous devez le recréer, vous pourrez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez une règle de mise en cache par extraction pour votre registre privé, Amazon ECR crée à nouveau le rôle lié à un service pour vous s'il n'existe pas déjà.

### Modification d'un rôle lié à un service pour Amazon ECR

Amazon ECR n'autorise pas la modification manuelle du rôle lié au `AWSServiceRoleForECRPullThroughCacheservice`. Après la création du rôle lié à un service, vous ne pouvez pas modifier le nom du rôle car diverses entités pourraient y faire référence. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez [Modification d'un rôle lié à un service](#) dans le guide de l'utilisateur IAM.

### Suppression du rôle lié à un service pour Amazon ECR

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, aucune entité inutilisée n'est surveillée ou gérée activement. Toutefois, vous devez supprimer les règles de mise en cache par extraction pour votre registre dans chaque région avant de pouvoir supprimer manuellement le rôle lié à un service.

**Note**

Si vous essayez de supprimer des ressources alors que le service Amazon ECR utilise toujours le rôle, votre action de suppression peut échouer. Si cela se produit, attendez quelques minutes et réessayez.

Pour supprimer les ressources Amazon ECR utilisées par `AWSServiceRoleForECRPullThroughCache` rôle lié à un service

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région où vos règles de mise en cache par extraction sont créées.
3. Dans le panneau de navigation, choisissez Registre de schémas.
4. Dans la page Registre privé, dans la section Configuration de la mise en cache par extraction, choisissez Modifier.
5. Pour chaque règle de cache d'extraction que vous avez créée, sélectionnez la règle, puis choisissez Supprimer la règle.

Pour supprimer manuellement le rôle lié au service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au `AWSServiceRoleForECRPullThroughCacheservice`. Pour plus d'informations, consultez [la section Suppression d'un Service-Linked rôle](#) dans le guide de l'utilisateur IAM.

## Rôle lié au service Amazon ECR pour les modèles de création de référentiels

Amazon ECR utilise un rôle lié à un service nommé `AWSServiceRoleForECRTemplate` qui autorise Amazon ECR à effectuer des actions en votre nom afin de terminer les actions de création de modèles de référentiel.

### Service-linked autorisations de rôle pour Amazon ECR

Le rôle `AWSServiceRoleForECRTemplate` lié à un service fait confiance au service suivant pour assumer le rôle.

- `ecr.amazonaws.com`

## Détails de l'autorisation

La politique d'autorisations [ECRTemplateServiceRolePolicy](#) est attachée au rôle lié à un service. Cette politique gérée accorde à Amazon ECR l'autorisation d'effectuer des actions de création de référentiels en votre nom.

La politique ECRTemplateServiceRolePolicy contient le JSON suivant.

### JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "CreateRepositoryWithTemplate",
 "Effect": "Allow",
 "Action": [
 "ecr:CreateRepository"
],
 "Resource": "*"
 }
]
}
```

Vous devez configurer les autorisations de manière à autoriser une entité IAM (comme un utilisateur, un groupe ou un rôle) à créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez les [autorisations relatives aux Service-linked rôles](#) dans le guide de l'utilisateur IAM.

### Création d'un rôle lié à un service pour Amazon ECR

Il n'est pas nécessaire de créer manuellement le rôle lié au service Amazon ECR pour le modèle de création de référentiel. Lorsque vous créez une règle de modèle de création de référentiel pour votre registre privé dans l'AWS Management Console AWS API AWS CLI, Amazon ECR crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous devez le recréer, vous pourrez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez une règle de création de référentiel pour votre registre privé, Amazon ECR crée à nouveau le rôle lié au service pour vous s'il n'existe pas déjà.

## Modification d'un rôle lié à un service pour Amazon ECR

Amazon ECR n'autorise pas la modification manuelle du rôle lié au `AWSServiceRoleForECRTemplateservice`. Après la création du rôle lié à un service, vous ne pouvez pas modifier le nom du rôle car diverses entités pourraient y faire référence. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez [Modification d'un rôle lié à un service](#) dans le guide de l'utilisateur IAM.

## Suppression du rôle lié à un service pour Amazon ECR

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, aucune entité inutilisée n'est surveillée ou gérée activement. Toutefois, vous devez supprimer les règles de création de référentiel pour votre registre dans chaque région avant de pouvoir supprimer manuellement le rôle lié à un service.

### Note

Si vous essayez de supprimer des ressources alors que le service Amazon ECR utilise toujours le rôle, votre action de suppression peut échouer. Si cela se produit, attendez quelques minutes et réessayez.

Pour supprimer les ressources Amazon ECR utilisées par `AWSServiceRoleForECRTemplate` rôle lié à un service

1. Ouvrez la console Amazon ECR à <https://console.aws.amazon.com/ecr/> l'adresse.
2. Dans la barre de navigation, choisissez la région dans laquelle les règles de création de votre référentiel sont créées.
3. Dans le panneau de navigation, choisissez Registre de schémas.
4. Sur la page Registre privé, dans la section Modèles de création de référentiels, choisissez Modifier.
5. Pour chaque règle de création de référentiel que vous avez créée, sélectionnez la règle, puis choisissez Supprimer la règle.

Pour supprimer manuellement le rôle lié au service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au `AWSServiceRoleForECRTemplateservice`. Pour plus d'informations, consultez [la section Suppression d'un Service-Linked rôle](#) dans le guide de l'utilisateur IAM.

## Dépannage de l'identité et de l'accès au registre de conteneur Amazon Elastic

Utilisez les informations suivantes pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lorsque vous travaillez avec Amazon ECR et IAM.

### Rubriques

- [Je ne suis pas autorisé à exécuter une action dans Amazon ECR](#)
- [Je ne suis pas autorisé à effectuer iam : PassRole](#)
- [Je souhaite autoriser des personnes extérieures à mon Compte AWS pour accéder à mes ressources Amazon ECR](#)

### Je ne suis pas autorisé à exécuter une action dans Amazon ECR

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 `ecr:GetWidget` fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
ecr: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 `ecr:GetWidget`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

### Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez une erreur selon laquelle vous n'êtes pas autorisé à exécuter l'action `iam:PassRole`, vos politiques doivent être mises à jour pour vous permettre de transmettre un rôle à Amazon ECR.

Certains vos 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 exécuter une action dans Amazon ECR. 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 mon Compte AWS pour accéder à mes ressources Amazon ECR

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour plus d'informations, consultez les éléments suivants :

- Pour savoir si Amazon ECR est compatible avec ces fonctions, consultez [Fonctionnement du registre de conteneur Amazon Elastic avec IAM](#).
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section [Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez](#) dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section [Fournir un accès à des ressources Comptes AWS détenues par des tiers](#) dans le guide de l'utilisateur IAM.

- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez [Fournir un accès à des utilisateurs authentifiés en externe \(fédération d'identité\)](#) dans le Guide de l'utilisateur IAM.
- Pour en savoir plus sur la différence entre l'utilisation des rôles et des politiques basées sur les ressources pour l'accès intercompte, consultez [Accès intercompte aux ressources dans IAM](#) dans le Guide de l'utilisateur IAM.

## Protection des données dans Amazon ECR

Le modèle de [responsabilité AWS partagée \(modèle de \)](#) s'applique à la protection des données dans Amazon Elastic Container Service. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les 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 . 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 de l'activité des utilisateurs avec AWS CloudTrail. Pour plus d'informations sur l'utilisation des CloudTrail sentiers pour capturer AWS des activités, consultez la section [Utilisation des CloudTrail sentiers](#) dans le guide de AWS CloudTrail l'utilisateur.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent 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 Amazon ECS ou une autre entreprise Services AWS à l'aide de la console, de l'API ou AWS des SDK. 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.

## Rubriques

- [Chiffrement au repos](#)

## Chiffrement au repos

### Important

Dual-layer le chiffrement côté serveur avec AWS KMS (DSSE-KMS) n'est disponible que dans les AWS GovCloud (US) régions.

Amazon ECR stocke les images dans des compartiments Amazon S3 gérés par Amazon ECR. Par défaut, Amazon ECR utilise le chiffrement côté serveur avec des clés de S3-managed chiffrement Amazon qui cryptent vos données au repos à l'aide d'un algorithme de chiffrement. AES-256 Ceci ne nécessite aucune action de votre part et est fourni sans frais supplémentaires. Pour plus d'informations, consultez [la section Protection des données à l'aide du Server-Side chiffrement avec les clés de S3-Managed chiffrement Amazon \(SSE-S3\)](#) dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Pour mieux contrôler le chiffrement de vos référentiels Amazon ECR, vous pouvez utiliser le chiffrement côté serveur avec des clés KMS stockées dans (). AWS Key Management Service AWS KMS Lorsque vous chiffrez vos données, vous pouvez soit utiliser la clé par défaut Clé gérée par AWS, qui est gérée par Amazon ECR, soit spécifier votre propre clé KMS (appelée clé gérée par le client). AWS KMS Pour plus d'informations, consultez [la section Protection des données à l'aide du](#)

[Server-Side chiffrement avec des clés KMS stockées dans AWS KMS \(SSE-KMS\)](#) dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Vous pouvez choisir d'appliquer deux couches de chiffrement à vos images Amazon ECR à l'aide d'un chiffrement double couche côté serveur avec (). AWS KMS DSSE-KMS DSSE-KMS l'option est similaire à SSE-KMS, mais applique deux couches de chiffrement individuelles au lieu d'une couche. Pour plus d'informations, voir [Utilisation du chiffrement double couche côté serveur avec des AWS KMS clés](#) (). DSSE-KMS

Chaque référentiel Amazon ECR dispose d'une configuration de chiffrement, qui est définie lors de la création du référentiel. Vous pouvez utiliser des configurations de chiffrement différentes dans chaque référentiel. Pour de plus amples informations, veuillez consulter [Création d'un référentiel privé Amazon ECR pour stocker des images](#).

Lorsqu'un référentiel est créé avec AWS KMS le chiffrement activé, une clé KMS est utilisée pour chiffrer le contenu du référentiel. De plus, Amazon ECR ajoute une AWS KMS subvention à la clé KMS avec le référentiel Amazon ECR comme bénéficiaire principal.

Ce qui suit fournit des informations de haut niveau afin de comprendre la façon dont Amazon ECR est intégré à AWS KMS pour chiffrer et déchiffrer vos référentiels :

1. Lors de la création d'un référentiel, Amazon ECR envoie un [DescribeKey](#) appel AWS KMS pour valider et récupérer le nom de ressource Amazon (ARN) de la clé KMS spécifiée dans la configuration de chiffrement.
2. Amazon ECR envoie deux [CreateGrant](#) demandes pour créer des autorisations sur la clé KMS AWS KMS afin de permettre à Amazon ECR de chiffrer et de déchiffrer les données à l'aide de la clé de données.
3. Lors du transfert d'une image, une [GenerateDataKey](#) demande AWS KMS indiquant la clé KMS à utiliser pour chiffrer la couche d'image et le manifeste est envoyée.
4. AWS KMS génère une nouvelle clé de données, la chiffre sous la clé KMS spécifiée et envoie la clé de données chiffrée à stocker avec les métadonnées de la couche d'image et le manifeste d'image.
5. Lors de l'extraction d'une image, une demande de [déchiffrement](#) est envoyée à AWS KMS, spécifiant la clé de données cryptée.
6. AWS KMS déchiffre la clé de données chiffrée et envoie la clé de données déchiffrée à Amazon S3.

7. La clé de données est utilisée pour déchiffrer la couche d'image avant la couche d'image en cours d'extraction.
8. Lorsqu'un référentiel est supprimé, Amazon ECR envoie deux [RetireGrant](#) demandes AWS KMS pour annuler les subventions créées pour le référentiel.

## Considérations

Les points suivants doivent être pris en compte lors de l'utilisation du chiffrement AWS KMS basé (SSE-KMS ou DSSE-KMS) avec Amazon ECR.

- Si vous créez votre référentiel Amazon ECR avec le chiffrement KMS et que vous ne spécifiez pas de clé KMS, Amazon ECR utilise une Clé gérée par AWS avec l'alias `aws/ecr` par défaut. Cette clé KMS sera créée dans votre compte la première fois que vous créez un référentiel avec le chiffrement KMS activé.
- La configuration du chiffrement du référentiel ne peut pas être modifiée après la création d'un référentiel.
- Lorsque vous utilisez le chiffrement KMS avec votre propre clé KMS, la clé doit exister dans la même région que votre référentiel.
- Les octrois créés par Amazon ECR en votre nom ne doivent pas être révoqués. Si vous révoquez l'autorisation qui autorise Amazon ECR à utiliser les AWS KMS clés de votre compte, Amazon ECR ne pourra pas accéder à ces données, chiffrer les nouvelles images envoyées au référentiel ou les déchiffrer lorsqu'elles sont extraites. Lorsque vous révoquez un octroi pour Amazon ECR, le changement est immédiat. Pour révoquer les droits d'accès, vous devez supprimer le référentiel plutôt que de révoquer l'octroi. Lorsqu'un référentiel est supprimé, Amazon ECR retire les octrois en votre nom.
- L'utilisation des AWS KMS clés entraîne un coût. Pour en savoir plus, consultez [Pricing AWS Key Management Service](#) (Tarification).
- L'utilisation du chiffrement double couche côté serveur entraîne un coût. Pour plus d'informations, consultez les [tarifs Amazon ECR](#)

## Autorisations IAM requises

Lorsque vous créez ou supprimez un référentiel Amazon ECR avec un chiffrement côté serveur à l'aide de AWS KMS, les autorisations requises dépendent de la clé KMS spécifique que vous utilisez.

## Autorisations IAM requises lors de l'utilisation du Clé gérée par AWS pour Amazon ECR

Par défaut, lorsque AWS KMS le chiffrement est activé pour un référentiel Amazon ECR mais qu'aucune clé KMS n'est spécifiée, la clé Clé gérée par AWS pour Amazon ECR est utilisée. Lorsque la clé KMS AWS gérée pour Amazon ECR est utilisée pour chiffrer un référentiel, tout principal autorisé à créer un référentiel peut également activer le AWS KMS chiffrement du référentiel. Toutefois, le principal IAM qui supprime le référentiel doit avoir l'autorisation `kms:RetireGrant`. Cela permet de retirer les subventions qui ont été ajoutées à la AWS KMS clé lors de la création du référentiel.

L'exemple de politique IAM suivant peut être ajouté en tant que politique intégrée à un utilisateur pour s'assurer qu'il dispose des autorisations minimales nécessaires pour supprimer un référentiel dont le chiffrement est activé. La clé KMS utilisée pour chiffrer le référentiel peut être spécifiée à l'aide du paramètre de ressource.

### JSON

```
{
 "Version": "2012-10-17",
 "Id": "ecr-kms-permissions",
 "Statement": [
 {
 "Sid": "AllowAccessToRetireTheGrantsAssociatedWithTheKey",
 "Effect": "Allow",
 "Action": [
 "kms:RetireGrant"
],
 "Resource": "arn:aws:kms:us-
west-2:111122223333:key/b8d9ae76-080c-4043-92EXAMPLE"
 }
]
}
```

## Autorisations IAM requises pour l'utilisation d'une clé gérée par le client

Lors de la création d'un référentiel dont le AWS KMS chiffrement est activé à l'aide d'une clé gérée par le client, des autorisations sont requises pour la politique de clé KMS et la politique IAM pour l'utilisateur ou le rôle qui crée le référentiel.

Lorsque vous créez votre propre clé KMS, vous pouvez utiliser la politique de clé AWS KMS par défaut créée ou vous pouvez spécifier la vôtre. Pour garantir que la clé gérée par le client reste gérable par le propriétaire du compte, la politique clé relative à la clé KMS doit autoriser toutes les AWS KMS actions pour l'utilisateur root du compte. Des autorisations étendues supplémentaires peuvent être ajoutées à la politique de clé, mais au minimum l'utilisateur racine doit être autorisé à gérer la clé KMS. Pour autoriser l'utilisation de la clé KMS uniquement pour les demandes provenant d'Amazon ECR, vous pouvez utiliser la [clé de ViaService condition kms](#) : avec la valeur `ecr.<region>.amazonaws.com`.

L'exemple de politique de clé suivant donne au AWS compte (utilisateur root) propriétaire de la clé KMS un accès complet à la clé KMS. Pour plus d'informations sur cet exemple de politique clé, voir [Autoriser l'accès au AWS compte et activer les politiques IAM](#) dans le Guide du AWS Key Management Service développeur.

JSON

```
{
 "Version": "2012-10-17",
 "Id": "ecr-key-policy",
 "Statement": [
 {
 "Sid": "EnableIAMUserPermissions",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::111122223333:root"
 },
 "Action": "kms:*",
 "Resource": "*"
 }
]
}
```

L'utilisateur IAM, le rôle IAM ou le AWS compte qui crée vos référentiels doit disposer de l'`kms:DescribeKey` autorisation `kms:CreateGrant` `kms:RetireGrant`, et en plus des autorisations Amazon ECR nécessaires.

**Note**

L'autorisation `kms:RetireGrant` doit être ajoutée à la politique IAM de l'utilisateur ou du rôle créant le référentiel. Les autorisations `kms:CreateGrant` et `kms:DescribeKey` peuvent être ajoutées à la politique de clé pour la clé KMS ou à la politique IAM de l'utilisateur ou du rôle créant le référentiel. Pour plus d'informations sur le fonctionnement AWS KMS des autorisations, voir [Autorisations d'AWS KMS API : référence aux actions et aux ressources](#) dans le guide du AWS Key Management Service développeur.

L'exemple de politique IAM suivant peut être ajouté en tant que politique intégrée à un utilisateur pour s'assurer qu'il dispose des autorisations minimales nécessaires pour créer un référentiel avec le chiffrement activé et supprimer le référentiel lorsqu'il aura terminé. La AWS KMS key utilisée pour chiffrer le référentiel peut être spécifiée à l'aide du paramètre de ressource.

## JSON

```
{
 "Version": "2012-10-17",
 "Id": "ecr-kms-permissions",
 "Statement": [
 {
 "Sid":
 "AllowAccessToCreateAndRetireTheGrantsAssociatedWithTheKeyAsWellAsDescribeTheKey",
 "Effect": "Allow",
 "Action": [
 "kms:CreateGrant",
 "kms:RetireGrant",
 "kms:DescribeKey"
],
 "Resource": "arn:aws:kms:us-
west-2:111122223333:key/b8d9ae76-080c-4043-92EXAMPLE"
 }
]
}
```

## Autoriser un utilisateur à répertorier les clés KMS dans la console lors de la création d'un référentiel

Lorsque vous utilisez la console Amazon ECR pour créer un référentiel, vous pouvez octroyer des autorisations afin de permettre à un utilisateur de répertorier les clés KMS gérées par le client dans la région lors de l'activation du chiffrement pour le référentiel. L'exemple de politique IAM suivant indique les autorisations nécessaires pour répertorier vos clés KMS et alias lors de l'utilisation de la console.

### JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "kms:ListKeys",
 "kms:ListAliases",
 "kms:DescribeKey"
],
 "Resource": "*"
 }
]
}
```

## Surveillance de l'interaction d'Amazon ECR avec AWS KMS

Vous pouvez l'utiliser AWS CloudTrail pour suivre les demandes qu'Amazon ECR envoie en votre AWS KMS nom. Les entrées du CloudTrail journal contiennent une clé contextuelle de chiffrement qui les rend plus facilement identifiables.

### Contexte de chiffrement Amazon ECR

Un contexte de chiffrement est un ensemble de paires clé-valeur qui contiennent des données non secrètes arbitraires. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, lie AWS KMS cryptographiquement le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez transmettre le même contexte de chiffrement.

Dans ses requêtes [GenerateDataKey](#) et [Decrypt](#) à AWS KMS, Amazon ECR utilise un contexte de chiffrement avec deux paires nom-valeur qui identifient le référentiel et le compartiment Amazon S3 utilisés. Voici un exemple : Les noms ne varient pas, mais les valeurs de contexte de chiffrement combinées sont différentes pour chaque valeur.

```
"encryptionContext": {
 "aws:s3:arn": "arn:aws:s3::us-west-2-starport-manifest-bucket/EXAMPLE1-90ab-cdef-fedc-ba987BUCKET1/sha256:a7766145a775d39e53a713c75b6fd6d318740e70327aaa3ed5d09e0ef33fc3df",
 "aws:ecr:arn": "arn:aws:ecr:us-west-2:111122223333:repository/repository-name"
}
```

Vous pouvez utiliser le contexte de chiffrement pour identifier ces opérations cryptographiques dans les enregistrements et journaux d'audit, tels que [AWS CloudTrail](#) Amazon CloudWatch Logs, et comme condition d'autorisation dans les politiques et les autorisations.

Le contexte de chiffrement Amazon ECR se compose de deux paires nom-valeur.

- `aws:s3:arn` – La première paire nom-valeur identifie le compartiment. La clé est `aws:s3:arn`. La valeur est l'Amazon Resource Name (ARN) du compartiment Amazon S3.

```
"aws:s3:arn": "ARN of an Amazon S3 bucket"
```

Par exemple, si l'ARN du compartiment est `arn:aws:s3::us-west-2-starport-manifest-bucket/EXAMPLE1-90ab-cdef-fedc-ba987BUCKET1/sha256:a7766145a775d39e53a713c75b6fd6d318740e70327aaa3ed5d09e0ef33fc3df`, le contexte de chiffrement devra inclure la paire suivante.

```
"arn:aws:s3::us-west-2-starport-manifest-bucket/EXAMPLE1-90ab-cdef-fedc-ba987BUCKET1/sha256:a7766145a775d39e53a713c75b6fd6d318740e70327aaa3ed5d09e0ef33fc3df"
```

- `aws:ecr:arn` – La deuxième paire nom-valeur identifie l'Amazon Resource Name (ARN) du référentiel. La clé est `aws:ecr:arn`. La valeur est l'ARN du référentiel.

```
"aws:ecr:arn": "ARN of an Amazon ECR repository"
```

Par exemple, si l'ARN du référentiel est `arn:aws:ecr:us-west-2:111122223333:repository/repository-name`, le contexte de chiffrement devra inclure la paire suivante.

```
"aws:ecr:arn": "arn:aws:ecr:us-west-2:111122223333:repository/repository-name"
```

## Résolution des problèmes

Lorsque vous supprimez un référentiel Amazon ECR avec la console, si le référentiel est supprimé correctement, mais qu'Amazon ECR ne parvient pas à retirer les octrois ajoutés à votre clé KMS pour votre référentiel, vous recevrez l'erreur suivante.

```
The repository [{repository-name}] has been deleted successfully but the grants created by the kmsKey [{kms_key}] failed to be retired
```

Dans ce cas, vous pouvez retirer vous-même les AWS KMS subventions pour le dépôt.

Pour prendre sa retraite AWS KMS subventions pour un dépôt manuellement

1. Répertoriez les autorisations pour la AWS KMS clé utilisée pour le référentiel. La valeur `key-id` est incluse dans l'erreur que vous recevez de la console. Vous pouvez également utiliser la `list-keys` commande pour répertorier à la fois les clés KMS Clés gérées par AWS et les clés KMS gérées par le client dans une région spécifique de votre compte.

```
aws kms list-grants \
 --key-id b8d9ae76-080c-4043-9237-c815bfc21dfc \
 --region us-west-2
```

La sortie inclut un `EncryptionContextSubset` avec l'Amazon Resource Name (ARN) de votre référentiel. Cela peut être utilisé pour déterminer quel octroi ajouté à la clé est celui que vous souhaitez retirer. La valeur `GrantId` sera utilisée lors de la suppression de l'octroi à l'étape suivante.

2. Retirez chaque subvention pour la AWS KMS clé ajoutée au référentiel. Remplacez la valeur pour `GrantId` par l'ID de la subvention indiqué dans le résultat de l'étape précédente.

```
aws kms retire-grant \
 --key-id b8d9ae76-080c-4043-9237-c815bfc21dfc \
 --grant-id GrantId \
 --region us-west-2
```

# Validation de conformité pour le registre de conteneur Amazon Elastic

Pour savoir si un [programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité](#) spécifiques, consultez Services AWS la section de conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de [AWS conformité Programmes AWS](#) de .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir [Téléchargement de rapports dans AWS Artifact](#) .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. Pour plus d'informations sur votre responsabilité en matière de conformité lors de l'utilisation Services AWS, consultez [AWS la documentation de sécurité](#).

# Sécurité de l'infrastructure dans le registre de conteneur Amazon Elastic

En tant que service géré, Amazon Elastic Container Registry est protégé par la sécurité du réseau AWS mondial. Pour plus d'informations sur les services AWS de sécurité et sur la manière dont AWS l'infrastructure est protégée, consultez la section [Sécurité du AWS cloud](#). Pour concevoir votre AWS environnement en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section [Protection de l'infrastructure](#) dans le cadre AWS bien architecturé du pilier de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder à Amazon ECR via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Des suites de chiffrement dotées d'un secret de transmission parfait (PFS), telles que DHE (Ephemeral) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

Vous pouvez appeler ces opérations d'API à partir de n'importe quel emplacement sur le réseau, mais Amazon ECR prend en charge les politiques d'accès basées sur les ressources, ce qui peut inclure des restrictions en fonction de l'adresse IP source. Vous pouvez également utiliser des

politiques Amazon ECR pour contrôler l'accès à partir de points de terminaison Amazon Virtual Private Cloud (Amazon VPC) ou de VPC spécifiques. En fait, cela isole l'accès réseau à une ressource Amazon ECR donnée uniquement du VPC spécifique au sein du réseau. AWS Pour de plus amples informations, veuillez consulter [Points de terminaison VPC de l'interface Amazon ECR \(AWS PrivateLink\)](#).

## Points de terminaison VPC de l'interface Amazon ECR (AWS PrivateLink)

Vous pouvez améliorer le niveau de sécurité de votre VPC en configurant Amazon ECR de façon à utiliser un point de terminaison d'un VPC d'interface. Les points de terminaison VPC sont alimentés par AWS PrivateLink une technologie qui vous permet d'accéder en privé aux API Amazon ECR via des adresses IP privées (IPv4 et IPv6). AWS PrivateLink restreint tout le trafic réseau entre votre VPC et Amazon ECR vers le réseau Amazon. Vous n'avez pas besoin d'une passerelle Internet, d'un périphérique NAT ni d'une passerelle privée virtuelle.

Pour plus d'informations sur AWS PrivateLink les points de terminaison VPC, consultez la section Points de terminaison [VPC dans le guide de l'utilisateur](#) Amazon VPC.

### Considérations relatives aux points de terminaison d'un VPC Amazon ECR

Avant de configurer les points de terminaison d'un VPC pour Amazon ECR, vous devez tenir compte des considérations suivantes.

- Pour permettre à vos tâches Amazon ECS hébergées sur des instances Amazon EC2 d'extraire des images privées d'Amazon ECR, créez les points de terminaison VPC d'interface pour Amazon ECS. Pour plus d'informations, consultez [Interface VPC Endpoints \(AWS PrivateLink\)](#) dans le manuel Amazon Elastic Container Service Developer Guide.
- Les tâches Amazon ECS hébergées sur Fargate qui extrait des images de conteneur à partir d'Amazon ECR peuvent limiter l'accès au VPC spécifique utilisé par ces tâches et au point de terminaison d'un VPC utilisé par le service en ajoutant des clés de condition à au rôle IAM d'exécution de la tâche. Pour en savoir plus, consultez [Autorisations IAM facultatives pour les tâches Fargate qui extraient des images Amazon ECR sur les points de terminaison d'interface](#) dans le guide du développeur du service de conteneur Amazon Elastic.
- Le groupe de sécurité attaché au point de terminaison d'un VPC doit autoriser les connexions entrantes sur le port 443 à partir du sous-réseau privé du VPC.
- Les points de terminaison Amazon ECR VPC prennent en charge la connectivité à double pile (IPv4 et IPv6). Lorsque vous créez un point de terminaison VPC à double pile, il peut gérer le trafic sur les adresses IP privées IPv4 et IPv6.

- Les points de terminaison VPC prennent en charge les référentiels publics Amazon ECR via le point de terminaison du SDK AWS API dans l'est des États-Unis (Virginie du Nord).
- Les points de terminaison VPC ne prennent en charge que le AWS DNS fourni via Amazon Route 53. Si vous souhaitez utiliser votre propre DNS, vous pouvez utiliser le transfert DNS conditionnel. Pour en savoir plus, consultez [Jeux d'options DHCP](#) dans le guide de l'utilisateur Amazon VPC.
- Si vos conteneurs ont des connexions existantes vers Amazon S3, leurs connexions peuvent être brièvement interrompues lorsque vous ajoutez le point de terminaison de passerelle Amazon S3. Si vous souhaitez éviter cette interruption, créez un VPC qui utilise le point de terminaison de passerelle Amazon S3, puis migrez votre cluster Amazon ECS et ses conteneurs dans le nouveau VPC.
- Lorsqu'une image est extraite à l'aide d'une règle de mise en cache par extraction pour la première fois, si vous avez configuré Amazon ECR pour utiliser un point de terminaison d'un VPC d'interface à l'aide de AWS PrivateLink, vous devez créer un sous-réseau public dans le même VPC, avec une passerelle NAT, puis acheminer tout le trafic sortant vers l'Internet depuis leur sous-réseau privé vers la passerelle NAT afin que l'extraction fonctionne. Les extractions d'images suivantes ne nécessitent pas cela. Pour plus d'informations, consultez la section [Scénario : Accéder à Internet depuis un sous-réseau privé](#) dans le Guide de l'utilisateur Amazon Virtual Private Cloud.
- Pour les charges de travail nécessitant des modules cryptographiques validés par la norme FIPS 140-3, Amazon ECR prend en charge les points de terminaison FIPS pour les points de terminaison VPC.

## Considérations relatives aux images Windows

Les images basées sur le système d'exploitation Windows contiennent des artefacts dont la distribution est restreinte par la licence. Par défaut, lorsque vous poussez des images Windows vers un référentiel Amazon ECR, les couches qui contiennent ces artefacts ne sont pas poussées, car elles sont considérées comme des couches étrangères. Lorsque les artefacts sont fournis par Microsoft, les couches étrangères sont récupérées à partir de l'infrastructure Microsoft Azure. Pour cette raison, afin de permettre à vos conteneurs d'extraire ces couches étrangères d'Azure, des étapes supplémentaires sont nécessaires au-delà de la création des points de terminaison d'un VPC.

Il est possible de remplacer ce comportement lorsque vous poussez des images Windows vers Amazon ECR à l'aide de la clé `--allow-nondistributable-artifacts` dans le démon Docker. Lorsqu'il est activé, cet indicateur envoie les couches sous licence vers Amazon ECR, ce qui permet d'extraire ces images d'Amazon ECR via le point de terminaison d'un VPC, sans qu'un accès supplémentaire à Azure soit requis.

**⚠ Important**

L'utilisation de l'indicateur `--allow-nondistributable-artifacts` n'exclut pas votre obligation de respecter les termes de la licence d'image de base du conteneur Windows ; vous ne pouvez pas publier du contenu Windows pour une redistribution publique ou tierce. L'utilisation dans votre propre environnement est autorisée.

Pour activer l'utilisation de cet indicateur pour votre installation Docker, vous devez modifier le fichier de configuration du démon Docker qui, en fonction de votre installation Docker, peut généralement être configuré dans le menu des paramètres ou des préférences sous l'onglet Moteur Docker ou en modifiant directement le fichier `C:\ProgramData\docker\config\daemon.json`.

Voici un exemple de configuration de la configuration requise. Remplacez la valeur par l'URI du référentiel vers lequel vous poussez les images.

```
{
 "allow-nondistributable-artifacts": [
 "111122223333.dkr.ecr.us-west-2.amazonaws.com"
]
}
```

Après avoir modifié le fichier de configuration du démon Docker, vous devrez redémarrer le démon Docker avant de tenter de pousser votre image. Confirmez que la transmission a fonctionné en vérifiant que la couche de base a fait l'objet d'une transmission de type push vers votre référentiel.

**ℹ Note**

Les couches de base des images Windows sont volumineuses. La taille de la couche entraînera un délai de transmission plus long et des coûts de stockage supplémentaires dans Amazon ECR pour ces images. Pour ces raisons, nous vous recommandons d'utiliser cette option uniquement lorsqu'elle est strictement nécessaire pour réduire les temps de création et les coûts de stockage continus. Par exemple, la taille de l'image `mcr.microsoft.com/windows/servercore` est d'environ 1,7 Go lorsqu'elle est compressée dans Amazon ECR.

## Créer des points de terminaison d'un VPC pour Amazon ECR

Pour créer des points de terminaison d'un VPC pour le service Amazon ECR, utilisez la procédure [Créer un point de terminaison d'interface](#) que vous trouverez dans le gui de l'utilisateur Amazon VPC.

Les points de terminaison Amazon ECR VPC prennent en charge la connectivité à double pile (IPv4 et IPv6). Lorsque vous créez un point de terminaison VPC à double pile, il gère automatiquement le trafic sur les adresses IP privées IPv4 et IPv6. Le point de terminaison acheminera le trafic en utilisant la version IP appropriée en fonction de la configuration réseau de votre client et des capacités du point de terminaison.

Si vous avez des points de terminaison IPv4-only VPC existants et que vous souhaitez migrer vers des points de terminaison à double pile, vous pouvez modifier vos points de terminaison existants pour prendre en charge la connectivité à double pile ou créer de nouveaux points de terminaison à double pile. Lorsque vous créez ou modifiez des points de terminaison, assurez-vous que votre VPC et vos sous-réseaux prennent en charge la version IP que vous souhaitez utiliser. Après avoir créé des points de terminaison à double pile, ils prendront en charge à la fois les protocoles IPv4 et IPv6.

Les tâches Amazon ECS hébergées sur des instances Amazon EC2 nécessitent des points de terminaison Amazon ECR et le point de terminaison de passerelle Amazon S3.

Les tâches Amazon ECS hébergées sur Fargate et utilisant la version 1.4.0 de la plateforme ou version ultérieure nécessitent à la fois les points de terminaison d'un VPC Amazon ECR et les points de terminaison de la passerelle Amazon S3.

Les tâches Amazon ECS hébergées sur Fargate qui utilisent une version de plateforme ou une **1.3.0** version antérieure nécessitent uniquement le fichier `com.amazonaws.region.ecr.dkr` Point de terminaison VPC Amazon ECR et points de terminaison de la passerelle Amazon S3.

### Note

L'ordre dans lequel les points de terminaison sont créés n'a pas d'importance.


`com.amazonaws.region.ecr.dkr`

Ce point de terminaison est utilisé pour les API de registre Docker. Les commandes du client Docker telles que `push` et `pull` utilisent ce point de terminaison.

Lorsque vous créez ce point de terminaison, vous devez activer un nom d'hôte DNS privé. Pour ce faire, assurez-vous que l'option Activer le nom DNS privé est sélectionnée dans la console Amazon VPC lorsque vous créez le point de terminaison d'un VPC.

Pour les connexions conformes à la norme FIPS 140-3, utilisez le nom du point de terminaison FIPS `com.amazonaws.region.ecr-fips.dkr`

`com.amazonaws.region.ecr.api`

 Note

La valeur spécifiée *region* représente l'identifiant de région d'une AWS région prise en charge par Amazon ECR, par exemple `us-east-2` pour la région USA Est (Ohio).

Pour les connexions conformes à la norme FIPS 140-3, utilisez les noms des points de terminaison FIPS : `com.amazonaws.region.ecr-fips.dkr` et `com.amazonaws.region.ecr-fips.api`.

Ce point de terminaison est utilisé pour les appels à l'API Amazon ECR. Les actions d'API telles que `DescribeImages` et `CreateRepository` vont jusqu'à ce point de terminaison.

Lorsque ce point de terminaison est créé, vous avez la possibilité d'activer un nom d'hôte DNS privé. Activez ce nom d'hôte en sélectionnant Activer le nom de DNS privé dans la console VPC lorsque vous créez le point de terminaison d'un VPC. Si vous activez un nom d'hôte DNS privé pour le point de terminaison VPC, mettez à jour votre SDK AWS CLI ou optez pour la dernière version afin qu'il ne soit pas nécessaire de spécifier une URL de point de terminaison lors de l'utilisation du SDK AWS CLI ou non.

Pour les connexions conformes à la norme FIPS 140-3, utilisez le nom du point de terminaison FIPS `com.amazonaws.region.ecr-fips.api`.

Si vous activez un nom d'hôte DNS privé et que vous utilisez un SDK ou une AWS CLI version publiée avant le 24 janvier 2019, vous devez utiliser le `--endpoint-url` paramètre pour spécifier les points de terminaison de l'interface. L'exemple suivant montre le format de l'URL du point de terminaison.

```
aws ecr create-repository --repository-name name --endpoint-url https://
api.ecr.region.amazonaws.com
```

Si vous n'activez pas un nom d'hôte DNS privé pour le point de terminaison d'un VPC, vous devrez utiliser le paramètre `--endpoint-url` en spécifiant l'ID du point de terminaison de VPC pour le point de terminaison d'interface. L'exemple suivant montre le format de l'URL du point de terminaison.

```
aws ecr create-repository --repository-name name --endpoint-url
https://VPC_endpoint_ID.api.ecr.region.vpce.amazonaws.com
```

Pour les connexions conformes à la norme FIPS 140-3, utilisez l'URL du point de terminaison FIPS :

```
aws ecr create-repository --repository-name name --endpoint-url https://api.ecr-
fips.region.amazonaws.com
```

## Créer le point de terminaison de la passerelle Amazon S3

Pour que vos tâches Amazon ECS puissent extraire des images privées d'Amazon ECR, vous devez créer un point de terminaison de passerelle pour Amazon S3. Le point de terminaison de passerelle est obligatoire, car Amazon ECR utilise Amazon S3 pour stocker vos couches d'images. Lorsque vos conteneurs téléchargent des images depuis Amazon ECR, ils doivent accéder à Amazon ECR pour obtenir le manifeste d'image et à Amazon S3 pour télécharger les couches d'image réelles. Voici l'Amazon Resource Name (ARN) du compartiment Amazon S3 contenant les couches pour chaque image Docker.

```
arn:aws:s3:::prod-region-starport-layer-bucket/*
```

### Note

Si vous créez un point de terminaison VPC à double pile pour Amazon ECR, vous devez également créer un point de terminaison Amazon S3 Gateway ou Interface à double pile. Reportez-vous à [la documentation S3](#) pour plus de détails.

Utilisez la procédure [Créer un point de terminaison de passerelle](#) dans le guide de l'utilisateur Amazon VPC pour créer le point de terminaison de la passerelle Amazon S3 suivant pour Amazon ECR. Lorsque vous créez le point de terminaison, veillez à sélectionner les tables de routage pour votre VPC.

com.amazonaws. *region*.s3

Le point de terminaison de la passerelle Amazon S3 utilise un document de politique IAM pour limiter l'accès au service. Vous pouvez utiliser la politique Accès total en raison des restrictions que vous avez placées dans les rôles IAM de votre tâche, ou si d'autres politiques utilisateur IAM restent applicables au-dessus de cette politique. Si vous souhaitez limiter l'accès au compartiment Amazon S3 aux autorisations minimales requises pour utiliser Amazon ECR, consultez [Autorisations minimales relatives aux compartiments Amazon S3 pour Amazon ECR](#).

### Autorisations minimales relatives aux compartiments Amazon S3 pour Amazon ECR

Le point de terminaison de la passerelle Amazon S3 utilise un document de politique IAM pour limiter l'accès au service. Pour accorder uniquement les autorisations minimales de compartiment Amazon S3 pour Amazon ECR, limitez l'accès au compartiment Amazon S3 utilisé par Amazon ECR lorsque vous créez le document de politique IAM pour le point de terminaison.

Le tableau suivant décrit les autorisations de la politique de compartiment Amazon S3 requises par Amazon ECR.

| Autorisations                                                         | Description                                                                                                                                                                                                                                   |
|-----------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>arn:aws:s3:::prod- <i>region</i>-starport-layer-bucket/*</code> | Fournit l'accès au compartiment Amazon S3 contenant les couches pour chaque image Docker. Représente l'identifiant de région d'une région AWS prise en charge par Amazon ECR, telle que <code>us-east-2</code> pour la région USA Est (Ohio). |

### Exemple

L'exemple suivant montre comment fournir l'accès aux compartiments Amazon S3 requis pour les opérations Amazon ECR.

```
{
 "Statement": [
 {
 "Sid": "Access-to-specific-bucket-only",
 "Principal": "*",
```

```
"Action": [
 "s3:GetObject"
],
"Effect": "Allow",
"Resource": ["arn:aws:s3:::prod-region-starport-layer-bucket/*"]
}
]
}
```

## Création du point de terminaison CloudWatch Logs

Les tâches Amazon ECS utilisant le type de lancement Fargate qui utilisent un VPC sans passerelle Internet et qui utilisent également le pilote de journal pour envoyer des informations de journal à Logs nécessitent que vous créiez **awslogs** le fichier CloudWatch com.amazonaws. **region** Point de terminaison CloudWatch VPC de l'interface .logs pour les journaux. Pour plus d'informations, consultez la section [Utilisation CloudWatch des journaux avec les points de terminaison VPC de l'interface dans le guide](#) de l'utilisateur Amazon CloudWatch Logs.

## Créer une politique de point de terminaison pour vos points de terminaison d'un VPC Amazon ECR

Une politique de point de terminaison d'un VPC est une politique de ressource IAM que vous attachez à un point de terminaison lorsque vous le créez ou le modifiez. Si vous n'attachez pas de politique lorsque vous créez un point de terminaison AWS, associez une politique par défaut qui permet un accès complet au service. Une stratégie de point de terminaison n'annule pas et ne remplace pas les stratégies utilisateur ou les stratégies propres au service. Il s'agit d'une politique distincte qui contrôle l'accès depuis le point de terminaison jusqu'au service spécifié. Les politiques de point de terminaison doivent être écrites au format JSON. Pour en savoir plus, consultez [Contrôle de l'accès aux services avec des points de terminaison d'un VPC](#) dans le guide de l'utilisateur Amazon VPC.

Nous vous recommandons de créer une politique de ressource IAM unique et de l'attacher aux deux points de terminaison d'un VPC Amazon ECR.

Voici un exemple de politique de point de terminaison pour l'API Amazon ECR. Cette politique permet à un rôle IAM spécifique d'extraire des images depuis Amazon ECR.

```
{
 "Statement": [{
```

```
"Sid": "AllowPull",
"Principal": {
 "AWS": "arn:aws:iam::1234567890:role/role_name"
},
"Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer",
 "ecr:GetAuthorizationToken"
],
"Effect": "Allow",
"Resource": "*"
}]
}
```

L'exemple de politique de point de terminaison suivant empêche la suppression d'un référentiel spécifié.

```
{
 "Statement": [{
 "Sid": "AllowAll",
 "Principal": "*",
 "Action": "*",
 "Effect": "Allow",
 "Resource": "*"
 },
 {
 "Sid": "PreventDelete",
 "Principal": "*",
 "Action": "ecr:DeleteRepository",
 "Effect": "Deny",
 "Resource": "arn:aws:ecr:region:1234567890:repository/repository_name"
 }
]
}
```

L'exemple de politique de point de terminaison suivant combine les deux exemples précédents en une seule politique.

```
{
 "Statement": [{
 "Sid": "AllowAll",
 "Effect": "Allow",
```

```
"Principal": "*",
"Action": "*",
"Resource": "*"
},
{
 "Sid": "PreventDelete",
 "Effect": "Deny",
 "Principal": "*",
 "Action": "ecr:DeleteRepository",
 "Resource": "arn:aws:ecr:region:1234567890:repository/repository_name"
},
{
 "Sid": "AllowPull",
 "Effect": "Allow",
 "Principal": {
 "AWS": "arn:aws:iam::1234567890:role/role_name"
 },
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer",
 "ecr:GetAuthorizationToken"
],
 "Resource": "*"
}
]
```

Pour modifier la politique de point de terminaison d'un VPC pour Amazon ECR

1. Ouvrez la console Amazon VPC à l'adresse <https://console.aws.amazon.com/vpc/>.
2. Dans le panneau de navigation, choisissez Points de terminaison.
3. Si vous n'avez pas encore créé les points de terminaison d'un VPC pour Amazon ECR, consultez [Créer des points de terminaison d'un VPC pour Amazon ECR](#).
4. Sélectionnez le point de terminaison d'un VPC Amazon ECR auquel ajouter une politique, puis choisissez l'onglet Politique dans la partie inférieure de l'écran.
5. Choisissez Modifier la politique, puis apportez les modifications souhaitées à la politique.
6. Choisissez Enregistrer pour enregistrer la politique.

## Sous-réseaux partagés

Vous ne pouvez pas créer, décrire, modifier ou supprimer des points de terminaison d'un VPC dans des sous-réseaux qui sont partagés avec vous. Toutefois, vous pouvez utiliser les points de terminaison d'un VPC dans les sous-réseaux qui sont partagés avec vous.

## Cross-service prévention confuse des adjoints

Le problème de député confus est un problème de sécurité dans lequel une entité qui n'est pas autorisée à effectuer une action peut contraindre une entité plus privilégiée à le faire. En AWS, l'usurpation d'identité interservices peut entraîner la confusion des adjoints. Cross-service l'usurpation d'identité peut se produire lorsqu'un service (le service appelant) appelle un autre service (le service appelé). Le service appelant peut être manipulé et ses autorisations utilisées pour agir sur les ressources d'un autre client auxquelles on ne serait pas autorisé à accéder autrement. Pour éviter cela, AWS fournit des outils qui vous aident à protéger vos données pour tous les services avec des principaux de service qui ont eu accès aux ressources de votre compte.

Nous vous recommandons d'utiliser les clés de contexte de condition globale [aws:SourceArn](#) ou [aws:SourceAccount](#) dans les politiques de ressources afin de limiter les autorisations à la ressource octroyées par Amazon ECR à un autre service. Utilisez `aws:SourceArn` si vous souhaitez qu'une seule ressource soit associée à l'accès entre services. Utilisez `aws:SourceAccount` si vous souhaitez autoriser l'association d'une ressource de ce compte à l'utilisation interservices.

Le moyen le plus efficace de se protéger contre le problème de député confus consiste à utiliser la clé de contexte de condition globale `aws:SourceArn` avec l'ARN complet de la ressource. Si vous ne connaissez pas l'ARN complet de la ressource ou si vous spécifiez plusieurs ressources, utilisez la clé de contexte de condition globale `aws:SourceArn` avec des caractères génériques (\*) pour les parties inconnues de l'ARN. Par exemple, `arn:aws:service:region:123456789012:*`.

Si la valeur `aws:SourceArn` ne contient pas l'ID du compte, tel qu'un ARN de compartiment Amazon S3, vous devez utiliser les deux clés de contexte de condition globale pour limiter les autorisations.

La valeur de `aws:SourceArn` doit être `ResourceDescription`.

L'exemple suivant montre comment vous pouvez utiliser les clés de contexte de condition `aws:SourceAccount` globale `aws:SourceArn` et les clés de contexte dans une politique de

référentiel Amazon ECR pour autoriser l' AWS CodeBuild accès aux actions de l'API Amazon ECR nécessaires à l'intégration à ce service, tout en évitant le problème de confusion lié aux adjoints.

## JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "CodeBuildAccess",
 "Effect": "Allow",
 "Principal": {
 "Service": "codebuild.amazonaws.com"
 },
 "Action": [
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer"
],
 "Resource": "*",
 "Condition": {
 "ArnLike": {
 "aws:SourceArn": "arn:aws:codebuild:us-
east-1:123456789012:project/project-name"
 },
 "StringEquals": {
 "aws:SourceAccount": "123456789012"
 }
 }
 }
]
}
```

# Surveiller Amazon ECR

Vous pouvez surveiller l'utilisation de l'API Amazon ECR avec Amazon CloudWatch, qui collecte et traite les données brutes d'Amazon ECR pour en faire des indicateurs lisibles en temps quasi réel. Ces statistiques sont enregistrées pendant une période de deux semaines afin que vous puissiez accéder aux informations historiques et avoir une idée plus précise de votre utilisation des API. Les données métriques Amazon ECR sont automatiquement envoyées par intervalles CloudWatch d'une minute. Pour plus d'informations à ce sujet CloudWatch, consultez le [guide de CloudWatch l'utilisateur Amazon](#).

Amazon ECR fournit des métriques basées sur l'utilisation de votre API en ce qui concerne les actions d'autorisation, de transmission d'image et d'extraction d'image.

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon ECR et de vos AWS solutions. Nous vous recommandons de collecter des données de surveillance à partir des ressources qui constituent votre AWS solution afin de pouvoir corriger plus facilement une défaillance multipoint, le cas échéant. Avant de commencer à surveiller Amazon ECR, vous devez cependant créer un plan de surveillance qui contient les réponses aux questions suivantes :

- Quels sont les objectifs de la surveillance ?
- Quelles sont les ressources à surveiller ?
- À quelle fréquence les ressources doivent-elles être surveillées ?
- Quels outils de surveillance utiliser ?
- Qui exécute les tâches de supervision ?
- Qui doit être informé en cas de problème ?

L'étape suivante consiste à établir une référence de performances Amazon ECR normales dans votre environnement, en mesurant la performance à différents moments et dans différentes conditions de charge. Lorsque vous surveillez Amazon ECR, conservez les données de l'historique de surveillance afin de pouvoir les comparer aux nouvelles données de performances, d'identifier les modèles de performances normales et les anomalies de performances, et de concevoir des méthodes pour résoudre les problèmes.

Rubriques

- [Visualiser vos quotas de service et définir des alarmes](#)
- [Métriques d'utilisation Amazon ECR](#)
- [Rapports d'utilisation d'Amazon ECR](#)
- [Métriques de référentiel Amazon ECR](#)
- [Événements Amazon ECR et EventBridge](#)
- [Journalisation des actions Amazon ECR avec AWS CloudTrail](#)

## Visualiser vos quotas de service et définir des alarmes

Vous pouvez utiliser la CloudWatch console pour visualiser vos quotas de service et comparer votre utilisation actuelle aux quotas de service. Vous pouvez également définir des alarmes afin d'être averti lorsque vous approchez un quota.

Visualiser un quota de service et définir éventuellement une alarme

1. Ouvrez la CloudWatch console à l'adresse <https://console.aws.amazon.com/cloudwatch/>.
2. Dans le panneau de navigation, choisissez Métriques.
3. Sous l'onglet Toutes les métriques, choisissez Utilisation, puis choisissez Par ressource AWS .

La liste des métriques d'utilisation des Service Quotas s'affiche.

4. Cochez la case en regard de l'une des métriques.

Le graphique indique votre utilisation actuelle de cette AWS ressource.

5. Pour ajouter votre quota de service au graphique, procédez comme suit :
  - a. Sélectionnez l'onglet Graphed metrics (Graphiques des métriques).
  - b. Choisissez Math expression (Expression mathématique), puis Start with an empty expression (Commencer par une expression vide). Ensuite, dans la nouvelle ligne, sous Détails, saisissez **SERVICE\_QUOTA(m1)**.

Une nouvelle ligne est ajoutée au graphique avec le quota de service de la ressource représentée dans la métrique.

6. Pour afficher votre utilisation actuelle sous forme de pourcentage du quota, ajoutez une nouvelle expression ou modifiez l'expression SERVICE\_QUOTA actuelle. Pour la nouvelle expression, utilisez **m1/60/SERVICE\_QUOTA(m1)\*100**

7. (Facultatif) Pour définir une alerte qui vous avertit si vous approchez du quota de service, procédez comme suit :
  - a. Sur la ligne **m1/60/SERVICE\_QUOTA(m1)\*100**, sous Actions, choisissez l'icône d'alarme. Elle ressemble à une cloche.

La page de création d'alerte s'affiche.
  - b. Sous Conditions, vérifiez que le Threshold type (Type de seuil) est Static (Statique) et que Whenever Expression1 is (Lorsque Expression1 est) est défini sur Greater (Supérieur). Sous than (à), entrez **80**. Cela crée une alerte qui passe à l'état alerte lorsque votre utilisation dépasse 80 % du quota.
  - c. Choisissez Suivant.
  - d. Dans la page suivante, sélectionnez une rubrique Amazon SNS ou créez-en une nouvelle. Cette rubrique est notifiée lorsque l'alarme passe à l'état ALARME. Ensuite, choisissez Suivant.
  - e. Dans la page suivante, saisissez le nom et la description de l'alarme, puis choisissez Suivant.
  - f. Choisissez Créer une alarme.

## Métriques d'utilisation Amazon ECR

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

Les métriques d'utilisation d'Amazon ECR correspondent aux quotas AWS de service. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d'un quota de service. Pour en savoir plus sur les quotas de service par défaut d'Amazon ECR, consultez [Service Quotas Amazon ECR](#).

Amazon ECR publie les métriques suivantes dans l'espace de noms AWS/Usage.

| Métrique  | Description                                                                                                                  |
|-----------|------------------------------------------------------------------------------------------------------------------------------|
| CallCount | Nombre d'appels d'action d'API depuis votre compte. Les ressources sont définies par les dimensions associées à la métrique. |

| Métrique      | Description                                                                                                                                                                                                                                                                                              |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | La statistique la plus utile pour cette métrique est SUM, qui représente la somme des valeurs de tous les contributeurs pendant la période définie.                                                                                                                                                      |
| ResourceCount | <p>Le nombre de ressources spécifiées dans votre compte. Les ressources sont définies par les dimensions associées à la métrique.</p> <p>La statistique la plus utile pour cette métrique est MAXIMUM, qui représente le nombre maximum de ressources utilisées au cours de la période de 5 minutes.</p> |

Les dimensions suivantes sont utilisées pour affiner les métriques d'utilisation des API publiées par Amazon ECR.

| Dimension | Description                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Service   | Nom du AWS service contenant la ressource. Pour les métriques d'utilisation d'Amazon ECR, la valeur de cette dimension est ECR.                                                                                                                                                                                                                                                                 |
| Type      | Type d'entité faisant l'objet d'un rapport. Actuellement, la seule valeur valide pour les métriques d'utilisation de l'API Amazon ECR est API.                                                                                                                                                                                                                                                  |
| Resource  | <p>Type de ressource en cours d'exécution. Actuellement, Amazon ECR renvoie des informations sur l'utilisation de votre API pour les actions d'API suivantes.</p> <ul style="list-style-type: none"> <li>• GetAuthorizationToken</li> <li>• BatchCheckLayerAvailability</li> <li>• InitiateLayerUpload</li> <li>• UploadLayerPart</li> <li>• CompleteLayerUpload</li> <li>• PutImage</li> </ul> |

| Dimension | Description                                                                                      |
|-----------|--------------------------------------------------------------------------------------------------|
|           | <ul style="list-style-type: none"><li>• BatchGetImage</li><li>• GetDownloadUrlForLayer</li></ul> |
| Class     | Classe de ressource suivie. Actuellement, Amazon ECR n'utilise pas la dimension Class.           |

Les dimensions suivantes sont utilisées pour affiner les métriques d'utilisation des ressources publiées par Amazon ECR.

| Dimension  | Description                                                                                                                                                                                                                                                                                         |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Service    | Nom du AWS service contenant la ressource. Pour les métriques d'utilisation d'Amazon ECR, la valeur de cette dimension est ECR.                                                                                                                                                                     |
| Type       | Type d'entité faisant l'objet d'un rapport. Actuellement, la seule valeur valide pour les métriques d'utilisation des ressources Amazon ECR est RESOURCE.                                                                                                                                           |
| Resource   | Type de ressource en cours d'exécution. Actuellement, Amazon ECR renvoie des informations sur l'utilisation de vos ressources pour les métriques suivantes. <ul style="list-style-type: none"><li>• RepositoryCount</li><li>• ImagesPerRepositoryCount</li></ul>                                    |
| ResourceId | Identifiant de la ressource à l'origine de l'utilisation. Actuellement, ResourceId ne concerne que ImagesPerRepositoryCount et sa valeur est formatée comme repository/your_repository_name. Par exemple : « repository/my-repo » renvoie le nombre d'images dans le dépôt avec le nom « my-repo ». |

## Rapports d'utilisation d'Amazon ECR

AWS fournit un outil de reporting gratuit appelé Cost Explorer qui vous permet d'analyser le coût et l'utilisation de vos ressources Amazon ECR.

Utilisez Cost Explorer pour afficher les graphiques de votre utilisation et de vos coûts. Vous pouvez afficher les données des 13 mois précédents et prévoir vos dépenses pour les trois prochains mois. Vous pouvez utiliser Cost Explorer pour afficher des modèles de vos dépenses en ressources AWS au fil du temps, identifier les domaines qui méritent d'être approfondis et connaître les tendances que vous pouvez utiliser pour comprendre vos coûts. Vous pouvez également préciser des plages de temps pour les données et afficher des données temporelles par jour ou par mois.

Les données de métriques de vos rapports sur les coûts et l'utilisation illustrent l'utilisation dans l'ensemble de vos référentiels Amazon ECR. Pour de plus amples informations, veuillez consulter [Identification de vos ressources pour facturation](#).

Pour plus d'informations sur la création d'un rapport sur les AWS coûts et l'utilisation, consultez le rapport sur [les AWS coûts et l'utilisation](#) dans le guide de AWS Billing l'utilisateur.

## Métriques de référentiel Amazon ECR

Amazon ECR envoie les statistiques du nombre d'appels du référentiel à Amazon CloudWatch. Les données métriques Amazon ECR sont automatiquement envoyées par tranches CloudWatch d'une minute. Pour plus d'informations à ce sujet CloudWatch, consultez le [guide de CloudWatch l'utilisateur Amazon](#).

### Rubriques

- [CloudWatch Indicateurs habilitants](#)
- [Métriques et dimensions disponibles](#)
- [Afficher les métriques Amazon ECR à l'aide de la console CloudWatch](#)

## CloudWatch Indicateurs habilitants

Amazon ECR envoie des métriques de référentiel automatiquement pour tous les référentiels. Il n'est pas nécessaire d'effectuer des étapes manuelles.

## Métriques et dimensions disponibles

Les sections suivantes répertorient les métriques et les dimensions qu'Amazon ECR envoie à Amazon CloudWatch.

### Métriques Amazon ECR

Amazon ECR fournit des métriques pour vous permettre de contrôler vos référentiels. Vous pouvez mesurer le nombre d'extractions.

L'espace de noms AWS/ECR inclut les métriques suivantes.

#### RepositoryPullCount

Le nombre total d'extractions pour les images dans le référentiel.

Dimensions valides : RepositoryName.

Statistiques valides : Moyenne, Minimum, Maximum, Somme, Nombre d'échantillons. La statistique la plus utile est Sum (Somme).

Unité : Entier.

### Dimensions pour les métriques Amazon ECR

Les métriques Amazon ECR utilisent l'espace de noms AWS/ECR et fournissent des métriques pour les dimensions suivantes.

#### RepositoryName

Cette dimension filtre les données que vous demandez pour toutes les images de conteneur dans un référentiel spécifié.

## Afficher les métriques Amazon ECR à l'aide de la console CloudWatch

Vous pouvez consulter les métriques du référentiel Amazon ECR sur la CloudWatch console. La CloudWatch console fournit un affichage précis et personnalisable de vos ressources. Pour plus d'informations, consultez le [guide de CloudWatch l'utilisateur Amazon](#).

# Événements Amazon ECR et EventBridge

Amazon vous EventBridge permet d'automatiser vos AWS services et de répondre automatiquement aux événements du système tels que les problèmes de disponibilité des applications ou les modifications des ressources. Les événements AWS liés aux services sont diffusés EventBridge en temps quasi réel. Vous pouvez écrire des règles simples pour indiquer quels événements vous intéressent et inclure les actions automatisées à effectuer quand un événement correspond à une règle. Les actions pouvant être déclenchées automatiquement sont les suivantes :

- Ajouter des événements à des groupes de CloudWatch journaux dans Logs
- Invoquer une fonction AWS Lambda
- Appel de la fonctionnalité Exécuter la commande d'Amazon EC2
- Relais de l'événement à Amazon Kinesis Data Streams
- Activation d'une machine à AWS Step Functions états
- Notification d'une rubrique Amazon SNS ou d'une file d'attente Amazon SQS

Pour plus d'informations, consultez [Getting Started with Amazon EventBridge](#) dans le guide de EventBridge l'utilisateur Amazon.

## Exemples d'événements d'Amazon ECR

Voici des exemples d'événements à partir d'Amazon ECR. Les événements sont générés dans la mesure du possible.

Événement pour un transfert d'image terminée

L'événement suivant est envoyé lorsque chaque transfert d'image est terminé. Pour de plus amples informations, veuillez consulter [Transférer une image Docker vers un référentiel privé Amazon ECR](#).

```
{
 "version": "0",
 "id": "13cde686-328b-6117-af20-0e5566167482",
 "detail-type": "ECR Image Action",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2019-11-16T01:54:34Z",
 "region": "us-west-2",
 "resources": [],
 "detail": {
```

```

 "result": "SUCCESS",
 "repository-name": "my-repository-name",
 "image-digest":
 "sha256:7f5b2640fe6fb4f46592dfd3410c4a79dac4f89e4782432e0378abcd1234",
 "action-type": "PUSH",
 "image-tag": "latest"
 }
}

```

### Événement pour une action de mise en cache par extraction

L'événement suivant est envoyé lorsqu'une action de mise en cache par extraction est tentée. Pour de plus amples informations, veuillez consulter [Synchroniser un registre en amont avec un registre privé Amazon ECR](#).

```

{
 "version": "0",
 "id": "85fc3613-e913-7fc4-a80c-a3753e4aa9ae",
 "detail-type": "ECR Pull Through Cache Action",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2023-02-29T02:36:48Z",
 "region": "us-west-2",
 "resources": [
 "arn:aws:ecr:us-west-2:123456789012:repository/docker-hub/alpine"
],
 "detail": {
 "rule-version": "1",
 "sync-status": "SUCCESS",
 "ecr-repository-prefix": "docker-hub",
 "repository-name": "docker-hub/alpine",
 "upstream-registry-url": "public.ecr.aws",
 "image-tag": "3.17.2",
 "image-digest":
 "sha256:4aa08ef415aecc80814cb42fa41b658480779d80c77ab15EXAMPLE",
 }
}

```

### Événement pour une analyse d'image terminée (analyse de base)

Lorsque l'analyse de base est activée pour votre registre, l'événement suivant est envoyé lorsque chaque analyse d'image est terminée. Le paramètre `finding-severity-counts` ne retournera une valeur pour un niveau de gravité que s'il en existe un. Par exemple, si l'image ne contient pas

de résultats au niveau CRITICAL, aucun nombre critique ne sera renvoyé. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités du système d'exploitation dans Amazon ECR](#).

### Note

Pour plus de détails sur les événements qu'Amazon Inspector émet lorsque l'analyse améliorée est activée, consultez [EventBridge événements envoyés pour une analyse améliorée dans Amazon ECR](#).

```
{
 "version": "0",
 "id": "85fc3613-e913-7fc4-a80c-a3753e4aa9ae",
 "detail-type": "ECR Image Scan",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2019-10-29T02:36:48Z",
 "region": "us-east-1",
 "resources": [
 "arn:aws:ecr:us-east-1:123456789012:repository/my-repository-name"
],
 "detail": {
 "scan-status": "COMPLETE",
 "repository-name": "my-repository-name",
 "finding-severity-counts": {
 "CRITICAL": 10,
 "MEDIUM": 9
 },
 "image-digest":
 "sha256:7f5b2640fe6fb4f46592dfd3410c4a79dac4f89e4782432e0378abcd1234",
 "image-tags": []
 }
}
```

Événement pour une notification de changement sur une ressource dont l'analyse améliorée est activée (analyse améliorée)

Lorsque l'analyse améliorée est activée pour votre registre, l'événement suivant est envoyé par Amazon ECR lorsqu'il y a un changement avec une ressource dont l'analyse améliorée est activée. Cela inclut la création de nouveaux référentiels, la modification de la fréquence d'analyse d'un

référentiel ou la création ou la suppression d'images dans des référentiels dont l'analyse améliorée est activée. Pour de plus amples informations, veuillez consulter [Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR](#).

```
{
 "version": "0",
 "id": "0c18352a-a4d4-6853-ef53-0ab8638973bf",
 "detail-type": "ECR Scan Resource Change",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2021-10-14T20:53:46Z",
 "region": "us-east-1",
 "resources": [],
 "detail": {
 "action-type": "SCAN_FREQUENCY_CHANGE",
 "repositories": [{
 "repository-name": "repository-1",
 "repository-arn": "arn:aws:ecr:us-east-1:123456789012:repository/repository-1",
 "scan-frequency": "SCAN_ON_PUSH",
 "previous-scan-frequency": "MANUAL"
 },
 {
 "repository-name": "repository-2",
 "repository-arn": "arn:aws:ecr:us-east-1:123456789012:repository/repository-2",
 "scan-frequency": "CONTINUOUS_SCAN",
 "previous-scan-frequency": "SCAN_ON_PUSH"
 },
 {
 "repository-name": "repository-3",
 "repository-arn": "arn:aws:ecr:us-east-1:123456789012:repository/repository-3",
 "scan-frequency": "CONTINUOUS_SCAN",
 "previous-scan-frequency": "SCAN_ON_PUSH"
 }
],
 "resource-type": "REPOSITORY",
 "scan-type": "ENHANCED"
}
```

## Événement pour une suppression d'image

L'événement suivant est envoyé lorsqu'une image est supprimée. Pour de plus amples informations, veuillez consulter [Supprimer une image dans Amazon ECR](#).

```
{
 "version": "0",
 "id": "dd3b46cb-2c74-f49e-393b-28286b67279d",
 "detail-type": "ECR Image Action",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2019-11-16T02:01:05Z",
 "region": "us-west-2",
 "resources": [],
 "detail": {
 "result": "SUCCESS",
 "repository-name": "my-repository-name",
 "image-digest":
 "sha256:7f5b2640fe6fb4f46592dfd3410c4a79dac4f89e4782432e0378abcd1234",
 "action-type": "DELETE",
 "image-tag": "latest"
 }
}
```

## Événement pour une action d'archivage d'images

L'événement suivant est envoyé lorsqu'une image est archivée. Le `target-storage-class` champ sera défini sur `ARCHIVE`. L'événement inclut les types de média manifeste et artefact afin d'identifier le type de contenu archivé.

```
{
 "version": "0",
 "id": "4f5ec4d5-4de4-7aad-a046-EXAMPLE",
 "detail-type": "ECR Image Action",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2019-08-06T00:58:09Z",
 "region": "us-east-1",
 "resources": [],
 "detail": {
 "action-type": "UPDATE_STORAGE_CLASS",
 "target-storage-class": "ARCHIVE",
 "image-digest":
 "sha256:f98d67af8e53a536502bfc600de3266556b06ed635a32d60aa7a5fe6d7e609d7",
 "repository-name": "ubuntu",
 "result": "SUCCESS",
 "manifest-media-type": "application/vnd.oci.image.manifest.v1+json",
 "artifact-media-type": "application/vnd.oci.image.config.v1+json"
 }
}
```

```
}
}
```

### Événement relatif à une action de restauration d'image

L'événement suivant est envoyé lorsqu'une image archivée est restaurée. Le `target-storage-class` champ sera défini sur `STANDARD`. L'événement inclut un `last-activated-at` champ indiquant la date de dernière restauration de l'image.

```
{
 "version": "0",
 "id": "7b8fc5e6-5ef5-8bbe-b157-EXAMPLE",
 "detail-type": "ECR Image Action",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2019-08-06T01:15:22Z",
 "region": "us-east-1",
 "resources": [],
 "detail": {
 "action-type": "UPDATE_STORAGE_CLASS",
 "target-storage-class": "STANDARD",
 "image-digest":
 "sha256:f98d67af8e53a536502bfc600de3266556b06ed635a32d60aa7a5fe6d7e609d7",
 "repository-name": "ubuntu",
 "result": "SUCCESS",
 "manifest-media-type": "application/vnd.oci.image.manifest.v1+json",
 "artifact-media-type": "application/vnd.oci.image.config.v1+json",
 "last-activated-at": "2025-10-10T19:13:02.74Z"
 }
}
```

### Événement pour une action de restauration du référent

L'événement suivant est envoyé lorsqu'un référent archivé (artefact de référence tel qu'un SBOM, une signature ou une attestation) est restauré. Notez que `detail-type` c'est `ECR Referrer Action` pour le distinguer des actions classiques sur les images. Les `artifact-media-type` champs `manifest-media-type` et identifient le type spécifique de référent à restaurer. Dans cet exemple, un artefact SBOM est en cours de restauration.

```
{
 "version": "0",
```

```

 "id": "8c9gd6f7-6fg6-9ccf-c268-EXAMPLE",
 "detail-type": "ECR Referrer Action",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2019-08-06T01:20:45Z",
 "region": "us-east-1",
 "resources": [],
 "detail": {
 "action-type": "UPDATE_STORAGE_CLASS",
 "target-storage-class": "STANDARD",
 "image-digest":
"sha256:f98d67af8e53a536502bfc600de3266556b06ed635a32d60aa7a5fe6d7e609d7",
 "repository-name": "sbom",
 "result": "SUCCESS",
 "manifest-media-type": "application/vnd.cncf.oras.artifact.manifest.v1+json",
 "artifact-media-type": "text/sbom+json",
 "last-activated-at": "2025-10-10T19:13:02.74Z"
 }
 }
}

```

## Événement relatif à une réplication d'image terminée

L'événement suivant est envoyé lorsque chaque réplication d'image est terminée. Pour de plus amples informations, veuillez consulter [Réplication d'images privées sur Amazon ECR](#).

```

{
 "version": "0",
 "id": "c8b133b1-6029-ee73-e2a1-4f466b8ba999",
 "detail-type": "ECR Replication Action",
 "source": "aws.ecr",
 "account": "123456789012",
 "time": "2024-05-08T20:44:54Z",
 "region": "us-east-1",
 "resources": [
 "arn:aws:ecr:us-east-1:123456789012:repository/docker-hub/alpine"
],
 "detail": {
 "result": "SUCCESS",
 "repository-name": "docker-hub/alpine",
 "image-digest":
"sha256:7f5b2640fe6fb4f46592dfd3410c4a79dac4f89e4782432e0378abcd1234",
 "source-account": "123456789012",
 "action-type": "REPLICATE",

```

```
"source-region": "us-west-2",
"image-tag": "3.17.2"
}
}
```

## Journalisation des actions Amazon ECR avec AWS CloudTrail

Amazon ECR est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Amazon ECR. CloudTrail capture les actions Amazon ECR suivantes sous forme d'événements :

- Tous les appels d'API, notamment les appels à partir de la console Amazon ECR
- Toutes les actions effectuées en raison des paramètres de chiffrement sur vos référentiels
- Toutes les actions entreprises en vertu des règles de politique de cycle de vie, qu'elles réussissent ou qu'elles échouent

### Important

En raison des limites de taille des CloudTrail événements individuels, pour les actions politiques relatives au cycle de vie impliquant l'expiration de 10 images ou plus, Amazon ECR envoie plusieurs événements à CloudTrail. En outre, Amazon ECR inclut un maximum de 100 identifications par image.

Lorsqu'un suivi est créé, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour Amazon ECR. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. En utilisant ces informations, vous pouvez déterminer la demande qui a été envoyée à Amazon ECR, l'adresse IP source, qui a effectué la demande, quand, ainsi que d'autres informations.

Pour plus d'informations, consultez le [Guide de l'utilisateur AWS CloudTrail](#).

## Informations Amazon ECR dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dans Amazon ECR, cette activité est enregistrée dans un CloudTrail événement avec d'autres

événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section [Affichage des événements à l'aide de l'historique des CloudTrail événements](#).

Pour un enregistrement continu des événements de votre AWS compte, y compris des événements pour Amazon ECR, créez un historique. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Si vous créez un journal d'activité dans la console, vous pourrez l'appliquer à une seule région ou à toutes les régions. Le journal enregistre les événements dans 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 les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour en savoir plus, consultez :

- [Création d'un parcours pour votre AWS compte](#)
- [AWS intégrations de services avec journaux CloudTrail](#)
- [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 de l'API Amazon ECR sont enregistrées CloudTrail et documentées dans le manuel [Amazon Elastic Container Registry API Reference](#). Lorsque vous effectuez des tâches courantes, des sections sont générées dans les fichiers CloudTrail journaux pour chaque action d'API faisant partie de cette tâche. Par exemple, lorsque vous créez un référentiel `GetAuthorizationToken`, `CreateRepository` et que `SetRepositoryPolicy` des sections sont générées dans les fichiers CloudTrail journaux. Lorsque vous transférez une image vers un référentiel, `InitiateLayerUpload`, `UploadLayerPart`, `CompleteLayerUpload`, et `PutImage`, si le montage par blob est activé, des `MountLayer` sections sont générées. Lorsque vous extrayez une image, les sections `GetDownloadUrlForLayer` et `BatchGetImage` sont générées. Lorsque vous archivez ou restaurez, une `UpdateImageStorageClass` section d'image est générée. Lorsque OCI les clients qui prennent en charge la OCI 1.1 spécification récupèrent la liste des référents, ou artefacts de référence, pour une image à l'aide de l'API `Referrers`, un événement est émis. `ListImageReferrers` CloudTrail Pour obtenir des exemples de ces tâches courantes, consultez [CloudTrail exemples de saisie de journal](#).

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 effectuée avec les informations d'identification utilisateur racine ou

- Si la demande a été effectuée avec des informations d'identification de sécurité temporaires pour un rôle ou un utilisateur fédéré
- Si la demande a été faite par un autre AWS service

Pour plus d'informations, consultez l'[élément userIdentity CloudTrail](#).

## Présentation des entrées des fichiers journaux Amazon ECR

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande et d'autres informations. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

### CloudTrail exemples de saisie de journal

Vous trouverez ci-dessous des exemples de saisie de CloudTrail journal pour quelques tâches Amazon ECR courantes.

Ces exemples ont été mis en forme pour faciliter la lecture. Dans un fichier CloudTrail journal, toutes les entrées et tous les événements sont concaténés sur une seule ligne. En outre, cet exemple se limite à une seule entrée Amazon ECR. Dans un véritable fichier CloudTrail journal, vous pouvez voir les entrées et les événements de plusieurs AWS services.

#### Important

La SourceIPAddress est l'adresse IP à partir de laquelle la demande a été effectuée. Pour les actions qui proviennent de la console de service, l'adresse indiquée est celle de votre ressource sous-jacente, et non celle du serveur Web de la console. Pour les services en AWS entrée, seul le nom DNS est affiché. Nous évaluons toujours l'authentification avec l'adresse IP source du client, même si elle est expurgée sous le nom DNS du AWS service.

### Rubriques

- [Exemple : Créer une action de référentiel](#)
- [Exemple : AWS KMS CreateGrantAction d'API lors de la création d'un référentiel Amazon ECR](#)

- [Exemple : Action de transmission d'image](#)
- [Exemple : Action d'extraction d'image](#)
- [Exemple : Action de politique de cycle de vie d'image](#)
- [Exemple : action d'archivage d'images](#)
- [Exemple : action de restauration d'image](#)
- [Exemple : action des référents d'images](#)

Exemple : Créer une action de référentiel

L'exemple suivant montre une entrée de CloudTrail journal illustrant l'CreateRepositoryaction.

```
{
 "eventVersion": "1.04",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
 "arn": "arn:aws:sts::123456789012:user/Mary_Major",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "sessionContext": {
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2018-07-11T21:54:07Z"
 },
 "sessionIssuer": {
 "type": "Role",
 "principalId": "AIDACKCEVSQ6C2EXAMPLE",
 "arn": "arn:aws:iam::123456789012:role/Admin",
 "accountId": "123456789012",
 "userName": "Admin"
 }
 }
 },
 "eventTime": "2018-07-11T22:17:43Z",
 "eventSource": "ecr.amazonaws.com",
 "eventName": "CreateRepository",
 "awsRegion": "us-east-2",
 "sourceIPAddress": "203.0.113.12",
 "userAgent": "console.amazonaws.com",
 "requestParameters": {
 "repositoryName": "testrepo"
 }
}
```

```

 },
 "responseElements": {
 "repository": {
 "repositoryArn": "arn:aws:ecr:us-east-2:123456789012:repository/testrepo",
 "repositoryName": "testrepo",
 "repositoryUri": "123456789012.dkr.ecr.us-east-2.amazonaws.com/testrepo",
 "createdAt": "Jul 11, 2018 10:17:44 PM",
 "registryId": "123456789012"
 }
 }
 },
 "requestID": "cb8c167e-EXAMPLE",
 "eventID": "e3c6f4ce-EXAMPLE",
 "resources": [
 {
 "ARN": "arn:aws:ecr:us-east-2:123456789012:repository/testrepo",
 "accountId": "123456789012"
 }
],
 "eventType": "AwsApiCall",
 "recipientAccountId": "123456789012"
}

```

Exemple : AWS KMS **CreateGrantAction** d'API lors de la création d'un référentiel Amazon ECR

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l' AWS KMS **CreateGrantaction** à effectuer lors de la création d'un référentiel Amazon ECR avec le chiffrement KMS activé. Pour chaque dépôt créé avec le chiffrement KMS activé, vous devriez voir apparaître deux entrées de **CreateGrant** journal CloudTrail.

```

{
 "eventVersion": "1.05",
 "userIdentity": {
 "type": "IAMUser",
 "principalId": "AIDAIEP6W46J43IG7LXAQ",
 "arn": "arn:aws:iam::123456789012:user/Mary_Major",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "userName": "Mary_Major",
 "sessionContext": {
 "sessionIssuer": {
 },
 "webIdFederationData": {

```

```
 },
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2020-06-10T19:22:10Z"
 }
 },
 "invokedBy": "AWS Internal"
},
"eventTime": "2020-06-10T19:22:10Z",
"eventSource": "kms.amazonaws.com",
"eventName": "CreateGrant",
"awsRegion": "us-west-2",
"sourceIPAddress": "203.0.113.12",
"userAgent": "console.amazonaws.com",
"requestParameters": {
 "keyId": "4b55e5bf-39c8-41ad-b589-18464af7758a",
 "granteePrincipal": "ecr.us-west-2.amazonaws.com",
 "operations": [
 "GenerateDataKey",
 "Decrypt"
],
 "retiringPrincipal": "ecr.us-west-2.amazonaws.com",
 "constraints": {
 "encryptionContextSubset": {
 "aws:ecr:arn": "arn:aws:ecr:us-west-2:123456789012:repository/testrepo"
 }
 }
},
"responseElements": {
 "grantId": "3636af9adfee1accb67b83941087dcd45e7fadc4e74ff0103bb338422b5055f3"
},
"requestID": "047b7dea-b56b-4013-87e9-a089f0f6602b",
"eventID": "af4c9573-c56a-4886-baca-a77526544469",
"readOnly": false,
"resources": [
 {
 "accountId": "123456789012",
 "type": "AWS::KMS::Key",
 "ARN": "arn:aws:kms:us-west-2:123456789012:key/4b55e5bf-39c8-41ad-
b589-18464af7758a"
 }
],
"eventType": "AwsApiCall",
```

```
"recipientAccountId": "123456789012"
}
```

### Exemple : Action de transmission d'image

L'exemple suivant montre une entrée de CloudTrail journal qui illustre une image push qui utilise l'PutImageaction.

#### Note

Lorsque vous insérez une image, vous verrez `InitiateLayerUpload` également des `CompleteLayerUpload` références et des références dans les CloudTrail journaux. `UploadLayerPart`

```
{
 "eventVersion": "1.04",
 "userIdentity": {
 "type": "IAMUser",
 "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
 "arn": "arn:aws:sts::123456789012:user/Mary_Major",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "userName": "Mary_Major",
 "sessionContext": {
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2019-04-15T16:42:14Z"
 }
 }
 },
 "eventTime": "2019-04-15T16:45:00Z",
 "eventSource": "ecr.amazonaws.com",
 "eventName": "PutImage",
 "awsRegion": "us-east-2",
 "sourceIPAddress": "AWS Internal",
 "userAgent": "AWS Internal",
 "requestParameters": {
 "repositoryName": "testrepo",
 "imageTag": "latest",
 "registryId": "123456789012",
 }
}
```

```

"imageManifest": "{\n \"schemaVersion\": 2,\n \"mediaType\": \"application/
vnd.docker.distribution.manifest.v2+json\",\n \"config\": {\n \"mediaType\":
\"application/vnd.docker.container.image.v1+json\",\n \"size\": 5543,\n
\"digest\": \"sha256:000b9b805af1cdb60628898c9f411996301a1c13afd3dbef1d8a16ac6dbf503a
\"\n },\n \"layers\": [\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 43252507,\n
\"digest\": \"sha256:3b37166ec61459e76e33282dda08f2a9cd698ca7e3d6bc44e6a6e7580cdeff8e
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 846,\n \"digest
\": \"sha256:504facff238fde83f1ca8f9f54520b4219c5b8f80be9616ddc52d31448a044bd
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 615,\n \"digest
\": \"sha256:ebbcacd28e101968415b0c812b2d2dc60f969e36b0b08c073bf796e12b1bb449\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 850,\n \"digest
\": \"sha256:c7fb3351ecad291a88b92b600037e2435c84a347683d540042086fe72c902b8a
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 168,\n \"digest\":
\"sha256:2e3debadcbf7e542e2aefbce1b64a358b1931fb403b3e4aeca27cb4d809d56c2\"\n },
\n {\n \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip
\",\n \"size\": 37720774,\n \"digest\":
\"sha256:f8c9f51ad524d8ae9bf4db69cd3e720ba92373ec265f5c390ffb21bb0c277941\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 30432107,\n
\"digest\": \"sha256:813a50b13f61cf1f8d25f19fa96ad3aa5b552896c83e86ce413b48b091d7f01b
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 197,\n \"digest
\": \"sha256:7ab043301a6187ea3293d80b30ba06c7bf1a0c3cd4c43d10353b31bc0cecf7d
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 154,\n \"digest
\": \"sha256:67012cca8f31dc3b8ee2305e7762fee20c250513effdedb38a1c37784a5a2e71\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 176,\n \"digest
\": \"sha256:3bc892145603fffc9b1c97c94e2985b4cb19ca508750b15845a5d97becbd1a0e
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 183,\n \"digest
\": \"sha256:6f1c79518f18251d35977e7e46bfa6c6b9cf50df2a79d4194941d95c54258d18\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 212,\n \"digest
\": \"sha256:b7bcfbc2e2888afebede4dd1cd5eebf029bb6315feeaf0b56e425e11a50afe42\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 212,\n \"digest\":
\"sha256:2b220f8b0f32b7c2ed8eaafe1c802633bbd94849b9ab73926f0ba46cdae91629\"\n
 }]\n}"

```

```

},
"responseElements": {
 "image": {
 "repositoryName": "testrepo",
 "imageManifest": "{\n \"schemaVersion\": 2,\n \"mediaType\": \"application/
vnd.docker.distribution.manifest.v2+json\",\n \"config\": {\n \"mediaType\":
\"application/vnd.docker.container.image.v1+json\",\n \"size\": 5543,\n
 \"digest\": \"sha256:000b9b805af1cdb60628898c9f411996301a1c13afd3dbef1d8a16ac6dbf503a
\"\n },\n \"layers\": [\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 43252507,\n
 \"digest\": \"sha256:3b37166ec61459e76e33282dda08f2a9cd698ca7e3d6bc44e6a6e7580cdeff8e
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 846,\n \"digest
\": \"sha256:504facff238fde83f1ca8f9f54520b4219c5b8f80be9616ddc52d31448a044bd
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 615,\n \"digest
\": \"sha256:ebbcacd28e101968415b0c812b2d2dc60f969e36b0b08c073bf796e12b1bb449\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 850,\n \"digest
\": \"sha256:c7fb3351ecad291a88b92b600037e2435c84a347683d540042086fe72c902b8a
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 168,\n \"digest\":
\"sha256:2e3debadcbf7e542e2aefbce1b64a358b1931fb403b3e4aeca27cb4d809d56c2\"\n },
\n {\n \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip
\",\n \"size\": 37720774,\n \"digest\":
\"sha256:f8c9f51ad524d8ae9bf4db69cd3e720ba92373ec265f5c390ffb21bb0c277941\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 30432107,\n
 \"digest\": \"sha256:813a50b13f61cf1f8d25f19fa96ad3aa5b552896c83e86ce413b48b091d7f01b
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 197,\n \"digest
\": \"sha256:7ab043301a6187ea3293d80b30ba06c7bf1a0c3cd4c43d10353b31bc0cecf7d
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 154,\n \"digest
\": \"sha256:67012cca8f31dc3b8ee2305e7762fee20c250513effdedb38a1c37784a5a2e71\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 176,\n \"digest
\": \"sha256:3bc892145603fffc9b1c97c94e2985b4cb19ca508750b15845a5d97becbd1a0e
\"\n },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 183,\n \"digest
\": \"sha256:6f1c79518f18251d35977e7e46bfa6c6b9cf50df2a79d4194941d95c54258d18\"\n
 },\n {\n \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n \"size\": 212,\n \"digest
\": \"sha256:b7bcfbc2e2888afebede4dd1cd5eebf029bb6315feeaf0b56e425e11a50afe42\"\n
 }
}

```

```

 },\n {\n \"mediaType\": \"application/\n vnd.docker.image.rootfs.diff.tar.gzip\",,\n \"size\": 212,\n \"digest\":\n \"sha256:2b220f8b0f32b7c2ed8eaafe1c802633bbd94849b9ab73926f0ba46cdae91629\",,\n]\n },\n \"registryId\": \"123456789012\",,\n \"imageId\": {\n \"imageDigest\":\n \"sha256:98c8b060c21d9adbb6b8c41b916e95e6307102786973ab93a41e8b86d1fc6d3e\",,\n \"imageTag\": \"latest\"\n }\n },\n \"requestID\": \"cf044b7d-5f9d-11e9-9b2a-95983139cc57\",,\n \"eventID\": \"2bfd4ee2-2178-4a82-a27d-b12939923f0f\",,\n \"resources\": [{\n \"ARN\": \"arn:aws:ecr:us-east-2:123456789012:repository/testrepo\",,\n \"accountId\": \"123456789012\"\n }],\n \"eventType\": \"AwsApiCall\",,\n \"recipientAccountId\": \"123456789012\"\n }\n}

```

### Exemple : Action d'extraction d'image

L'exemple suivant montre une entrée de CloudTrail journal qui illustre une extraction d'image utilisant l'BatchGetImageaction.

#### Note

Lors de l'extraction d'une image, si vous ne disposez pas déjà de l'image en local, des références `GetDownloadUrlForLayer` apparaîtront également dans les journaux CloudTrail .

```

{
 \"eventVersion\": \"1.04\",
 \"userIdentity\": {
 \"type\": \"IAMUser\",
 \"principalId\": \"AIDACKCEVSQ6C2EXAMPLE:account_name\",
 \"arn\": \"arn:aws:sts::123456789012:user/Mary_Major\",
 \"accountId\": \"123456789012\",
 \"accessKeyId\": \"AKIAIOSFODNN7EXAMPLE\",

```

```
"userName": "Mary_Major",
"sessionContext": {
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2019-04-15T16:42:14Z"
 }
},
"eventTime": "2019-04-15T17:23:20Z",
"eventSource": "ecr.amazonaws.com",
"eventName": "BatchGetImage",
"awsRegion": "us-east-2",
"sourceIPAddress": "ecr.amazonaws.com",
"userAgent": "ecr.amazonaws.com",
"requestParameters": {
 "imageIds": [{
 "imageTag": "latest"
 }],
 "acceptedMediaTypes": [
 "application/json",
 "application/vnd.oci.image.manifest.v1+json",
 "application/vnd.oci.image.index.v1+json",
 "application/vnd.docker.distribution.manifest.v2+json",
 "application/vnd.docker.distribution.manifest.list.v2+json",
 "application/vnd.docker.distribution.manifest.v1+prettyjws"
],
 "repositoryName": "testrepo",
 "registryId": "123456789012"
},
"responseElements": null,
"requestID": "2a1b97ee-5fa3-11e9-a8cd-cd2391aeda93",
"eventID": "c84f5880-c2f9-4585-9757-28fa5c1065df",
"resources": [{
 "ARN": "arn:aws:ecr:us-east-2:123456789012:repository/testrepo",
 "accountId": "123456789012"
}],
"eventType": "AwsApiCall",
"recipientAccountId": "123456789012"
}
```

## Exemple : Action de politique de cycle de vie d'image

L'exemple suivant montre une entrée de CloudTrail journal qui montre quand une image a expiré en raison d'une règle de politique de cycle de vie. Ce type d'événement peut être localisé en filtrant le `PolicyExecutionEvent` pour le champ du nom d'événement.

Lorsque vous testez un aperçu d'une politique de cycle de vie, Amazon ECR génère une entrée de CloudTrail journal avec le champ du nom de l'événement de `DryRunEvent`, avec exactement la même structure que le `PolicyExecutionEvent`. En modifiant le nom de l'événement en `DryRunEvent`, vous pouvez plutôt filtrer les événements de course à sec.

### Important

En raison des limites de taille des CloudTrail événements individuels, pour les actions politiques relatives au cycle de vie impliquant l'expiration de 10 images ou plus, Amazon ECR envoie plusieurs événements à CloudTrail. En outre, Amazon ECR inclut un maximum de 100 identifications par image.

```
{
 "eventVersion": "1.05",
 "userIdentity": {
 "accountId": "123456789012",
 "invokedBy": "AWS Internal"
 },
 "eventTime": "2020-03-12T20:22:12Z",
 "eventSource": "ecr.amazonaws.com",
 "eventName": "PolicyExecutionEvent",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "AWS Internal",
 "userAgent": "AWS Internal",
 "requestParameters": null,
 "responseElements": null,
 "eventID": "9354dd7f-9aac-4e9d-956d-12561a4923aa",
 "readOnly": true,
 "resources": [
 {
 "ARN": "arn:aws:ecr:us-west-2:123456789012:repository/testrepo",
 "accountId": "123456789012",
 "type": "AWS::ECR::Repository"
 }
]
}
```

```

],
 "eventType": "AwsServiceEvent",
 "recipientAccountId": "123456789012",
 "serviceEventDetails": {
 "repositoryName": "testrepo",
 "lifecycleEventPolicy": {
 "lifecycleEventRules": [
 {
 "rulePriority": 1,
 "description": "remove all images > 2",
 "lifecycleEventSelection": {
 "tagStatus": "Any",
 "tagPrefixList": [],
 "countType": "Image count more than",
 "countNumber": 2
 },
 "action": "expire"
 }
],
 "lastEvaluatedAt": 0,
 "policyVersion": 1,
 "policyId": "ceb86829-58e7-9498-920c-aa042e33037b"
 },
 "lifecycleEventImageActions": [
 {
 "lifecycleEventImage": {
 "digest":
"sha256:ddba4d27a7ffc3f86dd6c2f92041af252a1f23a8e742c90e6e1297bfa1bc0c45",
 "tagStatus": "Tagged",
 "tagList": [
 "alpine"
],
 "pushedAt": 1584042813000
 },
 "rulePriority": 1
 },
 {
 "lifecycleEventImage": {
 "digest":
"sha256:6ab380c5a5acf71c1b6660d645d2cd79cc8ce91b38e0352cbf9561e050427baf",
 "tagStatus": "Tagged",
 "tagList": [
 "centos"
],
 },
 }
]
 }
],
 "lastEvaluatedAt": 0,
 "policyVersion": 1,
 "policyId": "ceb86829-58e7-9498-920c-aa042e33037b"
},
{lifecycleEventImageActions: [
 {
 lifecycleEventImage: {
 digest:
"sha256:ddba4d27a7ffc3f86dd6c2f92041af252a1f23a8e742c90e6e1297bfa1bc0c45",
 tagStatus: "Tagged",
 tagList: [
 "alpine"
],
 pushedAt: 1584042813000
 },
 rulePriority: 1
 },
 {
 lifecycleEventImage: {
 digest:
"sha256:6ab380c5a5acf71c1b6660d645d2cd79cc8ce91b38e0352cbf9561e050427baf",
 tagStatus: "Tagged",
 tagList: [
 "centos"
],
 },
 }
]}

```

```

 "pushedAt": 1584042842000
 },
 "rulePriority": 1
 }
],
 "lifecycleEventFailureDetails": [
 {
 "lifecycleEventImage": {
 "digest":
"sha256:9117e1bc28cd20751e584b4ccd19b1178d14cf02d134b04ce6be0cc51bfff762a",
 "tagStatus": "Untagged",
 "tagList": [],
 "pushedAt": 1584042844000
 },
 "rulePriority": 1,
 "failureCode": "ImageReferencedByManifestList",
 "failureReason": "Requested image referenced by manifest list:
[sha256:4b27c83d44a18c31543039d9e8b2786043ec6c8d00804d5800c5148d6b6f65bc]"
 }
]
}

```

### Exemple : action d'archivage d'images

L'exemple suivant montre une entrée de CloudTrail journal qui montre qu'une image est archivée à l'aide de l'UpdateImageStorageClassaction targetStorageClass définie sur ARCHIVE.

```

{
 "eventVersion": "1.11",
 "userIdentity": {
 "type": "IAMUser",
 "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
 "arn": "arn:aws:sts::123456789012:user/Mary_Major",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 },
 "userName": "Mary_Major",
 "sessionContext": {
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2019-04-15T16:42:14Z"
 }
 }
}

```

```
},
"eventTime": "2019-04-15T16:45:00Z",
"eventSource": "ecr.amazonaws.com",
"eventName": "UpdateImageStorageClass",
"awsRegion": "us-east-2",
"sourceIPAddress": "AWS Internal",
"userAgent": "AWS Internal",
"requestParameters": {
 "repositoryName": "testrepo",
 "imageId": {
 "imageDigest":
"sha256:98c8b060c21d9adbb6b8c41b916e95e6307102786973ab93a41e8b86d1fc6d3e"
 },
 "targetStorageClass": "ARCHIVE",
 "registryId": "123456789012"
},
"responseElements": {
 "image": {
 "registryId": "123456789012",
 "repositoryName": "testrepo",
 "imageId": {
 "imageDigest":
"sha256:98c8b060c21d9adbb6b8c41b916e95e6307102786973ab93a41e8b86d1fc6d3e"
 },
 "imageStatus": "ARCHIVED"
 }
},
"requestID": "cf044b7d-EXAMPLE",
"eventID": "2bfd4ee2-EXAMPLE",
"readOnly": false,
"resources": [{
 "ARN": "arn:aws:ecr:us-east-2:123456789012:repository/testrepo",
 "accountId": "123456789012"
}],
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "123456789012",
"eventCategory": "Management"
}
```

## Exemple : action de restauration d'image

Les exemples suivants présentent des entrées de CloudTrail journal illustrant la restauration d'une image. Lorsque vous restaurez une image archivée, deux événements sont générés :

1. Un événement d'appel d'API lorsque la restauration est lancée
2. Un événement de service lorsque l'opération de restauration asynchrone est terminée

### Événement d'appel d'API (lancement de la restauration)

L'exemple suivant montre l'appel d'API initial pour restaurer une image à l'aide de l'UpdateImageStorageClass action targetStorageClass définie sur STANDARD. La réponse indique l'état de l'image sous la forme ACTIVATING.

```
{
 "eventVersion": "1.11",
 "userIdentity": {
 "type": "IAMUser",
 "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
 "arn": "arn:aws:sts::123456789012:user/Mary_Major",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 },
 "userName": "Mary_Major",
 "sessionContext": {
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2019-04-15T16:42:14Z"
 }
 },
},
"eventTime": "2019-04-15T16:45:00Z",
"eventSource": "ecr.amazonaws.com",
"eventName": "UpdateImageStorageClass",
"awsRegion": "us-east-2",
"sourceIPAddress": "AWS Internal",
"userAgent": "AWS Internal",
"requestParameters": {
 "repositoryName": "testrepo",
 "imageId": {
 "imageDigest":
"sha256:98c8b060c21d9adbb6b8c41b916e95e6307102786973ab93a41e8b86d1fc6d3e"
 },
},
```

```

 "targetStorageClass": "STANDARD",
 "registryId": "123456789012"
 },
 "responseElements": {
 "image": {
 "registryId": "123456789012",
 "repositoryName": "testrepo",
 "imageId": {
 "imageDigest":
"sha256:98c8b060c21d9adbb6b8c41b916e95e6307102786973ab93a41e8b86d1fc6d3e"
 },
 "imageStatus": "ACTIVATING"
 }
 },
 "requestID": "cf044b7d-EXAMPLE",
 "eventID": "2bfd4ee2-EXAMPLE",
 "readOnly": false,
 "resources": [{
 "ARN": "arn:aws:ecr:us-east-2:123456789012:repository/testrepo",
 "accountId": "123456789012"
 }],
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "123456789012",
 "eventCategory": "Management"
}

```

### Événement de service (fin de la restauration)

L'exemple suivant montre l'événement de service généré lorsque l'opération de restauration asynchrone est terminée. Ce type d'événement peut être localisé en filtrant le `ImageActivationEvent` pour le champ du nom d'événement. La `serviceEventDetails` section contient le résultat de la restauration et l'état final de l'image.

```

{
 "eventVersion": "1.11",
 "userIdentity": {
 "accountId": "123456789012",
 "invokedBy": "AWS Internal"
 },
 "eventTime": "2020-03-12T20:22:12Z",
 "eventSource": "ecr.amazonaws.com",
 "eventName": "ImageActivationEvent",

```

```

"awsRegion": "us-west-2",
"sourceIPAddress": "AWS Internal",
"userAgent": "AWS Internal",
"requestParameters": null,
"responseElements": null,
"eventID": "9354dd7f-EXAMPLE",
"readOnly": true,
"resources": [
 {
 "ARN": "arn:aws:ecr:us-west-2:123456789012:repository/testrepo",
 "accountId": "123456789012",
 "type": "AWS::ECR::Repository"
 }
],
"eventType": "AwsServiceEvent",
"managementEvent": true,
"recipientAccountId": "123456789012",
"serviceEventDetails": {
 "repositoryName": "testrepo",
 "imageDigest":
"sha256:98c8b060c21d9adbb6b8c41b916e95e6307102786973ab93a41e8b86d1fc6d3e",
 "targetStorageClass": "STANDARD",
 "result": "SUCCESS",
 "imageStatus": "ACTIVE"
},
"eventCategory": "Management"
}

```

### Exemple : action des référents d'images

L'exemple suivant montre une entrée de AWS CloudTrail journal qui montre quand un client OCI 1.1 conforme récupère une liste de référents, ou d'artefacts de référence, pour une image à l'aide de l'API. `Referrers`

```

{
 "eventVersion": "1.08",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
 "arn": "arn:aws:sts::123456789012:user/Mary_Major",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "sessionContext": {

```

```
 "sessionIssuer": {
 "type": "Role",
 "principalId": "AIDACKCEVSQ6C2EXAMPLE",
 "arn": "arn:aws:iam::123456789012:role/Admin",
 "accountId": "123456789012",
 "userName": "Admin"
 },
 "webIdFederationData": {},
 "attributes": {
 "creationDate": "2024-10-08T16:38:39Z",
 "mfaAuthenticated": "false"
 },
 "ec2RoleDelivery": "2.0"
 },
 "invokedBy": "ecr.amazonaws.com"
},
"eventTime": "2024-10-08T17:22:51Z",
"eventSource": "ecr.amazonaws.com",
"eventName": "ListImageReferrers",
"awsRegion": "us-east-2",
"sourceIPAddress": "ecr.amazonaws.com",
"userAgent": "ecr.amazonaws.com",
"requestParameters": {
 "registryId": "123456789012",
 "repositoryName": "testrepo",
 "subjectId": {
 "imageDigest":
"sha256:000b9b805af1cdb60628898c9f411996301a1c13afd3dbef1d8a16ac6dbf503a"
 },
 "nextToken": "urD72mdD/mC8b5-EXAMPLE"
},
"responseElements": null,
"requestID": "cb8c167e-EXAMPLE",
"eventID": "e3c6f4ce-EXAMPLE",
"readOnly": true,
"resources": [
 {
 "accountId": "123456789012",
 "ARN": "arn:aws:ecr:us-east-2:123456789012:repository/testrepo"
 }
],
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "123456789012",
```

```
"eventCategory": "Management"
}
```

# Utilisation d'Amazon ECR avec un AWS Kit SDK

AWS des kits de développement logiciel (SDK) sont disponibles pour de nombreux langages de programmation populaires. Chaque kit SDK fournit une API, des exemples de code et de la documentation qui facilitent la création d'applications par les développeurs dans leur langage préféré.

| Documentation SDK                           | Exemples de code                                             |
|---------------------------------------------|--------------------------------------------------------------|
| <a href="#">AWS SDK pour C++</a>            | <a href="#">AWS SDK pour C++ exemples de code</a>            |
| <a href="#">AWS CLI</a>                     | <a href="#">AWS CLI exemples de code</a>                     |
| <a href="#">AWS SDK pour Go</a>             | <a href="#">AWS SDK pour Go exemples de code</a>             |
| <a href="#">AWS SDK pour Java</a>           | <a href="#">AWS SDK pour Java exemples de code</a>           |
| <a href="#">AWS SDK pour JavaScript</a>     | <a href="#">AWS SDK pour JavaScript exemples de code</a>     |
| <a href="#">AWS SDK pour Kotlin</a>         | <a href="#">AWS SDK pour Kotlin exemples de code</a>         |
| <a href="#">AWS SDK pour .NET</a>           | <a href="#">AWS SDK pour .NET exemples de code</a>           |
| <a href="#">AWS SDK pour PHP</a>            | <a href="#">AWS SDK pour PHP exemples de code</a>            |
| <a href="#">Outils AWS pour PowerShell</a>  | <a href="#">Outils AWS pour PowerShell exemples de code</a>  |
| <a href="#">AWS SDK pour Python (Boto3)</a> | <a href="#">AWS SDK pour Python (Boto3) exemples de code</a> |
| <a href="#">AWS SDK pour Ruby</a>           | <a href="#">AWS SDK pour Ruby exemples de code</a>           |
| <a href="#">AWS SDK pour Rust</a>           | <a href="#">AWS SDK pour Rust exemples de code</a>           |
| <a href="#">AWS SDK pour SAP ABAP</a>       | <a href="#">AWS SDK pour SAP ABAP exemples de code</a>       |
| <a href="#">AWS SDK pour Swift</a>          | <a href="#">AWS SDK pour Swift exemples de code</a>          |

 Exemple de disponibilité

Vous n'avez pas trouvé ce dont vous avez besoin ? Demandez un exemple de code en utilisant le lien [Provide feedback](#) (Fournir un commentaire) en bas de cette page.

# Exemples de code pour Amazon ECR à l'aide de kits SDK AWS

Les exemples de code suivants montrent comment utiliser Amazon ECR avec un kit de développement AWS logiciel (SDK).

Les principes de base sont des exemples de code qui vous montrent comment effectuer les opérations essentielles au sein d'un service.

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service individuelles, vous pouvez les voir en contexte dans leurs scénarios associés.

Les scénarios sont des exemples de code qui vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions au sein d'un même service ou combinés à d'autres Services AWS.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit de développement logiciel (SDK).

## Exemples de code

- [Exemples de base pour Amazon ECR utilisant des SDK AWS](#)
  - [Bonjour Amazon ECR](#)
  - [Découvrez les bases d'Amazon ECR avec un SDK AWS](#)
  - [Actions pour Amazon ECR à l'aide de kits SDK AWS](#)
    - [Utilisation CreateRepository avec un AWS SDK ou une CLI](#)
    - [Utilisation DeleteRepository avec un AWS SDK ou une CLI](#)
    - [Utilisation DescribeImages avec un AWS SDK ou une CLI](#)
    - [Utilisation DescribeRepositories avec un AWS SDK ou une CLI](#)
    - [Utilisation GetAuthorizationToken avec un AWS SDK ou une CLI](#)
    - [Utilisation GetRepositoryPolicy avec un AWS SDK ou une CLI](#)
    - [Utilisation ListImages avec un AWS SDK ou une CLI](#)

- [Utilisation PushImageCmd avec un AWS SDK](#)
- [Utilisation PutLifecyclePolicy avec un AWS SDK ou une CLI](#)
- [Utilisation SetRepositoryPolicy avec un AWS SDK ou une CLI](#)
- [Utilisation StartLifecyclePolicyPreview avec un AWS SDK ou une CLI](#)
- [Scénarios pour Amazon ECR utilisant des SDK AWS](#)
  - [Commencer à utiliser Amazon ECR](#)

## Exemples de base pour Amazon ECR utilisant des SDK AWS

Les exemples de code suivants illustrent comment utiliser les principes de base d'Amazon Elastic Container Registry avec des kits AWS SDK.

### Exemples


- [Bonjour Amazon ECR](#)
- [Découvrez les bases d'Amazon ECR avec un SDK AWS](#)
- [Actions pour Amazon ECR à l'aide de kits SDK AWS](#)
  - [Utilisation CreateRepository avec un AWS SDK ou une CLI](#)
  - [Utilisation DeleteRepository avec un AWS SDK ou une CLI](#)
  - [Utilisation DescribeImages avec un AWS SDK ou une CLI](#)
  - [Utilisation DescribeRepositories avec un AWS SDK ou une CLI](#)
  - [Utilisation GetAuthorizationToken avec un AWS SDK ou une CLI](#)
  - [Utilisation GetRepositoryPolicy avec un AWS SDK ou une CLI](#)
  - [Utilisation ListImages avec un AWS SDK ou une CLI](#)
  - [Utilisation PushImageCmd avec un AWS SDK](#)
  - [Utilisation PutLifecyclePolicy avec un AWS SDK ou une CLI](#)
  - [Utilisation SetRepositoryPolicy avec un AWS SDK ou une CLI](#)
  - [Utilisation StartLifecyclePolicyPreview avec un AWS SDK ou une CLI](#)

## Bonjour Amazon ECR

Les exemples de code suivants montrent comment démarrer avec Amazon ECR.

## Java

## SDK pour Java 2.x

 Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ecr.EcrClient;
import software.amazon.awssdk.services.ecr.model.EcrException;
import software.amazon.awssdk.services.ecr.model.ListImagesRequest;
import software.amazon.awssdk.services.ecr.paginators.ListImagesIterable;

public class HelloECR {

 public static void main(String[] args) {
 final String usage = ""
 Usage: <repositoryName>

 Where:
 repositoryName - The name of the Amazon ECR repository.
 "";

 if (args.length != 1) {
 System.out.println(usage);
 System.exit(1);
 }

 String repoName = args[0];
 EcrClient ecrClient = EcrClient.builder()
 .region(Region.US_EAST_1)
 .build();

 listImageTags(ecrClient, repoName);
 }

 public static void listImageTags(EcrClient ecrClient, String repoName){
 ListImagesRequest listImagesPaginator = ListImagesRequest.builder()
 .repositoryName(repoName)
 .build();
```

```
ListImagesIterable imagesIterable =
ecrClient.listImagesPaginator(listImagesPaginator);
imagesIterable.stream()
 .flatMap(r -> r.imageIds().stream())
 .forEach(image -> System.out.println("The docker image tag is: "
+image.imageTag()));
}
```

- Pour plus de détails sur l'API, consultez [listImages](#) dans Référence des API du kit AWS SDK for Java 2.x .

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
import aws.sdk.kotlin.services.ecr.EcrClient
import aws.sdk.kotlin.services.ecr.model.ListImagesRequest
import kotlin.system.exitProcess

suspend fun main(args: Array<String>) {
 val usage = """
 Usage: <repositoryName>

 Where:
 repositoryName - The name of the Amazon ECR repository.

 """.trimIndent()

 if (args.size != 1) {
 println(usage)
 exitProcess(1)
 }
}
```

```
 val repoName = args[0]
 listImageTags(repoName)
}

suspend fun listImageTags(repoName: String?) {
 val listImages =
 ListImagesRequest {
 repositoryName = repoName
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val imageResponse = ecrClient.listImages(listImages)
 imageResponse.imageIds?.forEach { imageId ->
 println("Image tag: ${imageId.imageTag}")
 }
 }
}
```

- Pour plus de détails sur l'API, consultez [listImages](#) dans la Référence des API du kit AWS SDK pour Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
import boto3
import argparse
from boto3 import client

def hello_ecr(ecr_client: client, repository_name: str) -> None:
 """
```

```
Use the AWS SDK for Python (Boto3) to create an Amazon Elastic Container
Registry (Amazon ECR)
client and list the images in a repository.
This example uses the default settings specified in your shared credentials
and config files.

:param ecr_client: A Boto3 Amazon ECR Client object. This object wraps
 the low-level Amazon ECR service API.
:param repository_name: The name of an Amazon ECR repository in your account.
"""
print(
 f"Hello, Amazon ECR! Let's list some images in the repository
'{repository_name}':\n"
)
paginator = ecr_client.get_paginator("list_images")
page_iterator = paginator.paginate(
 repositoryName=repository_name, PaginationConfig={"MaxItems": 10}
)

image_names: [str] = []
for page in page_iterator:
 for schedule in page["imageIds"]:
 image_names.append(schedule["imageTag"])

print(f"{len(image_names)} image(s) retrieved.")
for schedule_name in image_names:
 print(f"\t{schedule_name}")

if __name__ == "__main__":
 parser = argparse.ArgumentParser(description="Run hello Amazon ECR.")
 parser.add_argument(
 "--repository-name",
 type=str,
 help="the name of an Amazon ECR repository in your account.",
 required=True,
)
 args = parser.parse_args()

 hello_ecr(boto3.client("ecr"), args.repository_name)
```

- Pour plus de détails sur l'API, consultez [listImages](#) dans la Référence des API du kit AWS SDK pour Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Découvrez les bases d'Amazon ECR avec un SDK AWS

Les exemples de code suivants montrent comment :

- créer un référentiel Amazon ECR ;
- définir des politiques de référentiel ;
- extraire des URI de référentiel ;
- obtenir des jetons d'autorisation Amazon ECR ;
- définir des politiques de cycle de vie pour les référentiels Amazon ECR ;
- transmettre une image Docker à un référentiel Amazon ECR ;
- vérifier l'existence d'une image dans un référentiel Amazon ECR ;
- répertorier les référentiels Amazon ECR pour votre compte et obtenez des informations les concernant ;
- supprimer des référentiels Amazon ECR.

### Java

#### SDK pour Java 2.x

##### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

Exécutez un scénario interactif illustrant les fonctionnalités Amazon ECR.

```
import software.amazon.awssdk.services.ecr.model.EcrException;
import
software.amazon.awssdk.services.ecr.model.RepositoryPolicyNotFoundException;
```

```
import java.util.Scanner;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 *
 * This Java code example requires an IAM Role that has permissions to interact
 * with the Amazon ECR service.
 *
 * To create an IAM role, see:
 *
 * https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_create.html
 *
 * This Java scenario example requires a local docker image named echo-text.
 * Without a local image,
 * this Java program will not successfully run. For more information including
 * how to create the local
 * image, see:
 *
 * /scenarios/basics/ecr/README
 */
public class ECRScenario {
 public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
 public static void main(String[] args) {
 final String usage = ""
 Usage: <iamRoleARN> <accountId>

 Where:
 iamRoleARN - The IAM role ARN that has the necessary permissions
to access and manage the Amazon ECR repository.
 accountId - Your AWS account number.
 """;

 if (args.length != 2) {
 System.out.println(usage);
 return;
 }
 }
}
```

```
}

 ECRActions ecrActions = new ECRActions();
 String iamRole = args[0];
 String accountId = args[1];
 String localImageName;

 Scanner scanner = new Scanner(System.in);
 System.out.println("""
 The Amazon Elastic Container Registry (ECR) is a fully-managed
 Docker container registry
 service provided by AWS. It allows developers and organizations to
 securely
 store, manage, and deploy Docker container images.
 ECR provides a simple and scalable way to manage container images
 throughout their lifecycle,
 from building and testing to production deployment.\s

 The `EcrAsyncClient` interface in the AWS SDK for Java 2.x provides
 a set of methods to
 programmatically interact with the Amazon ECR service. This allows
 developers to
 automate the storage, retrieval, and management of container images
 as part of their application
 deployment pipelines. With ECR, teams can focus on building and
 deploying their
 applications without having to worry about the underlying
 infrastructure required to
 host and manage a container registry.

 This scenario walks you through how to perform key operations for
 this service.
 Let's get started...

 You have two choices:
 1 - Run the entire program.
 2 - Delete an existing Amazon ECR repository named echo-text (created
 from a previous execution of
 this program that did not complete).
 """);

 while (true) {
 String input = scanner.nextLine();
 if (input.trim().equalsIgnoreCase("1")) {
```

```
 System.out.println("Continuing with the program...");
 System.out.println("");
 break;
 } else if (input.trim().equalsIgnoreCase("2")) {
 String repoName = "echo-text";
 ecrActions.deleteECRRepository(repoName);
 return;
 } else {
 // Handle invalid input.
 System.out.println("Invalid input. Please try again.");
 }
}

waitForInputToContinue(scanner);
System.out.println(DASHES);

System.out.println("""
 1. Create an ECR repository.

 The first task is to ensure we have a local Docker image named echo-
text.

 If this image exists, then an Amazon ECR repository is created.

 An ECR repository is a private Docker container repository provided
by Amazon Web Services (AWS). It is a managed service that makes it
easy

 to store, manage, and deploy Docker container images.\s
 """);

// Ensure that a local docker image named echo-text exists.
boolean doesExist = ecrActions.isEchoTextImagePresent();
String repoName;
if (!doesExist){
 System.out.println("The local image named echo-text does not exist");
 return;
} else {
 localImageName = "echo-text";
 repoName = "echo-text";
}

try {
 String repoArn = ecrActions.createECRRepository(repoName);
 System.out.println("The ARN of the ECR repository is " + repoArn);
```

```

 } catch (IllegalArgumentException e) {
 System.err.println("Invalid repository name: " + e.getMessage());
 return;
 } catch (RuntimeException e) {
 System.err.println("An error occurred while creating the ECR
repository: " + e.getMessage());
 e.printStackTrace();
 return;
 }
 waitForInputToContinue(scanner);

 System.out.println(DASHES);
 System.out.println("""
2. Set an ECR repository policy.

 Setting an ECR repository policy using the `setRepositoryPolicy` function
is crucial for maintaining
 the security and integrity of your container images. The repository
policy allows you to
 define specific rules and restrictions for accessing and managing the
images stored within your ECR
repository.
""");
 waitForInputToContinue(scanner);
 try {
 ecrActions.setRepoPolicy(repoName, iamRole);

 } catch (RepositoryPolicyNotFoundException e) {
 System.err.println("Invalid repository name: " + e.getMessage());
 return;
 } catch (EcrException e) {
 System.err.println("An ECR exception occurred: " + e.getMessage());
 return;
 } catch (RuntimeException e) {
 System.err.println("An error occurred while creating the ECR
repository: " + e.getMessage());
 return;
 }
 waitForInputToContinue(scanner);

 System.out.println(DASHES);
 System.out.println("""
3. Display ECR repository policy.

```

```
Now we will retrieve the ECR policy to ensure it was successfully set.
""");
waitForInputToContinue(scanner);
try {
 String policyText = ecrActions.getRepoPolicy(repoName);
 System.out.println("Policy Text:");
 System.out.println(policyText);

} catch (EcrException e) {
 System.err.println("An ECR exception occurred: " + e.getMessage());
 return;
} catch (RuntimeException e) {
 System.err.println("An error occurred while creating the ECR
repository: " + e.getMessage());
 return;
}

waitForInputToContinue(scanner);

System.out.println(DASHES);
System.out.println("""
4. Retrieve an ECR authorization token.
```

You need an authorization token to securely access and interact with the Amazon ECR registry.

The `getAuthorizationToken` method of the `EcrAsyncClient` is responsible for securely accessing and interacting with an Amazon ECR repository. This operation is responsible for obtaining a valid authorization token, which is required to authenticate your requests to the ECR service.

Without a valid authorization token, you would not be able to perform any operations on the ECR repository, such as pushing, pulling, or managing your Docker images.

```
""");
waitForInputToContinue(scanner);
try {
 ecrActions.getAuthToken();

} catch (EcrException e) {
 System.err.println("An ECR exception occurred: " + e.getMessage());
 return;
```

```
 } catch (RuntimeException e) {
 System.err.println("An error occurred while retrieving the
authorization token: " + e.getMessage());
 return;
 }
 waitForInputToContinue(scanner);

 System.out.println(DASHES);
 System.out.println("""
5. Get the ECR Repository URI.
```

The URI of an Amazon ECR repository is important. When you want to deploy a container image to a container orchestration platform like Amazon Elastic Kubernetes Service (EKS) or Amazon Elastic Container Service (ECS), you need to specify the full image URI, which includes the ECR repository URI. This allows the container runtime to pull the correct container image from the ECR repository.

```
""");
 waitForInputToContinue(scanner);

 try {
 ecrActions.getRepositoryURI(repoName);
 } catch (EcrException e) {
 System.err.println("An ECR exception occurred: " + e.getMessage());
 return;
 } catch (RuntimeException e) {
 System.err.println("An error occurred while retrieving the URI: " +
e.getMessage());
 return;
 }
 waitForInputToContinue(scanner);

 System.out.println(DASHES);
 System.out.println("""
6. Set an ECR Lifecycle Policy.
```

An ECR Lifecycle Policy is used to manage the lifecycle of Docker images stored in your ECR repositories.

These policies allow you to automatically remove old or unused Docker images from your repositories, freeing up storage space and reducing costs.

This example policy helps to maintain the size and efficiency of the container registry by automatically removing older and potentially unused images, ensuring that the storage is optimized and the registry remains up-to-date.

```
 """);
 waitForInputToContinue(scanner);
 try {
 ecrActions.setLifecyclePolicy(repoName);

 } catch (RuntimeException e) {
 System.err.println("An error occurred while setting the lifecycle
policy: " + e.getMessage());
 e.printStackTrace();
 return;
 }
 waitForInputToContinue(scanner);

 System.out.println(DASHES);
 System.out.println("""
7. Push a docker image to the Amazon ECR Repository.
```

The `pushImageCmd()` method pushes a local Docker image to an Amazon ECR repository.

It sets up the Docker client by connecting to the local Docker host using the default port.

It then retrieves the authorization token for the ECR repository by making a call to the AWS SDK.

The method uses the authorization token to create an `AuthConfig` object, which is used to authenticate

the Docker client when pushing the image. Finally, the method tags the Docker image with the specified

repository name and image tag, and then pushes the image to the ECR repository using the Docker client.

If the push operation is successful, the method prints a message indicating that the image was pushed to ECR.

```
 """);
 waitForInputToContinue(scanner);
```

```
try {
 ecrActions.pushDockerImage(repoName, localImageName);

} catch (RuntimeException e) {
 System.err.println("An error occurred while pushing a local Docker
image to Amazon ECR: " + e.getMessage());
 e.printStackTrace();
 return;
}
waitForInputToContinue(scanner);

System.out.println(DASHES);
System.out.println("8. Verify if the image is in the ECR Repository.");
waitForInputToContinue(scanner);
try {
 ecrActions.verifyImage(repoName, localImageName);

} catch (EcrException e) {
 System.err.println("An ECR exception occurred: " + e.getMessage());
 return;
} catch (RuntimeException e) {
 System.err.println("An error occurred " + e.getMessage());
 e.printStackTrace();
 return;
}
waitForInputToContinue(scanner);

System.out.println(DASHES);
System.out.println("9. As an optional step, you can interact with the
image in Amazon ECR by using the CLI.");
System.out.println("Would you like to view instructions on how to use the
CLI to run the image? (y/n)");
String ans = scanner.nextLine().trim();
if (ans.equalsIgnoreCase("y")) {
 String instructions = ""
 1. Authenticate with ECR - Before you can pull the image from Amazon
ECR, you need to authenticate with the registry. You can do this using the AWS
CLI:

 aws ecr get-login-password --region us-east-1 | docker login --
username AWS --password-stdin %s.dkr.ecr.us-east-1.amazonaws.com

 2. Describe the image using this command:
```

```

aws ecr describe-images --repository-name %s --image-ids imageTag=
%s

3. Run the Docker container and view the output using this command:

docker run --rm %s.dkr.ecr.us-east-1.amazonaws.com/%s:%s
""";

instructions = String.format(instructions, accountId, repoName,
localImageName, accountId, repoName, localImageName);
System.out.println(instructions);
}
waitForInputToContinue(scanner);

System.out.println(DASHES);
System.out.println("10. Delete the ECR Repository.");
System.out.println(
""")
If the repository isn't empty, you must either delete the contents of the
repository
or use the force option (used in this scenario) to delete the repository
and have Amazon ECR delete all of its contents
on your behalf.
""");
System.out.println("Would you like to delete the Amazon ECR Repository?
(y/n)");
String delAns = scanner.nextLine().trim();
if (delAns.equalsIgnoreCase("y")) {
 System.out.println("You selected to delete the AWS ECR resources.");

 try {
 ecrActions.deleteECRRepository(repoName);

 } catch (EcrException e) {
 System.err.println("An ECR exception occurred: " +
e.getMessage());
 return;
 } catch (RuntimeException e) {
 System.err.println("An error occurred while deleting the Docker
image: " + e.getMessage());
 e.printStackTrace();
 return;
 }
}
}

```

```
 System.out.println(DASHES);
 System.out.println("This concludes the Amazon ECR SDK scenario");
 System.out.println(DASHES);
 }

 private static void waitForInputToContinue(Scanner scanner) {
 while (true) {
 System.out.println("");
 System.out.println("Enter 'c' followed by <ENTER> to continue:");
 String input = scanner.nextLine();

 if (input.trim().equalsIgnoreCase("c")) {
 System.out.println("Continuing with the program...");
 System.out.println("");
 break;
 } else {
 // Handle invalid input.
 System.out.println("Invalid input. Please try again.");
 }
 }
 }
}
```

Une classe d'encapsuleur pour les méthodes du kit SDK Amazon ECR.

```
import com.github.dockerjava.api.DockerClient;
import com.github.dockerjava.api.exception.DockerClientException;
import com.github.dockerjava.api.model.AuthConfig;
import com.github.dockerjava.api.model.Image;
import com.github.dockerjava.core.DockerClientBuilder;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.http.async.SdkAsyncHttpClient;
import software.amazon.awssdk.http.nio.netty.NettyNioAsyncHttpClient;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ecr.EcrAsyncClient;
import software.amazon.awssdk.services.ecr.model.AuthorizationData;
import software.amazon.awssdk.services.ecr.model.CreateRepositoryRequest;
import software.amazon.awssdk.services.ecr.model.CreateRepositoryResponse;
import software.amazon.awssdk.services.ecr.model.DeleteRepositoryRequest;
```

```
import software.amazon.awssdk.services.ecr.model.DeleteRepositoryResponse;
import software.amazon.awssdk.services.ecr.model.DescribeImagesRequest;
import software.amazon.awssdk.services.ecr.model.DescribeImagesResponse;
import software.amazon.awssdk.services.ecr.model.DescribeRepositoriesRequest;
import software.amazon.awssdk.services.ecr.model.DescribeRepositoriesResponse;
import software.amazon.awssdk.services.ecr.model.EcrException;
import software.amazon.awssdk.services.ecr.model.GetAuthorizationTokenResponse;
import software.amazon.awssdk.services.ecr.model.GetRepositoryPolicyRequest;
import software.amazon.awssdk.services.ecr.model.GetRepositoryPolicyResponse;
import software.amazon.awssdk.services.ecr.model.ImageIdentifier;
import software.amazon.awssdk.services.ecr.model.Repository;
import
 software.amazon.awssdk.services.ecr.model.RepositoryPolicyNotFoundException;
import software.amazon.awssdk.services.ecr.model.SetRepositoryPolicyRequest;
import software.amazon.awssdk.services.ecr.model.SetRepositoryPolicyResponse;
import
 software.amazon.awssdk.services.ecr.model.StartLifecyclePolicyPreviewRequest;
import
 software.amazon.awssdk.services.ecr.model.StartLifecyclePolicyPreviewResponse;
import com.github.dockerjava.api.command.DockerCmdExecFactory;
import com.github.dockerjava.netty.NettyDockerCmdExecFactory;
import java.time.Duration;
import java.util.Base64;
import java.util.List;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.CompletionException;

public class ECRActions {
 private static EcrAsyncClient ecrClient;

 private static DockerClient dockerClient;

 private static Logger logger = LoggerFactory.getLogger(ECRActions.class);

 /**
 * Creates an Amazon Elastic Container Registry (Amazon ECR) repository.
 *
 * @param repoName the name of the repository to create.
 * @return the Amazon Resource Name (ARN) of the created repository, or an
 empty string if the operation failed.
 * @throws IllegalArgumentException If repository name is invalid.
 * @throws RuntimeException if an error occurs while creating the
 repository.
 */
}
```

```
public String createECRRepository(String repoName) {
 if (repoName == null || repoName.isEmpty()) {
 throw new IllegalArgumentException("Repository name cannot be null or
empty");
 }

 CreateRepositoryRequest request = CreateRepositoryRequest.builder()
 .repositoryName(repoName)
 .build();

 CompletableFuture<CreateRepositoryResponse> response =
getAsyncClient().createRepository(request);
 try {
 CreateRepositoryResponse result = response.join();
 if (result != null) {
 System.out.println("The " + repoName + " repository was created
successfully.");
 return result.repository().repositoryArn();
 } else {
 throw new RuntimeException("Unexpected response type");
 }
 } catch (CompletionException e) {
 Throwable cause = e.getCause();
 if (cause instanceof EcrException ex) {
 if
("RepositoryAlreadyExistsException".equals(ex.awsErrorDetails().errorCode())) {
 System.out.println("The Amazon ECR repository already exists,
moving on...");
 DescribeRepositoriesRequest describeRequest =
DescribeRepositoriesRequest.builder()
 .repositoryNames(repoName)
 .build();
 DescribeRepositoriesResponse describeResponse =
getAsyncClient().describeRepositories(describeRequest).join();
 return
describeResponse.repositories().get(0).repositoryArn();
 } else {
 throw new RuntimeException(ex);
 }
 } else {
 throw new RuntimeException(e);
 }
 }
}
```

```
/**
 * Deletes an ECR (Elastic Container Registry) repository.
 *
 * @param repoName the name of the repository to delete.
 * @throws IllegalArgumentException if the repository name is null or empty.
 * @throws EcrException if there is an error deleting the repository.
 * @throws RuntimeException if an unexpected error occurs during the deletion
process.
 */
public void deleteECRRepository(String repoName) {
 if (repoName == null || repoName.isEmpty()) {
 throw new IllegalArgumentException("Repository name cannot be null or
empty");
 }

 DeleteRepositoryRequest repositoryRequest =
DeleteRepositoryRequest.builder()
 .force(true)
 .repositoryName(repoName)
 .build();

 CompletableFuture<DeleteRepositoryResponse> response =
getAsyncClient().deleteRepository(repositoryRequest);
 response.whenComplete((deleteRepositoryResponse, ex) -> {
 if (deleteRepositoryResponse != null) {
 System.out.println("You have successfully deleted the " +
repoName + " repository");
 } else {
 Throwable cause = ex.getCause();
 if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
 }
 }
 });

 // Wait for the CompletableFuture to complete
 response.join();
}
```

```
private static DockerClient getDockerClient() {
 String osName = System.getProperty("os.name");
 if (osName.startsWith("Windows")) {
 // Make sure Docker Desktop is running.
 String dockerHost = "tcp://localhost:2375"; // Use the Docker Desktop
default port.
 DockerCmdExecFactory dockerCmdExecFactory = new
NettyDockerCmdExecFactory().withReadTimeout(20000).withConnectTimeout(20000);
 dockerClient =
DockerClientBuilder.getInstance(dockerHost).withDockerCmdExecFactory(dockerCmdExecFactory)
 } else {
 dockerClient = DockerClientBuilder.getInstance().build();
 }
 return dockerClient;
}

/**
 * Retrieves an asynchronous Amazon Elastic Container Registry (ECR) client.
 *
 * @return the configured ECR asynchronous client.
 */
private static EcrAsyncClient getAsyncClient() {

 /*
 The `NettyNioAsyncHttpClient` class is part of the AWS SDK for Java,
version 2,
 and it is designed to provide a high-performance, asynchronous HTTP
client for interacting with AWS services.
 It uses the Netty framework to handle the underlying network
communication and the Java NIO API to
 provide a non-blocking, event-driven approach to HTTP requests and
responses.
 */
 SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
 .maxConcurrency(50) // Adjust as needed.
 .connectionTimeout(Duration.ofSeconds(60)) // Set the connection
timeout.
 .readTimeout(Duration.ofSeconds(60)) // Set the read timeout.
 .writeTimeout(Duration.ofSeconds(60)) // Set the write timeout.
 .build();

 ClientOverrideConfiguration overrideConfig =
ClientOverrideConfiguration.builder()
```

```
 .apiCallTimeout(Duration.ofMinutes(2)) // Set the overall API call
timeout.
 .apiCallAttemptTimeout(Duration.ofSeconds(90)) // Set the individual
call attempt timeout.
 .build();

 if (ecrClient == null) {
 ecrClient = EcrAsyncClient.builder()
 .region(Region.US_EAST_1)
 .httpClient(httpClient)
 .overrideConfiguration(overrideConfig)
 .build();
 }
 return ecrClient;
}

/**
 * Sets the lifecycle policy for the specified repository.
 *
 * @param repoName the name of the repository for which to set the lifecycle
policy.
 */
public void setLifeCyclePolicy(String repoName) {
 /*
 This policy helps to maintain the size and efficiency of the container
registry
by automatically removing older and potentially unused images,
ensuring that the storage is optimized and the registry remains up-to-
date.
 */
 String polText = ""
 {
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images older than 14 days",
 "selection": {
 "tagStatus": "any",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 14
 },
 "action": {
 "type": "expire"
 }
 }
]
 }
```

```

 }
 }
}
}
}";

 StartLifecyclePolicyPreviewRequest lifecyclePolicyPreviewRequest =
StartLifecyclePolicyPreviewRequest.builder()
 .lifecyclePolicyText(polText)
 .repositoryName(repoName)
 .build();

 CompletableFuture<StartLifecyclePolicyPreviewResponse> response =
getAsyncClient().startLifecyclePolicyPreview(lifecyclePolicyPreviewRequest);
 response.whenComplete((lifecyclePolicyPreviewResponse, ex) -> {
 if (lifecyclePolicyPreviewResponse != null) {
 System.out.println("Lifecycle policy preview started
successfully.");
 } else {
 if (ex.getCause() instanceof EcrException) {
 throw (EcrException) ex.getCause();
 } else {
 String errorMessage = "Unexpected error occurred: " +
ex.getMessage();
 throw new RuntimeException(errorMessage, ex);
 }
 }
 });
 // Wait for the CompletableFuture to complete.
 response.join();
}

/**
 * Verifies the existence of an image in an Amazon Elastic Container Registry
(Amazon ECR) repository asynchronously.
 *
 * @param repositoryName The name of the Amazon ECR repository.
 * @param imageTag The tag of the image to verify.
 * @throws EcrException if there is an error retrieving the image
information from Amazon ECR.
 * @throws CompletionException if the asynchronous operation completes
exceptionally.
 */
public void verifyImage(String repositoryName, String imageTag) {

```

```
DescribeImagesRequest request = DescribeImagesRequest.builder()
 .repositoryName(repositoryName)
 .imageIds(ImageIdentifier.builder().imageTag(imageTag).build())
 .build();

CompletableFuture<DescribeImagesResponse> response =
getAsyncClient().describeImages(request);
response.whenComplete((describeImagesResponse, ex) -> {
 if (ex != null) {
 if (ex instanceof CompletionException) {
 Throwable cause = ex.getCause();
 if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
 }
 } else {
 throw new RuntimeException("Unexpected error: " +
ex.getCause());
 }
 } else if (describeImagesResponse != null && !
describeImagesResponse.imageDetails().isEmpty()) {
 System.out.println("Image is present in the repository.");
 } else {
 System.out.println("Image is not present in the repository.");
 }
});

// Wait for the CompletableFuture to complete.
response.join();
}

/**
 * Retrieves the repository URI for the specified repository name.
 *
 * @param repoName the name of the repository to retrieve the URI for.
 * @return the repository URI for the specified repository name.
 * @throws EcrException if there is an error retrieving the repository
information.
 * @throws CompletionException if the asynchronous operation completes
exceptionally.
 */
public void getRepositoryURI(String repoName) {
```

```
DescribeRepositoriesRequest request =
DescribeRepositoriesRequest.builder()
 .repositoryNames(repoName)
 .build();

CompletableFuture<DescribeRepositoriesResponse> response =
getAsyncClient().describeRepositories(request);
response.whenComplete((describeRepositoriesResponse, ex) -> {
 if (ex != null) {
 Throwable cause = ex.getCause();
 if (cause instanceof InterruptedException) {
 Thread.currentThread().interrupt();
 String errorMessage = "Thread interrupted while waiting for
asynchronous operation: " + cause.getMessage();
 throw new RuntimeException(errorMessage, cause);
 } else if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 String errorMessage = "Unexpected error: " +
cause.getMessage();
 throw new RuntimeException(errorMessage, cause);
 }
 } else {
 if (describeRepositoriesResponse != null) {
 if (!describeRepositoriesResponse.repositories().isEmpty()) {
 String repositoryUri =
describeRepositoriesResponse.repositories().get(0).repositoryUri();
 System.out.println("Repository URI found: " +
repositoryUri);
 } else {
 System.out.println("No repositories found for the given
name.");
 }
 } else {
 System.err.println("No response received from
describeRepositories.");
 }
 }
});
response.join();
}

/**
```

```
 * Retrieves the authorization token for Amazon Elastic Container Registry
 (ECR).
 * This method makes an asynchronous call to the ECR client to retrieve the
 authorization token.
 * If the operation is successful, the method prints the token to the
 console.
 * If an exception occurs, the method handles the exception and prints the
 error message.
 *
 * @throws EcrException if there is an error retrieving the authorization
 token from ECR.
 * @throws RuntimeException if there is an unexpected error during the
 operation.
 */
 public void getAuthToken() {
 CompletableFuture<GetAuthorizationTokenResponse> response =
getAsyncClient().getAuthorizationToken();
 response.whenComplete((authorizationTokenResponse, ex) -> {
 if (authorizationTokenResponse != null) {
 AuthorizationData authorizationData =
authorizationTokenResponse.authorizationData().get(0);
 String token = authorizationData.authorizationToken();
 if (!token.isEmpty()) {
 System.out.println("The token was successfully retrieved.");
 }
 } else {
 if (ex.getCause() instanceof EcrException) {
 throw (EcrException) ex.getCause();
 } else {
 String errorMessage = "Unexpected error occurred: " +
ex.getMessage();
 throw new RuntimeException(errorMessage, ex); // Rethrow the
exception
 }
 }
 });
 response.join();
 }

/**
 * Gets the repository policy for the specified repository.
 *
 * @param repoName the name of the repository.
 */
```

```
 * @throws EcrException if an AWS error occurs while getting the repository
 policy.
 */
 public String getRepoPolicy(String repoName) {
 if (repoName == null || repoName.isEmpty()) {
 throw new IllegalArgumentException("Repository name cannot be null or
empty");
 }

 GetRepositoryPolicyRequest getRepositoryPolicyRequest =
GetRepositoryPolicyRequest.builder()
 .repositoryName(repoName)
 .build();

 CompletableFuture<GetRepositoryPolicyResponse> response =
getAsyncClient().getRepositoryPolicy(getRepositoryPolicyRequest);
 response.whenComplete((resp, ex) -> {
 if (resp != null) {
 System.out.println("Repository policy retrieved successfully.");
 } else {
 if (ex.getCause() instanceof EcrException) {
 throw (EcrException) ex.getCause();
 } else {
 String errorMessage = "Unexpected error occurred: " +
ex.getMessage();
 throw new RuntimeException(errorMessage, ex);
 }
 }
 });

 GetRepositoryPolicyResponse result = response.join();
 return result != null ? result.policyText() : null;
 }

/**
 * Sets the repository policy for the specified ECR repository.
 *
 * @param repoName the name of the ECR repository.
 * @param iamRole the IAM role to be granted access to the repository.
 * @throws RepositoryPolicyNotFoundException if the repository policy does
not exist.
 * @throws EcrException if there is an unexpected error
setting the repository policy.
 */
```

```
public void setRepoPolicy(String repoName, String iamRole) {
 /*
 This example policy document grants the specified AWS principal the
 permission to perform the
 `ecr:BatchGetImage` action. This policy is designed to allow the
 specified principal
 to retrieve Docker images from the ECR repository.
 */
 String policyDocumentTemplate = ""
 {
 "Version": "2012-10-17",
 "Statement" : [{
 "Sid" : "new statement",
 "Effect" : "Allow",
 "Principal" : {
 "AWS" : "%s"
 },
 "Action" : "ecr:BatchGetImage"
 }]
 }
 """;

 String policyDocument = String.format(policyDocumentTemplate, iamRole);
 SetRepositoryPolicyRequest setRepositoryPolicyRequest =
 SetRepositoryPolicyRequest.builder()
 .repositoryName(repoName)
 .policyText(policyDocument)
 .build();

 CompletableFuture<SetRepositoryPolicyResponse> response =
 getAsyncClient().setRepositoryPolicy(setRepositoryPolicyRequest);
 response.whenComplete((resp, ex) -> {
 if (resp != null) {
 System.out.println("Repository policy set successfully.");
 } else {
 Throwable cause = ex.getCause();
 if (cause instanceof RepositoryPolicyNotFoundException) {
 throw (RepositoryPolicyNotFoundException) cause;
 } else if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 String errorMessage = "Unexpected error: " +
 cause.getMessage();
 throw new RuntimeException(errorMessage, cause);
 }
 }
 });
}
```

```

 }
 }
});
response.join();
}

/**
 * Pushes a Docker image to an Amazon Elastic Container Registry (ECR)
 * repository.
 *
 * @param repoName the name of the ECR repository to push the image to.
 * @param imageName the name of the Docker image.
 */
public void pushDockerImage(String repoName, String imageName) {
 System.out.println("Pushing " + imageName + " to Amazon ECR will take a
few seconds.");
 CompletableFuture<AuthConfig> authResponseFuture =
getAsyncClient().getAuthorizationToken()
 .thenApply(response -> {
 String token =
response.authorizationData().get(0).authorizationToken();
 String decodedToken = new
String(Base64.getDecoder().decode(token));
 String password = decodedToken.substring(4);

 DescribeRepositoriesResponse descrRepoResponse =
getAsyncClient().describeRepositories(b -> b.repositoryNames(repoName)).join();
 Repository repoData =
descrRepoResponse.repositories().stream().filter(r ->
r.repositoryName().equals(repoName)).findFirst().orElse(null);
 assert repoData != null;
 String registryURL = repoData.repositoryUri().split("/")[0];

 AuthConfig authConfig = new AuthConfig()
 .withUsername("AWS")
 .withPassword(password)
 .withRegistryAddress(registryURL);
 return authConfig;
 })
 .thenCompose(authConfig -> {
 DescribeRepositoriesResponse descrRepoResponse =
getAsyncClient().describeRepositories(b -> b.repositoryNames(repoName)).join();

```

```
 Repository repoData =
descrRepoResponse.repositories().stream().filter(r ->
r.repositoryName().equals(repoName)).findFirst().orElse(null);
 getDockerClient().tagImageCmd(imageName + ":latest",
repoData.repositoryUri() + ":latest", imageName).exec();
 try {

getDockerClient().pushImageCmd(repoData.repositoryUri()).withTag("echo-
text").withAuthConfig(authConfig).start().awaitCompletion();
 System.out.println("The " + imageName + " was pushed to
ECR");

 } catch (InterruptedException e) {
 throw (RuntimeException) e.getCause();
 }
 return CompletableFuture.completedFuture(authConfig);
 });

 authResponseFuture.join();
}

// Make sure local image echo-text exists.
public boolean isEchoTextImagePresent() {
 try {
 List<Image> images = getDockerClient().listImagesCmd().exec();
 boolean helloWorldFound = false;
 for (Image image : images) {
 String[] repoTags = image.getRepoTags();
 if (repoTags != null) {
 for (String tag : repoTags) {
 if (tag.startsWith("echo-text")) {
 System.out.println(tag);
 helloWorldFound = true;
 }
 }
 }
 }
 if (helloWorldFound) {
 System.out.println("The local image named echo-text exists.");
 return true;
 } else {
 System.out.println("The local image named echo-text does not
exist.");
 return false;
 }
 }
}
```

```
 }
 } catch (DockerClientException ex) {
 logger.error("ERROR: " + ex.getMessage());
 return false;
 }
 }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Java 2.x .
  - [CreateRepository](#)
  - [DeleteRepository](#)
  - [DescribeImages](#)
  - [DescribeRepositories](#)
  - [GetAuthorizationToken](#)
  - [GetRepositoryPolicy](#)
  - [SetRepositoryPolicy](#)
  - [StartLifecyclePolicyPreview](#)

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

Exécutez un scénario interactif illustrant les fonctionnalités Amazon ECR.

```
import java.util.Scanner

/**
 * Before running this Kotlin code example, set up your development environment,
 * including your credentials.
 */
```

```
* For more information, see the following documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html
*
* This code example requires an IAM Role that has permissions to interact with
the Amazon ECR service.
*
* To create an IAM role, see:
*
* https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_create.html
*
* This code example requires a local docker image named echo-text. Without a
local image,
* this program will not successfully run. For more information including how to
create the local
* image, see:
*
* /scenarios/basics/ecr/README
*
```

```
val DASHES = String(CharArray(80)).replace("\u0000", "-")
```

```
suspend fun main(args: Array<String>) {
 val usage =
 """
 Usage: <iamRoleARN> <accountId>

 Where:
 iamRoleARN - The IAM role ARN that has the necessary permissions to
access and manage the Amazon ECR repository.
 accountId - Your AWS account number.

 """.trimIndent()

 if (args.size != 2) {
 println(usage)
 return
 }

 var iamRole = args[0]
 var localImageName: String
 var accountId = args[1]
 val ecrActions = ECRActions()
```

```
val scanner = Scanner(System.`in`)

println(
 """
 The Amazon Elastic Container Registry (ECR) is a fully-managed Docker
 container registry
 service provided by AWS. It allows developers and organizations to
 securely
 store, manage, and deploy Docker container images.
 ECR provides a simple and scalable way to manage container images
 throughout their lifecycle,
 from building and testing to production deployment.

 The `EcrClient` service client that is part of the AWS SDK for Kotlin
 provides a set of methods to
 programmatically interact with the Amazon ECR service. This allows
 developers to
 automate the storage, retrieval, and management of container images as
 part of their application
 deployment pipelines. With ECR, teams can focus on building and deploying
 their
 applications without having to worry about the underlying infrastructure
 required to
 host and manage a container registry.

 This scenario walks you through how to perform key operations for this
 service.
 Let's get started...

 You have two choices:
 1 - Run the entire program.
 2 - Delete an existing Amazon ECR repository named echo-text (created
 from a previous execution of
 this program that did not complete).

 """.trimIndent(),
)

while (true) {
 val input = scanner.nextLine()
 if (input.trim { it <= ' ' }.equals("1", ignoreCase = true)) {
 println("Continuing with the program...")
 println("")
 break
 }
}
```

```

 } else if (input.trim { it <= ' ' }.equals("2", ignoreCase = true)) {
 val repoName = "echo-text"
 ecrActions.deleteECRRepository(repoName)
 return
 } else {
 // Handle invalid input.
 println("Invalid input. Please try again.")
 }
}

```

```

waitForInputToContinue(scanner)
println(DASHES)
println(
 """

```

1. Create an ECR repository.

The first task is to ensure we have a local Docker image named echo-text.

If this image exists, then an Amazon ECR repository is created.

An ECR repository is a private Docker container repository provided by Amazon Web Services (AWS). It is a managed service that makes it easy to store, manage, and deploy Docker container images.

```

 """).trimIndent(),
)

```

```

// Ensure that a local docker image named echo-text exists.
val doesExist = ecrActions.listLocalImages()
val repoName: String
if (!doesExist) {
 println("The local image named echo-text does not exist")
 return
} else {
 localImageName = "echo-text"
 repoName = "echo-text"
}

```

```

val repoArn = ecrActions.createECRRepository(repoName).toString()
println("The ARN of the ECR repository is $repoArn")
waitForInputToContinue(scanner)

```

```

println(DASHES)
println(

```

```
"""
```

## 2. Set an ECR repository policy.

Setting an ECR repository policy using the `setRepositoryPolicy` function is crucial for maintaining the security and integrity of your container images. The repository policy allows you to define specific rules and restrictions for accessing and managing the images stored within your ECR repository.

```
 """.trimIndent(),
)
 waitForInputToContinue(scanner)
 ecrActions.setRepoPolicy(repoName, iamRole)
 waitForInputToContinue(scanner)
```

```
println(DASHES)
println(
```

```
 """
```

## 3. Display ECR repository policy.

Now we will retrieve the ECR policy to ensure it was successfully set.

```
 """.trimIndent(),
)
 waitForInputToContinue(scanner)
 val policyText = ecrActions.getRepoPolicy(repoName)
 println("Policy Text:")
 println(policyText)
 waitForInputToContinue(scanner)
```

```
println(DASHES)
println(
```

```
 """
```

## 4. Retrieve an ECR authorization token.

You need an authorization token to securely access and interact with the Amazon ECR registry.

The `getAuthorizationToken` method of the `EcrAsyncClient` is responsible for securely accessing and interacting with an Amazon ECR repository. This operation is responsible for obtaining a valid authorization token, which is required to authenticate your requests to the ECR service.

Without a valid authorization token, you would not be able to perform any operations on the ECR repository, such as pushing, pulling, or managing your Docker images.

```
 """.trimIndent(),
)
 waitForInputToContinue(scanner)
 ecrActions.getAuthToken()
 waitForInputToContinue(scanner)
```

```
println(DASHES)
println(
 """
```

5. Get the ECR Repository URI.

The URI of an Amazon ECR repository is important. When you want to deploy a container image to a container orchestration platform like Amazon Elastic Kubernetes Service (EKS)

or Amazon Elastic Container Service (ECS), you need to specify the full image URI, which includes the ECR repository URI. This allows the container runtime to pull the correct container image from the ECR repository.

```
 """.trimIndent(),
)
 waitForInputToContinue(scanner)
 val repositoryURI: String? = ecrActions.getRepositoryURI(repoName)
 println("The repository URI is $repositoryURI")
 waitForInputToContinue(scanner)
```

```
println(DASHES)
println(
 """
```

6. Set an ECR Lifecycle Policy.

An ECR Lifecycle Policy is used to manage the lifecycle of Docker images stored in your ECR repositories.

These policies allow you to automatically remove old or unused Docker images from your repositories, freeing up storage space and reducing costs.

```
 """.trimIndent(),
)
 waitForInputToContinue(scanner)
 val pol = ecrActions.setLifeCyclePolicy(repoName)
 println(pol)
 waitForInputToContinue(scanner)
```

```
println(DASHES)
println(
```

```
 ""
```

```
 7. Push a docker image to the Amazon ECR Repository.
```

The `pushImageCmd()` method pushes a local Docker image to an Amazon ECR repository.

It sets up the Docker client by connecting to the local Docker host using the default port.

It then retrieves the authorization token for the ECR repository by making a call to the AWS SDK.

The method uses the authorization token to create an `AuthConfig` object, which is used to authenticate

the Docker client when pushing the image. Finally, the method tags the Docker image with the specified repository name and image tag, and then pushes the image to the ECR repository using the Docker client.

If the push operation is successful, the method prints a message indicating that the image was pushed to ECR.

```
 """.trimIndent(),
)

 waitForInputToContinue(scanner)
 ecrActions.pushDockerImage(repoName, localImageName)
 waitForInputToContinue(scanner)

 println(DASHES)
 println("8. Verify if the image is in the ECR Repository.")
 waitForInputToContinue(scanner)
 ecrActions.verifyImage(repoName, localImageName)
 waitForInputToContinue(scanner)

 println(DASHES)
```

```

println("9. As an optional step, you can interact with the image in Amazon
ECR by using the CLI.")
println("Would you like to view instructions on how to use the CLI to run the
image? (y/n)")
val ans = scanner.nextLine().trim()
if (ans.equals("y", true)) {
 val instructions = """
 1. Authenticate with ECR - Before you can pull the image from Amazon ECR,
you need to authenticate with the registry. You can do this using the AWS CLI:

 aws ecr get-login-password --region us-east-1 | docker login --
username AWS --password-stdin $accountId.dkr.ecr.us-east-1.amazonaws.com

 2. Describe the image using this command:

 aws ecr describe-images --repository-name $repoName --image-ids
imageTag=$localImageName

 3. Run the Docker container and view the output using this command:

 docker run --rm $accountId.dkr.ecr.us-east-1.amazonaws.com/$repoName:
$localImageName
 """
 println(instructions)
 }
 waitForInputToContinue(scanner)

 println(DASHES)
 println("10. Delete the ECR Repository.")
 println(
 """
 If the repository isn't empty, you must either delete the contents of the
repository
 or use the force option (used in this scenario) to delete the repository
and have Amazon ECR delete all of its contents
on your behalf.

 """.trimIndent(),
)
 println("Would you like to delete the Amazon ECR Repository? (y/n)")
 val delAns = scanner.nextLine().trim { it <= ' ' }
 if (delAns.equals("y", ignoreCase = true)) {
 println("You selected to delete the AWS ECR resources.")
 waitForInputToContinue(scanner)
 }
}

```

```

 ecrActions.deleteECRRepository(repoName)
 }

 println(DASHES)
 println("This concludes the Amazon ECR SDK scenario")
 println(DASHES)
}

private fun waitForInputToContinue(scanner: Scanner) {
 while (true) {
 println("")
 println("Enter 'c' followed by <ENTER> to continue:")
 val input = scanner.nextLine()
 if (input.trim { it <= ' ' }.equals("c", ignoreCase = true)) {
 println("Continuing with the program...")
 println("")
 break
 } else {
 // Handle invalid input.
 println("Invalid input. Please try again.")
 }
 }
}
}

```

Une classe d'encapsuleur pour les méthodes du kit SDK Amazon ECR.

```

import aws.sdk.kotlin.services.ecr.EcrClient
import aws.sdk.kotlin.services.ecr.model.CreateRepositoryRequest
import aws.sdk.kotlin.services.ecr.model.DeleteRepositoryRequest
import aws.sdk.kotlin.services.ecr.model.DescribeImagesRequest
import aws.sdk.kotlin.services.ecr.model.DescribeRepositoriesRequest
import aws.sdk.kotlin.services.ecr.model.EcrException
import aws.sdk.kotlin.services.ecr.model.GetRepositoryPolicyRequest
import aws.sdk.kotlin.services.ecr.model.ImageIdentifier
import aws.sdk.kotlin.services.ecr.model.RepositoryAlreadyExistsException
import aws.sdk.kotlin.services.ecr.model.SetRepositoryPolicyRequest
import aws.sdk.kotlin.services.ecr.model.StartLifecyclePolicyPreviewRequest
import com.github.dockerjava.api.DockerClient
import com.github.dockerjava.api.command.DockerCmdExecFactory
import com.github.dockerjava.api.model.AuthConfig
import com.github.dockerjava.core.DockerClientBuilder
import com.github.dockerjava.netty.NettyDockerCmdExecFactory

```

```
import java.io.IOException
import java.util.Base64

class ECRActions {
 private var dockerClient: DockerClient? = null

 private fun getDockerClient(): DockerClient? {
 val osName = System.getProperty("os.name")
 if (osName.startsWith("Windows")) {
 // Make sure Docker Desktop is running.
 val dockerHost = "tcp://localhost:2375" // Use the Docker Desktop
default port.
 val dockerCmdExecFactory: DockerCmdExecFactory =
NettyDockerCmdExecFactory().withReadTimeout(20000).withConnectTimeout(20000)
 dockerClient =
DockerClientBuilder.getInstance(dockerHost).withDockerCmdExecFactory(dockerCmdExecFactory)
 } else {
 dockerClient = DockerClientBuilder.getInstance().build()
 }
 return dockerClient
 }

 /**
 * Sets the lifecycle policy for the specified repository.
 *
 * @param repoName the name of the repository for which to set the lifecycle
policy.
 */
 suspend fun setLifeCyclePolicy(repoName: String): String? {
 val polText =
 """
 {
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images older than 14 days",
 "selection": {
 "tagStatus": "any",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 14
 },
 },
],
 }
 """
 }
}
```

```

 "action": {
 "type": "expire"
 }
 }
}

"".trimIndent()
val lifecyclePolicyPreviewRequest =
 StartLifecyclePolicyPreviewRequest {
 lifecyclePolicyText = polText
 repositoryName = repoName
 }

// Execute the request asynchronously.
EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val response =
 ecrClient.startLifecyclePolicyPreview(lifecyclePolicyPreviewRequest)
 return response.lifecyclePolicyText
}
}

/**
 * Retrieves the repository URI for the specified repository name.
 *
 * @param repoName the name of the repository to retrieve the URI for.
 * @return the repository URI for the specified repository name.
 */
suspend fun getRepositoryURI(repoName: String?): String? {
 require(!(repoName == null || repoName.isEmpty())) { "Repository name
cannot be null or empty" }
 val request =
 DescribeRepositoriesRequest {
 repositoryNames = listOf(repoName)
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val describeRepositoriesResponse =
 ecrClient.describeRepositories(request)
 if (!describeRepositoriesResponse.repositories?.isEmpty()!!) {
 return
describeRepositoriesResponse?.repositories?.get(0)?.repositoryUri
 } else {

```

```
 println("No repositories found for the given name.")
 return ""
 }
}

/**
 * Retrieves the authorization token for Amazon Elastic Container Registry
 * (ECR).
 *
 */
suspend fun getAuthToken() {
 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 // Retrieve the authorization token for ECR.
 val response = ecrClient.getAuthorizationToken()
 val authorizationData = response.authorizationData?.get(0)
 val token = authorizationData?.authorizationToken
 if (token != null) {
 println("The token was successfully retrieved.")
 }
 }
}

/**
 * Gets the repository policy for the specified repository.
 *
 * @param repoName the name of the repository.
 */
suspend fun getRepoPolicy(repoName: String?): String? {
 require(!(repoName == null || repoName.isEmpty())) { "Repository name
cannot be null or empty" }

 // Create the request
 val getRepositoryPolicyRequest =
 GetRepositoryPolicyRequest {
 repositoryName = repoName
 }
 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val response =
ecrClient.getRepositoryPolicy(getRepositoryPolicyRequest)
 val responseText = response.policyText
 return responseText
 }
}
```

```
 }
 }

 /**
 * Sets the repository policy for the specified ECR repository.
 *
 * @param repoName the name of the ECR repository.
 * @param iamRole the IAM role to be granted access to the repository.
 */
 suspend fun setRepoPolicy(
 repoName: String?,
 iamRole: String?,
) {
 val policyDocumentTemplate =
 """
 {
 "Version": "2012-10-17",
 "Statement" : [{
 "Sid" : "new statement",
 "Effect" : "Allow",
 "Principal" : {
 "AWS" : "$iamRole"
 },
 "Action" : "ecr:BatchGetImage"
 }]
 }
 """.trimIndent()
 val setRepositoryPolicyRequest =
 SetRepositoryPolicyRequest {
 repositoryName = repoName
 policyText = policyDocumentTemplate
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val response =
 ecrClient.setRepositoryPolicy(setRepositoryPolicyRequest)
 if (response != null) {
 println("Repository policy set successfully.")
 }
 }
 }
}
```

```
/**
 * Creates an Amazon Elastic Container Registry (Amazon ECR) repository.
 *
 * @param repoName the name of the repository to create.
 * @return the Amazon Resource Name (ARN) of the created repository, or an
empty string if the operation failed.
 * @throws RepositoryAlreadyExistsException if the repository exists.
 * @throws EcrException if an error occurs while creating the
repository.
 */
suspend fun createECRRepository(repoName: String?): String? {
 val request =
 CreateRepositoryRequest {
 repositoryName = repoName
 }

 return try {
 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val response = ecrClient.createRepository(request)
 response.repository?.repositoryArn
 }
 } catch (e: RepositoryAlreadyExistsException) {
 println("Repository already exists: $repoName")
 repoName?.let { getRepoARN(it) }
 } catch (e: EcrException) {
 println("An error occurred: ${e.message}")
 null
 }
}

suspend fun getRepoARN(repoName: String): String? {
 // Fetch the existing repository's ARN.
 val describeRequest =
 DescribeRepositoriesRequest {
 repositoryNames = listOf(repoName)
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val describeResponse =
 ecrClient.describeRepositories(describeRequest)
 return describeResponse.repositories?.get(0)?.repositoryArn
 }
}
```

```
fun listLocalImages(): Boolean = try {
 val images = getDockerClient()?.listImagesCmd()?.exec()
 images?.any { image ->
 image.repoTags?.any { tag -> tag.startsWith("echo-text") } ?: false
 } ?: false
} catch (ex: Exception) {
 println("ERROR: ${ex.message}")
 false
}

/**
 * Pushes a Docker image to an Amazon Elastic Container Registry (ECR)
repository.
 *
 * @param repoName the name of the ECR repository to push the image to.
 * @param imageName the name of the Docker image.
 */
suspend fun pushDockerImage(
 repoName: String,
 imageName: String,
) {
 println("Pushing $imageName to $repoName will take a few seconds")
 val authConfig = getAuthConfig(repoName)

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val desRequest =
 DescribeRepositoriesRequest {
 repositoryNames = listOf(repoName)
 }

 val describeRepoResponse = ecrClient.describeRepositories(desRequest)
 val repoData =
 describeRepoResponse.repositories?.firstOrNull
 { it.repositoryName == repoName }
 ?: throw RuntimeException("Repository not found: $repoName")

 val tagImageCmd = getDockerClient()?.tagImageCmd("$imageName",
 "${repoData.repositoryUri}", imageName)
 if (tagImageCmd != null) {
 tagImageCmd.exec()
 }
 val pushImageCmd =
 repoData.repositoryUri?.let {
```

```

 dockerClient?.pushImageCmd(it)
 // ?.withTag("latest")
 ?.withAuthConfig(authConfig)
 }

 try {
 if (pushImageCmd != null) {
 pushImageCmd.start().awaitCompletion()
 }
 println("The $imageName was pushed to Amazon ECR")
 } catch (e: IOException) {
 throw RuntimeException(e)
 }
}

/**
 * Verifies the existence of an image in an Amazon Elastic Container Registry
 * (Amazon ECR) repository asynchronously.
 *
 * @param repositoryName The name of the Amazon ECR repository.
 * @param imageTag The tag of the image to verify.
 */
suspend fun verifyImage(
 repoName: String?,
 imageTagVal: String?,
) {
 require(!(repoName == null || repoName.isEmpty())) { "Repository name
cannot be null or empty" }
 require(!(imageTagVal == null || imageTagVal.isEmpty())) { "Image tag
cannot be null or empty" }

 val imageId =
 ImageIdentifier {
 imageTag = imageTagVal
 }
 val request =
 DescribeImagesRequest {
 repositoryName = repoName
 imageIds = listOf(imageId)
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->

```

```
 val describeImagesResponse = ecrClient.describeImages(request)
 if (describeImagesResponse != null && !
describeImagesResponse.imageDetails?.isEmpty()!!) {
 println("Image is present in the repository.")
 } else {
 println("Image is not present in the repository.")
 }
 }
}

/**
 * Deletes an ECR (Elastic Container Registry) repository.
 *
 * @param repoName the name of the repository to delete.
 */
suspend fun deleteECRRepository(repoName: String) {
 if (repoName.isNullOrEmpty()) {
 throw IllegalArgumentException("Repository name cannot be null or
empty")
 }

 val repositoryRequest =
 DeleteRepositoryRequest {
 force = true
 repositoryName = repoName
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 ecrClient.deleteRepository(repositoryRequest)
 println("You have successfully deleted the $repoName repository")
 }
}

// Return an AuthConfig.
private suspend fun getAuthConfig(repoName: String): AuthConfig {
 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 // Retrieve the authorization token for ECR.
 val response = ecrClient.getAuthorizationToken()
 val authorizationData = response.authorizationData?.get(0)
 val token = authorizationData?.authorizationToken
 val decodedToken = String(Base64.getDecoder().decode(token))
 val password = decodedToken.substring(4)
 }
}
```

```
 val request =
 DescribeRepositoriesRequest {
 repositoryNames = listOf(repoName)
 }

 val descrRepoResponse = ecrClient.describeRepositories(request)
 val repoData = descrRepoResponse.repositories?.firstOrNull
 { it.repositoryName == repoName }
 val registryURL: String =
 repoData?.repositoryUri?.split("/")?.get(0) ?: ""

 return AuthConfig()
 .withUsername("AWS")
 .withPassword(password)
 .withRegistryAddress(registryURL)
 }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Kotlin.
  - [CreateRepository](#)
  - [DeleteRepository](#)
  - [DescribeImages](#)
  - [DescribeRepositories](#)
  - [GetAuthorizationToken](#)
  - [GetRepositoryPolicy](#)
  - [SetRepositoryPolicy](#)
  - [StartLifecyclePolicyPreview](#)

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

Exécutez un scénario interactif à une invite de commande.

```
class ECRGettingStarted:
 """
 A scenario that demonstrates how to use Boto3 to perform basic operations
 using
 Amazon ECR.
 """

 def __init__(
 self,
 ecr_wrapper: ECRWrapper,
 docker_client: docker.DockerClient,
):
 self.ecr_wrapper = ecr_wrapper
 self.docker_client = docker_client
 self.tag = "echo-text"
 self.repository_name = "ecr-basics"
 self.docker_image = None
 self.full_tag_name = None
 self.repository = None

 def run(self, role_arn: str) -> None:
 """
 Runs the scenario.
 """
 print(
```

The Amazon Elastic Container Registry (ECR) is a fully-managed Docker container registry service provided by AWS. It allows developers and organizations to securely store, manage, and deploy Docker container images.

ECR provides a simple and scalable way to manage container images throughout their lifecycle, from building and testing to production deployment.

The `ECRWrapper` class is a wrapper for the Boto3 `ecr` client. The `ecr` client provides a set of methods to programmatically interact with the Amazon ECR service. This allows developers to automate the storage, retrieval, and management of container images as part of their application deployment pipelines. With ECR, teams can focus on building and deploying their applications without having to worry about the underlying infrastructure required to host and manage a container registry.

This scenario walks you through how to perform key operations for this service. Let's get started...

```

 """
)
 press_enter_to_continue()
 print_dashes()
 print(
 f"""
* Create an ECR repository.

```

An ECR repository is a private Docker container repository provided by Amazon Web Services (AWS). It is a managed service that makes it easy to store, manage, and deploy Docker container images.

```

 """
)
 print(f"Creating a repository named {self.repository_name}")
 self.repository =
self.ecr_wrapper.create_repository(self.repository_name)
 print(f"The ARN of the ECR repository is
{self.repository['repositoryArn']}")
 repository_uri = self.repository["repositoryUri"]
 press_enter_to_continue()
 print_dashes()

 print(
 f"""
* Build a Docker image.

```

Create a local Docker image if it does not already exist. A Python Docker client is used to execute Docker commands.

You must have Docker installed and running.

```
 """
)
 print(f"Building a docker image from 'docker_files/Dockerfile'")
 self.full_tag_name = f"{repository_uri}:{self.tag}"
 self.docker_image = self.docker_client.images.build(
 path="docker_files", tag=self.full_tag_name
)[0]
 print(f"Docker image {self.full_tag_name} successfully built.")
 press_enter_to_continue()
 print_dashes()

 if role_arn is None:
 print(
 """
* Because an IAM role ARN was not provided, a role policy will not be set for
this repository.
 """
)
 else:
 print(
 """
* Set an ECR repository policy.

Setting an ECR repository policy using the `setRepositoryPolicy` function is
crucial for maintaining
the security and integrity of your container images. The repository policy allows
you to
define specific rules and restrictions for accessing and managing the images
stored within your ECR
repository.
 """
)

 self.grant_role_download_access(role_arn)
 print(f"Download access granted to the IAM role ARN {role_arn}")
 press_enter_to_continue()
 print_dashes()

 print(
 """
* Display ECR repository policy.
 """
)
```

Now we will retrieve the ECR policy to ensure it was successfully set.

```
 """
)

 policy_text =
self.ecr_wrapper.get_repository_policy(self.repository_name)
 print("Policy Text:")
 print(f"{policy_text}")
 press_enter_to_continue()
 print_dashes()
```

```
 print(
 """
```

\* Retrieve an ECR authorization token.

You need an authorization token to securely access and interact with the Amazon ECR registry.

The `get_authorization_token` method of the `ecr` client is responsible for securely accessing

and interacting with an Amazon ECR repository. This operation is responsible for obtaining a

valid authorization token, which is required to authenticate your requests to the ECR service.

Without a valid authorization token, you would not be able to perform any operations on the

ECR repository, such as pushing, pulling, or managing your Docker images.

```
 """
)
```

```
 authorization_token = self.ecr_wrapper.get_authorization_token()
 print("Authorization token retrieved.")
 press_enter_to_continue()
 print_dashes()
 print(
 """
```

\* Get the ECR Repository URI.

The URI of an Amazon ECR repository is important. When you want to deploy a container image to

a container orchestration platform like Amazon Elastic Kubernetes Service (EKS) or Amazon Elastic Container Service (ECS), you need to specify the full image URI,

which includes the ECR repository URI. This allows the container runtime to pull the

```
correct container image from the ECR repository.
 """
)
 repository_descriptions = self.ecr_wrapper.describe_repositories(
 [self.repository_name]
)
 repository_uri = repository_descriptions[0]["repositoryUri"]
 print(f"Repository URI found: {repository_uri}")
 press_enter_to_continue()
 print_dashes()
```

```
print(
 """
```

\* Set an ECR Lifecycle Policy.

An ECR Lifecycle Policy is used to manage the lifecycle of Docker images stored in your ECR repositories.

These policies allow you to automatically remove old or unused Docker images from your repositories, freeing up storage space and reducing costs.

This example policy helps to maintain the size and efficiency of the container registry by automatically removing older and potentially unused images, ensuring that the storage is optimized and the registry remains up-to-date.

```
 """
)
 press_enter_to_continue()
 self.put_expiration_policy()
 print(f"An expiration policy was added to the repository.")
 print_dashes()
```

```
print(
 """
```

\* Push a docker image to the Amazon ECR Repository.

The Docker client uses the authorization token is used to authenticate the when pushing the image to the ECR repository.

```
 """
)
 decoded_authorization =
base64.b64decode(authorization_token).decode("utf-8")
 username, password = decoded_authorization.split(":")
```

```
resp = self.docker_client.api.push(
 repository=repository_uri,
 auth_config={"username": username, "password": password},
 tag=self.tag,
 stream=True,
 decode=True,
)
for line in resp:
 print(line)

print_dashes()

print("* Verify if the image is in the ECR Repository.")
image_descriptions = self.ecr_wrapper.describe_images(
 self.repository_name, [self.tag]
)
if len(image_descriptions) > 0:
 print("Image found in ECR Repository.")
else:
 print("Image not found in ECR Repository.")
press_enter_to_continue()
print_dashes()

print(
 "* As an optional step, you can interact with the image in Amazon ECR
 by using the CLI."
)
if q.ask(
 "Would you like to view instructions on how to use the CLI to run the
 image? (y/n)",
 q.is_yesno,
):
 print(
 f"""
1. Authenticate with ECR - Before you can pull the image from Amazon ECR, you
 need to authenticate with the registry. You can do this using the AWS CLI:

 aws ecr get-login-password --region us-east-1 | docker login --username AWS
 --password-stdin {repository_uri.split("/")[0]}

2. Describe the image using this command:
```

```
aws ecr describe-images --repository-name {self.repository_name} --image-ids
imageTag={self.tag}
```

3. Run the Docker container and view the output using this command:

```
docker run --rm {self.full_tag_name}
"""
)

 self.cleanup(True)

def cleanup(self, ask: bool):
 """
 Deletes the resources created in this scenario.
 :param ask: If True, prompts the user to confirm before deleting the
resources.
 """
 if self.repository is not None and (
 not ask
 or q.ask(
 f"Would you like to delete the ECR repository
'{self.repository_name}'? (y/n) "
)
):
 print(f"Deleting the ECR repository '{self.repository_name}'.")
 self.ecr_wrapper.delete_repository(self.repository_name)

 if self.full_tag_name is not None and (
 not ask
 or q.ask(
 f"Would you like to delete the local Docker image
'{self.full_tag_name}'? (y/n) "
)
):
 print(f"Deleting the docker image '{self.full_tag_name}'.")
 self.docker_client.images.remove(self.full_tag_name)

def grant_role_download_access(self, role_arn: str):
 """
 Grants the specified role access to download images from the ECR
repository.

 :param role_arn: The ARN of the role to grant access to.
 """
```

```
policy_json = {
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "AllowDownload",
 "Effect": "Allow",
 "Principal": {"AWS": role_arn},
 "Action": ["ecr:BatchGetImage"],
 }
],
}

self.ecr_wrapper.set_repository_policy(
 self.repository_name, json.dumps(policy_json)
)

def put_expiration_policy(self):
 """
 Puts an expiration policy on the ECR repository.
 """
 policy_json = {
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images older than 14 days",
 "selection": {
 "tagStatus": "any",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 14,
 },
 "action": {"type": "expire"},
 }
]
 }

 self.ecr_wrapper.put_lifecycle_policy(
 self.repository_name, json.dumps(policy_json)
)

if __name__ == "__main__":
```

```
parser = argparse.ArgumentParser(
 description="Run Amazon ECR getting started scenario."
)
parser.add_argument(
 "--iam-role-arn",
 type=str,
 default=None,
 help="an optional IAM role ARN that will be granted access to download
images from a repository.",
 required=False,
)
parser.add_argument(
 "--no-art",
 action="store_true",
 help="accessibility setting that suppresses art in the console output.",
)
args = parser.parse_args()
no_art = args.no_art
iam_role_arn = args.iam_role_arn
demo = None
a_docker_client = None
try:
 a_docker_client = docker.from_env()
 if not a_docker_client.ping():
 raise docker.errors.DockerException("Docker is not running.")
except docker.errors.DockerException as err:
 logging.error(
 """
 The Python Docker client could not be created.
 Do you have Docker installed and running?
 Here is the error message:
 %s
 """,
 err,
)
 sys.exit("Error with Docker.")
try:
 an_ecr_wrapper = ECRWrapper.from_client()
 demo = ECRGettingStarted(an_ecr_wrapper, a_docker_client)
 demo.run(iam_role_arn)
except Exception as exception:
 logging.exception("Something went wrong with the demo!")
 if demo is not None:
```

```
demo.cleanup(False)
```

Classe ECRWrapper qui encapsule les actions Amazon ECR.

```
class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

 def create_repository(self, repository_name: str) -> dict[str, any]:
 """
 Creates an ECR repository.

 :param repository_name: The name of the repository to create.
 :return: A dictionary of the created repository.
 """
 try:
 response =
self.ecr_client.create_repository(repositoryName=repository_name)
 return response["repository"]
 except ClientError as err:
 if err.response["Error"]["Code"] ==
"RepositoryAlreadyExistsException":
 print(f"Repository {repository_name} already exists.")
 response = self.ecr_client.describe_repositories(
 repositoryNames=[repository_name]
)
 return self.describe_repositories([repository_name])[0]
 else:
 logger.error(
```

```
 "Error creating repository %s. Here's why %s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise

def delete_repository(self, repository_name: str):
 """
 Deletes an ECR repository.

 :param repository_name: The name of the repository to delete.
 """
 try:
 self.ecr_client.delete_repository(
 repositoryName=repository_name, force=True
)
 print(f"Deleted repository {repository_name}.")
 except ClientError as err:
 logger.error(
 "Couldn't delete repository %s.. Here's why %s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise

def set_repository_policy(self, repository_name: str, policy_text: str):
 """
 Sets the policy for an ECR repository.

 :param repository_name: The name of the repository to set the policy for.
 :param policy_text: The policy text to set.
 """
 try:
 self.ecr_client.set_repository_policy(
 repositoryName=repository_name, policyText=policy_text
)
 print(f"Set repository policy for repository {repository_name}.")
 except ClientError as err:
 if err.response["Error"]["Code"] ==
"RepositoryPolicyNotFoundException":
 logger.error("Repository does not exist. %s.", repository_name)
 raise
```

```
 else:
 logger.error(
 "Couldn't set repository policy for repository %s. Here's why
%s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise

def get_repository_policy(self, repository_name: str) -> str:
 """
 Gets the policy for an ECR repository.

 :param repository_name: The name of the repository to get the policy for.
 :return: The policy text.
 """
 try:
 response = self.ecr_client.get_repository_policy(
 repositoryName=repository_name
)
 return response["policyText"]
 except ClientError as err:
 if err.response["Error"]["Code"] ==
"RepositoryPolicyNotFoundException":
 logger.error("Repository does not exist. %s.", repository_name)
 raise
 else:
 logger.error(
 "Couldn't get repository policy for repository %s. Here's why
%s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise

def get_authorization_token(self) -> str:
 """
 Gets an authorization token for an ECR repository.

 :return: The authorization token.
 """
 try:
```

```
 response = self.ecr_client.get_authorization_token()
 return response["authorizationData"][0]["authorizationToken"]
 except ClientError as err:
 logger.error(
 "Couldn't get authorization token. Here's why %s",
 err.response["Error"]["Message"],
)
 raise

def describe_repositories(self, repository_names: list[str]) -> list[dict]:
 """
 Describes ECR repositories.

 :param repository_names: The names of the repositories to describe.
 :return: The list of repository descriptions.
 """
 try:
 response = self.ecr_client.describe_repositories(
 repositoryNames=repository_names
)
 return response["repositories"]
 except ClientError as err:
 logger.error(
 "Couldn't describe repositories. Here's why %s",
 err.response["Error"]["Message"],
)
 raise

def put_lifecycle_policy(self, repository_name: str, lifecycle_policy_text:
str):
 """
 Puts a lifecycle policy for an ECR repository.

 :param repository_name: The name of the repository to put the lifecycle
policy for.
 :param lifecycle_policy_text: The lifecycle policy text to put.
 """
 try:
 self.ecr_client.put_lifecycle_policy(
 repositoryName=repository_name,
 lifecyclePolicyText=lifecycle_policy_text,
)
```

```
 print(f"Put lifecycle policy for repository {repository_name}.")
 except ClientError as err:
 logger.error(
 "Couldn't put lifecycle policy for repository %s. Here's why %s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise

def describe_images(
 self, repository_name: str, image_ids: list[str] = None
) -> list[dict]:
 """
 Describes ECR images.

 :param repository_name: The name of the repository to describe images
for.
 :param image_ids: The optional IDs of images to describe.
 :return: The list of image descriptions.
 """
 try:
 params = {
 "repositoryName": repository_name,
 }
 if image_ids is not None:
 params["imageIds"] = [{"imageTag": tag} for tag in image_ids]

 paginator = self.ecr_client.get_paginator("describe_images")
 image_descriptions = []
 for page in paginator.paginate(**params):
 image_descriptions.extend(page["imageDetails"])
 return image_descriptions
 except ClientError as err:
 logger.error(
 "Couldn't describe images. Here's why %s",
 err.response["Error"]["Message"],
)
 raise
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - [CreateRepository](#)
  - [DeleteRepository](#)
  - [DescribeImages](#)
  - [DescribeRepositories](#)
  - [GetAuthorizationToken](#)
  - [GetRepositoryPolicy](#)
  - [SetRepositoryPolicy](#)
  - [StartLifecyclePolicyPreview](#)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Actions pour Amazon ECR à l'aide de kits SDK AWS

Les exemples de code suivants montrent comment effectuer des actions Amazon ECR individuelles avec des AWS SDK. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Ces extraits appellent l'API Amazon ECR et sont des extraits de code de programmes plus volumineux qui doivent être exécutés en contexte. Vous pouvez voir les actions dans leur contexte dans [Scénarios pour Amazon ECR utilisant des SDK AWS](#).

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, consultez la [Référence des API Amazon Elastic Container Registry](#).

### Exemples

- [Utilisation CreateRepository avec un AWS SDK ou une CLI](#)
- [Utilisation DeleteRepository avec un AWS SDK ou une CLI](#)
- [Utilisation DescribeImages avec un AWS SDK ou une CLI](#)
- [Utilisation DescribeRepositories avec un AWS SDK ou une CLI](#)
- [Utilisation GetAuthorizationToken avec un AWS SDK ou une CLI](#)
- [Utilisation GetRepositoryPolicy avec un AWS SDK ou une CLI](#)

- [Utilisation ListImages avec un AWS SDK ou une CLI](#)
- [Utilisation PushImageCmd avec un AWS SDK](#)
- [Utilisation PutLifecyclePolicy avec un AWS SDK ou une CLI](#)
- [Utilisation SetRepositoryPolicy avec un AWS SDK ou une CLI](#)
- [Utilisation StartLifecyclePolicyPreview avec un AWS SDK ou une CLI](#)

## Utilisation **CreateRepository** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `CreateRepository`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- [Principes de base](#)
- [Commencer à utiliser Amazon ECR](#)

### CLI

#### AWS CLI

Exemple 1 : pour créer un référentiel

L'exemple `create-repository` suivant crée un référentiel dans l'espace de noms spécifié dans le registre par défaut d'un compte.

```
aws ecr create-repository \
 --repository-name project-a/sample-repo
```

Sortie :

```
{
 "repository": {
 "registryId": "123456789012",
 "repositoryName": "project-a/sample-repo",
 "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/project-
a/sample-repo"
 }
}
```

```
}
```

Pour plus d'informations, consultez [Création d'un référentiel](#) dans le Guide de l'utilisateur Amazon ECR.

Exemple 2 : pour créer un référentiel configuré avec l'immutabilité des balises d'image

L'exemple `create-repository` suivant crée un référentiel configuré pour l'immutabilité des balises dans le registre par défaut d'un compte.

```
aws ecr create-repository \
 --repository-name project-a/sample-repo \
 --image-tag-mutability IMMUTABLE
```

Sortie :

```
{
 "repository": {
 "registryId": "123456789012",
 "repositoryName": "project-a/sample-repo",
 "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/project-a/sample-repo",
 "imageTagMutability": "IMMUTABLE"
 }
}
```

Pour plus d'informations, consultez [Caractère immuable des balises d'image](#) dans le Guide de l'utilisateur Amazon ECR.

Exemple 3 : pour créer un référentiel configuré avec une configuration d'analyse

L'exemple `create-repository` suivant crée un référentiel configuré pour effectuer une analyse de vulnérabilité d'une transmission d'image dans le registre par défaut d'un compte.

```
aws ecr create-repository \
 --repository-name project-a/sample-repo \
 --image-scanning-configuration scanOnPush=true
```

Sortie :

```
{
```

```
"repository": {
 "registryId": "123456789012",
 "repositoryName": "project-a/sample-repo",
 "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/project-
a/sample-repo",
 "imageScanningConfiguration": {
 "scanOnPush": true
 }
}
```

Pour plus d'informations, consultez [Numérisation d'images](#) dans le Guide de l'utilisateur Amazon ECR.

- Pour plus de détails sur l'API, reportez-vous [CreateRepository](#) à la section Référence des AWS CLI commandes.

## Java

### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Creates an Amazon Elastic Container Registry (Amazon ECR) repository.
 *
 * @param repoName the name of the repository to create.
 * @return the Amazon Resource Name (ARN) of the created repository, or an
empty string if the operation failed.
 * @throws IllegalArgumentException If repository name is invalid.
 * @throws RuntimeException if an error occurs while creating the
repository.
 */
public String createECRRepository(String repoName) {
 if (repoName == null || repoName.isEmpty()) {
 throw new IllegalArgumentException("Repository name cannot be null or
empty");
 }
}
```

```
 CreateRepositoryRequest request = CreateRepositoryRequest.builder()
 .repositoryName(repoName)
 .build();

 CompletableFuture<CreateRepositoryResponse> response =
getAsyncClient().createRepository(request);
 try {
 CreateRepositoryResponse result = response.join();
 if (result != null) {
 System.out.println("The " + repoName + " repository was created
successfully.");
 return result.repository().repositoryArn();
 } else {
 throw new RuntimeException("Unexpected response type");
 }
 } catch (CompletionException e) {
 Throwable cause = e.getCause();
 if (cause instanceof EcrException ex) {
 if
("RepositoryAlreadyExistsException".equals(ex.awsErrorDetails().errorCode())) {
 System.out.println("The Amazon ECR repository already exists,
moving on...");

 DescribeRepositoriesRequest describeRequest =
DescribeRepositoriesRequest.builder()
 .repositoryNames(repoName)
 .build();

 DescribeRepositoriesResponse describeResponse =
getAsyncClient().describeRepositories(describeRequest).join();
 return
describeResponse.repositories().get(0).repositoryArn();
 } else {
 throw new RuntimeException(ex);
 }
 } else {
 throw new RuntimeException(e);
 }
 }
 }
}
```

- Pour plus de détails sur l'API, reportez-vous [CreateRepository](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet [GitHub](#). Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Creates an Amazon Elastic Container Registry (Amazon ECR) repository.
 *
 * @param repoName the name of the repository to create.
 * @return the Amazon Resource Name (ARN) of the created repository, or an
empty string if the operation failed.
 * @throws RepositoryAlreadyExistsException if the repository exists.
 * @throws EcrException if an error occurs while creating the
repository.
 */
suspend fun createECRRepository(repoName: String?): String? {
 val request =
 CreateRepositoryRequest {
 repositoryName = repoName
 }

 return try {
 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val response = ecrClient.createRepository(request)
 response.repository?.repositoryArn
 }
 } catch (e: RepositoryAlreadyExistsException) {
 println("Repository already exists: $repoName")
 repoName?.let { getRepoARN(it) }
 } catch (e: EcrException) {
 println("An error occurred: ${e.message}")
 null
 }
}
```

- Pour plus de détails sur l'API, consultez [CreateRepository](#) la section AWS SDK pour la référence de l'API Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

 def create_repository(self, repository_name: str) -> dict[str, any]:
 """
 Creates an ECR repository.

 :param repository_name: The name of the repository to create.
 :return: A dictionary of the created repository.
 """
 try:
 response =
self.ecr_client.create_repository(repositoryName=repository_name)
 return response["repository"]
 except ClientError as err:
```

```

 if err.response["Error"]["Code"] ==
"RepositoryAlreadyExistsException":
 print(f"Repository {repository_name} already exists.")
 response = self.ecr_client.describe_repositories(
 repositoryNames=[repository_name]
)
 return self.describe_repositories([repository_name])[0]
 else:
 logger.error(
 "Error creating repository %s. Here's why %s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise

```

- Pour plus de détails sur l'API, consultez [CreateRepository](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

TRY.
 " iv_repository_name = 'my-repository'
 oo_result = lo_ecr->createrepository(
 iv_repositoryname = iv_repository_name).
 DATA(lv_repository_uri) = oo_result->get_repository()-
>get_repositoryuri().
 MESSAGE |Repository created with URI: { lv_repository_uri }| TYPE 'I'.
CATCH /aws1/cx_ecrrepositoryalrexex.
 " If repository already exists, retrieve it
 DATA lt_repo_names TYPE /aws1/
cl_ecrrepositorynamels00=>tt_repositorynamelist.

```

```
APPEND NEW /aws1/cl_ecrrepositorynames00(iv_value =
iv_repository_name) TO lt_repo_names.
DATA(lo_describe_result) = lo_ecr-
>describerepositories(it_repositorynames = lt_repo_names).
DATA(lt_repos) = lo_describe_result->get_repositories().
IF lines(lt_repos) > 0.
 READ TABLE lt_repos INDEX 1 INTO DATA(lo_repo).
 oo_result = NEW /aws1/cl_ecrcrerepositoryrsp(io_repository =
lo_repo).
 MESSAGE |Repository { iv_repository_name } already exists.| TYPE 'I'.
ENDIF.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous [CreateRepository](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **DeleteRepository** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteRepository`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- [Principes de base](#)
- [Commencer à utiliser Amazon ECR](#)

### CLI

#### AWS CLI

Pour supprimer un référentiel

L'exemple de commande `delete-repository` suivant supprime le référentiel spécifié dans le registre par défaut d'un compte. L'indicateur `--force` est obligatoire si le référentiel contient des images.

```
aws ecr delete-repository \
 --repository-name ubuntu \
 --force
```

Sortie :

```
{
 "repository": {
 "registryId": "123456789012",
 "repositoryName": "ubuntu",
 "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/ubuntu"
 }
}
```

Pour plus d'informations, consultez [Suppression d'un référentiel](#) dans le Guide de l'utilisateur Amazon ECR.

- Pour plus de détails sur l'API, reportez-vous [DeleteRepository](#) à la section Référence des AWS CLI commandes.

## Java

### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Deletes an ECR (Elastic Container Registry) repository.
 *
 * @param repoName the name of the repository to delete.
 * @throws IllegalArgumentException if the repository name is null or empty.
 * @throws EcrException if there is an error deleting the repository.
 * @throws RuntimeException if an unexpected error occurs during the deletion
 process.
 */
public void deleteECRRepository(String repoName) {
```

```
 if (repoName == null || repoName.isEmpty()) {
 throw new IllegalArgumentException("Repository name cannot be null or
empty");
 }

 DeleteRepositoryRequest repositoryRequest =
DeleteRepositoryRequest.builder()
 .force(true)
 .repositoryName(repoName)
 .build();

 CompletableFuture<DeleteRepositoryResponse> response =
getAsyncClient().deleteRepository(repositoryRequest);
 response.whenComplete((deleteRepositoryResponse, ex) -> {
 if (deleteRepositoryResponse != null) {
 System.out.println("You have successfully deleted the " +
repoName + " repository");
 } else {
 Throwable cause = ex.getCause();
 if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
 }
 }
 });

 // Wait for the CompletableFuture to complete
 response.join();
 }
}
```

- Pour plus de détails sur l'API, reportez-vous [DeleteRepository](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Deletes an ECR (Elastic Container Registry) repository.
 *
 * @param repoName the name of the repository to delete.
 */
suspend fun deleteECRRepository(repoName: String) {
 if (repoName.isNullOrEmpty()) {
 throw IllegalArgumentException("Repository name cannot be null or empty")
 }

 val repositoryRequest =
 DeleteRepositoryRequest {
 force = true
 repositoryName = repoName
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 ecrClient.deleteRepository(repositoryRequest)
 println("You have successfully deleted the $repoName repository")
 }
}
```

- Pour plus de détails sur l'API, consultez [DeleteRepository](#) la section AWS SDK pour la référence de l'API Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

 def delete_repository(self, repository_name: str):
 """
 Deletes an ECR repository.

 :param repository_name: The name of the repository to delete.
 """
 try:
 self.ecr_client.delete_repository(
 repositoryName=repository_name, force=True
)
 print(f"Deleted repository {repository_name}.")
 except ClientError as err:
 logger.error(
 "Couldn't delete repository %s.. Here's why %s",
 repository_name,
 err.response["Error"]["Message"],
```

```
)
raise
```

- Pour plus de détails sur l'API, consultez [DeleteRepository](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
TRY.
 " iv_repository_name = 'my-repository'
 lo_ecr->deleterepository(
 iv_repositoryname = iv_repository_name
 iv_force = abap_true).
 MESSAGE |Repository { iv_repository_name } deleted.| TYPE 'I'.
CATCH /aws1/cx_ecrrepositorynotfound.
 MESSAGE 'Repository not found.' TYPE 'I'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous [DeleteRepository](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **DescribeImages** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `DescribeImages`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- [Principes de base](#)

## CLI

### AWS CLI

Pour décrire une image dans un référentiel

L'`describe-images` exemple suivant affiche les détails d'une image dans le `cluster-autoscaler` référentiel avec la balise `v1.13.6`.

```
aws ecr describe-images \
 --repository-name cluster-autoscaler \
 --image-ids imageTag=v1.13.6
```


Sortie :

```
{
 "imageDetails": [
 {
 "registryId": "012345678910",
 "repositoryName": "cluster-autoscaler",
 "imageDigest":
"sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",
 "imageTags": [
 "v1.13.6"
],
 "imageSizeInBytes": 48318255,
 "imagePushedAt": 1565128275.0
 }
]
}
```

- Pour plus de détails sur l'API, reportez-vous [DescriberImages](#) à la section Référence des AWS CLI commandes.

## Java

## SDK pour Java 2.x

 Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Verifies the existence of an image in an Amazon Elastic Container Registry
 (Amazon ECR) repository asynchronously.
 *
 * @param repositoryName The name of the Amazon ECR repository.
 * @param imageTag The tag of the image to verify.
 * @throws EcrException if there is an error retrieving the image
 information from Amazon ECR.
 * @throws CompletionException if the asynchronous operation completes
 exceptionally.
 */
public void verifyImage(String repositoryName, String imageTag) {
 DescribeImagesRequest request = DescribeImagesRequest.builder()
 .repositoryName(repositoryName)
 .imageIds(ImageIdentifier.builder().imageTag(imageTag).build())
 .build();

 CompletableFuture<DescribeImagesResponse> response =
getAsyncClient().describeImages(request);
 response.whenComplete((describeImagesResponse, ex) -> {
 if (ex != null) {
 if (ex instanceof CompletionException) {
 Throwable cause = ex.getCause();
 if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
 }
 } else {
 throw new RuntimeException("Unexpected error: " +
ex.getCause());
 }
 }
 });
}
```

```
 }
 } else if (describeImagesResponse != null && !
describeImagesResponse.imageDetails().isEmpty()) {
 System.out.println("Image is present in the repository.");
 } else {
 System.out.println("Image is not present in the repository.");
 }
});

// Wait for the CompletableFuture to complete.
response.join();
}
```

- Pour plus de détails sur l'API, reportez-vous [DescribeImages](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Verifies the existence of an image in an Amazon Elastic Container Registry
 (Amazon ECR) repository asynchronously.
 *
 * @param repositoryName The name of the Amazon ECR repository.
 * @param imageTag The tag of the image to verify.
 */
suspend fun verifyImage(
 repoName: String?,
 imageTagVal: String?,
) {
 require(!(repoName == null || repoName.isEmpty())) { "Repository name
cannot be null or empty" }
```

```
require(!(imageTagVal == null || imageTagVal.isEmpty())) { "Image tag
cannot be null or empty" }

val imageId =
 ImageIdentifier {
 imageTag = imageTagVal
 }
val request =
 DescribeImagesRequest {
 repositoryName = repoName
 imageIds = listOf(imageId)
 }

EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val describeImagesResponse = ecrClient.describeImages(request)
 if (describeImagesResponse != null && !
describeImagesResponse.imageDetails?.isEmpty()!!) {
 println("Image is present in the repository.")
 } else {
 println("Image is not present in the repository.")
 }
}
}
```

- Pour plus de détails sur l'API, consultez [DescribeImages](#) la section AWS SDK pour la référence de l'API Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client
```

```
@classmethod
def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

def describe_images(
 self, repository_name: str, image_ids: list[str] = None
) -> list[dict]:
 """
 Describes ECR images.

 :param repository_name: The name of the repository to describe images
 for.
 :param image_ids: The optional IDs of images to describe.
 :return: The list of image descriptions.
 """
 try:
 params = {
 "repositoryName": repository_name,
 }
 if image_ids is not None:
 params["imageIds"] = [{"imageTag": tag} for tag in image_ids]

 paginator = self.ecr_client.get_paginator("describe_images")
 image_descriptions = []
 for page in paginator.paginate(**params):
 image_descriptions.extend(page["imageDetails"])
 return image_descriptions
 except ClientError as err:
 logger.error(
 "Couldn't describe images. Here's why %s",
 err.response["Error"]["Message"],
)
 raise
```

- Pour plus de détails sur l'API, consultez [DescribeImages](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
TRY.
 " iv_repository_name = 'my-repository'
 " it_image_ids = VALUE #((NEW /aws1/cl_ecrimageidentifier(iv_imagetag
= 'latest')))
 IF it_image_ids IS NOT INITIAL.
 oo_result = lo_ecr->describeimages(
 iv_repositoryname = iv_repository_name
 it_imageids = it_image_ids).
 ELSE.
 oo_result = lo_ecr->describeimages(
 iv_repositoryname = iv_repository_name).
 ENDIF.
 DATA(lt_image_details) = oo_result->get_imagedetails().
 MESSAGE |Found { lines(lt_image_details) } images in repository.| TYPE
'I'.
 CATCH /aws1/cx_ecrrepositorynotfound.
 MESSAGE 'Repository not found.' TYPE 'I'.
 CATCH /aws1/cx_ecrimagenotfound.
 MESSAGE 'Image not found.' TYPE 'I'.
 CATCH /aws1/cx_ecrinvalidparameter.
 MESSAGE 'Invalid parameter provided.' TYPE 'I'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous [DescribeImages](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **DescribeRepositories** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `DescribeRepositories`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- [Principes de base](#)

### CLI

#### AWS CLI

Pour décrire les référentiels dans un registre

Cet exemple décrit les référentiels du registre par défaut d'un compte.

Commande :

```
aws ecr describe-repositories
```

Sortie :

```
{
 "repositories": [
 {
 "registryId": "012345678910",
 "repositoryName": "ubuntu",
 "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/
ubuntu"
 },
 {
 "registryId": "012345678910",
 "repositoryName": "test",
 "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/test"
 }
]
}
```

```
}
```

- Pour plus de détails sur l'API, reportez-vous [DescribeRepositories](#) à la section Référence des AWS CLI commandes.

## Java

### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Retrieves the repository URI for the specified repository name.
 *
 * @param repoName the name of the repository to retrieve the URI for.
 * @return the repository URI for the specified repository name.
 * @throws EcrException if there is an error retrieving the repository
information.
 * @throws CompletionException if the asynchronous operation completes
exceptionally.
 */
public void getRepositoryURI(String repoName) {
 DescribeRepositoriesRequest request =
DescribeRepositoriesRequest.builder()
 .repositoryNames(repoName)
 .build();

 CompletableFuture<DescribeRepositoriesResponse> response =
getAsyncClient().describeRepositories(request);
 response.whenComplete((describeRepositoriesResponse, ex) -> {
 if (ex != null) {
 Throwable cause = ex.getCause();
 if (cause instanceof InterruptedException) {
 Thread.currentThread().interrupt();
 String errorMessage = "Thread interrupted while waiting for
asynchronous operation: " + cause.getMessage();
 throw new RuntimeException(errorMessage, cause);
 }
 }
 });
}
```

```
 } else if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 String errorMessage = "Unexpected error: " +
cause.getMessage();
 throw new RuntimeException(errorMessage, cause);
 }
 } else {
 if (describeRepositoriesResponse != null) {
 if (!describeRepositoriesResponse.repositories().isEmpty()) {
 String repositoryUri =
describeRepositoriesResponse.repositories().get(0).repositoryUri();
 System.out.println("Repository URI found: " +
repositoryUri);
 } else {
 System.out.println("No repositories found for the given
name.");
 }
 } else {
 System.err.println("No response received from
describeRepositories.");
 }
 }
});
response.join();
}
```

- Pour plus de détails sur l'API, reportez-vous [DescribeRepositories](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Retrieves the repository URI for the specified repository name.
 *
 * @param repoName the name of the repository to retrieve the URI for.
 * @return the repository URI for the specified repository name.
 */
suspend fun getRepositoryURI(repoName: String?): String? {
 require(!(repoName == null || repoName.isEmpty())) { "Repository name
cannot be null or empty" }
 val request =
 DescribeRepositoriesRequest {
 repositoryNames = listOf(repoName)
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val describeRepositoriesResponse =
ecrClient.describeRepositories(request)
 if (!describeRepositoriesResponse.repositories?.isEmpty()!!) {
 return
describeRepositoriesResponse?.repositories?.get(0)?.repositoryUri
 } else {
 println("No repositories found for the given name.")
 return ""
 }
 }
}
```

- Pour plus de détails sur l'API, consultez [DescribeRepositories](#) la section AWS SDK pour la référence de l'API Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

 def describe_repositories(self, repository_names: list[str]) -> list[dict]:
 """
 Describes ECR repositories.

 :param repository_names: The names of the repositories to describe.
 :return: The list of repository descriptions.
 """
 try:
 response = self.ecr_client.describe_repositories(
 repositoryNames=repository_names
)
 return response["repositories"]
 except ClientError as err:
 logger.error(
 "Couldn't describe repositories. Here's why %s",
 err.response["Error"]["Message"],
)
 raise
```

- Pour plus de détails sur l'API, consultez [DescribeRepositories](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## Rust

### SDK pour Rust

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
async fn show_repos(client: &aws_sdk_ecr::Client) -> Result<(),
aws_sdk_ecr::Error> {
 let rsp = client.describe_repositories().send().await?;

 let repos = rsp.repositories();

 println!("Found {} repositories:", repos.len());

 for repo in repos {
 println!(" ARN: {}", repo.repository_arn().unwrap());
 println!(" Name: {}", repo.repository_name().unwrap());
 }

 Ok(())
}
```

- Pour plus de détails sur l'API, voir [DescribeRepositories](#) la section de référence de l'API AWS SDK for Rust.

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
TRY.
 " it_repository_names = VALUE #((NEW /aws1/
cl_ecrrepositorynames00(iv_value = 'my-repository')))
 oo_result = lo_ecr->describerepositories(
 it_repository_names = it_repository_names).
 DATA(lt_repositories) = oo_result->get_repositories().
 MESSAGE |Found { lines(lt_repositories) } repositories.| TYPE 'I'.
CATCH /aws1/cx_ecrrepositorynotfound.
 MESSAGE 'Repository not found.' TYPE 'I'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous [DescribeRepositories](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **GetAuthorizationToken** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetAuthorizationToken`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- [Principes de base](#)

### CLI

#### AWS CLI

Pour obtenir un jeton d'autorisation pour votre registre par défaut

L'exemple de commande `get-authorization-token` suivant obtient un jeton d'autorisation pour votre registre par défaut.

```
aws ecr get-authorization-token
```

## Sortie :

```
{
 "authorizationData": [
 {
 "authorizationToken": "QVdT0kN...",
 "expiresAt": 1448875853.241,
 "proxyEndpoint": "https://123456789012.dkr.ecr.us-
west-2.amazonaws.com"
 }
]
}
```

- Pour plus de détails sur l'API, reportez-vous [GetAuthorizationToken](#) à la section Référence des AWS CLI commandes.

## Java

### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Retrieves the authorization token for Amazon Elastic Container Registry
 (ECR).
 * This method makes an asynchronous call to the ECR client to retrieve the
 authorization token.
 * If the operation is successful, the method prints the token to the
 console.
 * If an exception occurs, the method handles the exception and prints the
 error message.
 *
 * @throws EcrException if there is an error retrieving the authorization
 token from ECR.
 * @throws RuntimeException if there is an unexpected error during the
 operation.
 */
```

```
public void getAuthToken() {
 CompletableFuture<GetAuthorizationTokenResponse> response =
getAsyncClient().getAuthorizationToken();
 response.whenComplete((authorizationTokenResponse, ex) -> {
 if (authorizationTokenResponse != null) {
 AuthorizationData authorizationData =
authorizationTokenResponse.authorizationData().get(0);
 String token = authorizationData.authorizationToken();
 if (!token.isEmpty()) {
 System.out.println("The token was successfully retrieved.");
 }
 } else {
 if (ex.getCause() instanceof EcrException) {
 throw (EcrException) ex.getCause();
 } else {
 String errorMessage = "Unexpected error occurred: " +
ex.getMessage();
 throw new RuntimeException(errorMessage, ex); // Rethrow the
exception
 }
 }
 });
 response.join();
}
```

- Pour plus de détails sur l'API, reportez-vous [GetAuthorizationToken](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
```

```

 * Retrieves the authorization token for Amazon Elastic Container Registry
 (ECR).
 *
 */
suspend fun getAuthToken() {
 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 // Retrieve the authorization token for ECR.
 val response = ecrClient.getAuthorizationToken()
 val authorizationData = response.authorizationData?.get(0)
 val token = authorizationData?.authorizationToken
 if (token != null) {
 println("The token was successfully retrieved.")
 }
 }
}
}

```

- Pour plus de détails sur l'API, consultez [GetAuthorizationToken](#) la section AWS SDK pour la référence de l'API Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.

```

```
"""
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

def get_authorization_token(self) -> str:
 """
 Gets an authorization token for an ECR repository.

 :return: The authorization token.
 """
 try:
 response = self.ecr_client.get_authorization_token()
 return response["authorizationData"][0]["authorizationToken"]
 except ClientError as err:
 logger.error(
 "Couldn't get authorization token. Here's why %s",
 err.response["Error"]["Message"],
)
 raise
```

- Pour plus de détails sur l'API, consultez [GetAuthorizationToken](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

TRY.

```
oo_result = lo_ecr->getauthorizationtoken().
DATA(lt_auth_data) = oo_result->get_authorizationdata().
IF lines(lt_auth_data) > 0.
 READ TABLE lt_auth_data INDEX 1 INTO DATA(lo_auth_data).
```

```
DATA(lv_token) = lo_auth_data->get_authorizationtoken().
MESSAGE 'Authorization token retrieved.' TYPE 'I'.
ENDIF.
CATCH /aws1/cx_ecrserverexception.
MESSAGE 'Server exception occurred.' TYPE 'I'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous [GetAuthorizationToken](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **GetRepositoryPolicy** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetRepositoryPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- [Principes de base](#)

### CLI

#### AWS CLI

Pou extraire la politique d'un référentiel spécifié

L'exemple `get-repository-policy` suivant affiche des informations sur la politique de référentiel du référentiel `cluster-autoscaler`.

```
aws ecr get-repository-policy \
 --repository-name cluster-autoscaler
```

Sortie :

```
{
 "registryId": "012345678910",
```

```

"repositoryName": "cluster-autoscaler",
"policyText": "{\n \"Version\" : \"2008-10-17\",\n \"Statement\" :\n [\n {\n \"Sid\" : \"allow public pull\",\n \"Effect\" : \"Allow\",\n \"Principal\" : \"*\",\n \"Action\" : [\"ecr:BatchCheckLayerAvailability\",\n \"ecr:BatchGetImage\", \"ecr:GetDownloadUrlForLayer\"]\n }\n]\n}"
}

```

- Pour plus de détails sur l'API, reportez-vous [GetRepositoryPolicy](#) à la section Référence des AWS CLI commandes.

## Java

### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

/**
 * Gets the repository policy for the specified repository.
 *
 * @param repoName the name of the repository.
 * @throws EcrException if an AWS error occurs while getting the repository
 * policy.
 */
public String getRepoPolicy(String repoName) {
 if (repoName == null || repoName.isEmpty()) {
 throw new IllegalArgumentException("Repository name cannot be null or
empty");
 }

 GetRepositoryPolicyRequest getRepositoryPolicyRequest =
GetRepositoryPolicyRequest.builder()
 .repositoryName(repoName)
 .build();

 CompletableFuture<GetRepositoryPolicyResponse> response =
getAsyncClient().getRepositoryPolicy(getRepositoryPolicyRequest);
 response.whenComplete((resp, ex) -> {

```

```
 if (resp != null) {
 System.out.println("Repository policy retrieved successfully.");
 } else {
 if (ex.getCause() instanceof EcrException) {
 throw (EcrException) ex.getCause();
 } else {
 String errorMessage = "Unexpected error occurred: " +
ex.getMessage();
 throw new RuntimeException(errorMessage, ex);
 }
 }
 });

 GetRepositoryPolicyResponse result = response.join();
 return result != null ? result.policyText() : null;
}
```

- Pour plus de détails sur l'API, reportez-vous [GetRepositoryPolicy](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Gets the repository policy for the specified repository.
 *
 * @param repoName the name of the repository.
 */
suspend fun getRepoPolicy(repoName: String?): String? {
 require(!(repoName == null || repoName.isEmpty())) { "Repository name
cannot be null or empty" }
}
```

```

// Create the request
val getRepositoryPolicyRequest =
 GetRepositoryPolicyRequest {
 repositoryName = repoName
 }
EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val response =
ecrClient.getRepositoryPolicy(getRepositoryPolicyRequest)
 val responseText = response.policyText
 return responseText
}
}

```

- Pour plus de détails sur l'API, consultez [GetRepositoryPolicy](#) la section AWS SDK pour la référence de l'API Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

```

```
def get_repository_policy(self, repository_name: str) -> str:
 """
 Gets the policy for an ECR repository.

 :param repository_name: The name of the repository to get the policy for.
 :return: The policy text.
 """
 try:
 response = self.ecr_client.get_repository_policy(
 repositoryName=repository_name
)
 return response["policyText"]
 except ClientError as err:
 if err.response["Error"]["Code"] ==
"RepositoryPolicyNotFoundException":
 logger.error("Repository does not exist. %s.", repository_name)
 raise
 else:
 logger.error(
 "Couldn't get repository policy for repository %s. Here's why
%s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise
```

- Pour plus de détails sur l'API, consultez [GetRepositoryPolicy](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
TRY.
 " iv_repository_name = 'my-repository'
 oo_result = lo_ecr->getrepositorypolicy(
 iv_repositoryname = iv_repository_name).
 DATA(lv_policy_text) = oo_result->get_policytext().
 MESSAGE 'Repository policy retrieved.' TYPE 'I'.
CATCH /aws1/cx_ecrrepositorynotfound.
 MESSAGE 'Repository not found.' TYPE 'I'.
CATCH /aws1/cx_ecrrepositoryplynot00.
 MESSAGE 'Repository policy not found.' TYPE 'I'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous [GetRepositoryPolicy](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **ListImages** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `ListImages`.

### CLI

#### AWS CLI

Pour répertorier les images dans un référentiel

L'exemple `list-images` suivant affiche la liste des images du référentiel `cluster-autoscaler`.

```
aws ecr list-images \
 --repository-name cluster-autoscaler
```

Sortie :

```
{
 "imageIds": [
 {
```

```
 "imageDigest":
"sha256:99c6fb4377e9a420a1eb3b410a951c9f464eff3b7dbc76c65e434e39b94b6570",
 "imageTag": "v1.13.8"
 },
 {
 "imageDigest":
"sha256:99c6fb4377e9a420a1eb3b410a951c9f464eff3b7dbc76c65e434e39b94b6570",
 "imageTag": "v1.13.7"
 },
 {
 "imageDigest":
"sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",
 "imageTag": "v1.13.6"
 }
]
}
```

- Pour plus de détails sur l'API, reportez-vous [ListImages](#) à la section Référence des AWS CLI commandes.

## Rust

### SDK pour Rust

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
async fn show_images(
 client: &aws_sdk_ecr::Client,
 repository: &str,
) -> Result<(), aws_sdk_ecr::Error> {
 let rsp = client
 .list_images()
 .repository_name(repository)
 .send()
 .await?;

 let images = rsp.image_ids();
```

```
println!("found {} images", images.len());

for image in images {
 println!(
 "image: {}:{}",
 image.image_tag().unwrap(),
 image.image_digest().unwrap()
);
}

Ok(())
}
```

- Pour plus de détails sur l'API, voir [ListImages](#) la section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **PushImageCmd** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `PushImageCmd`.

### Java

#### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Pushes a Docker image to an Amazon Elastic Container Registry (ECR)
 * repository.
 *
 * @param repoName the name of the ECR repository to push the image to.
```

```

 * @param imageName the name of the Docker image.
 */
 public void pushDockerImage(String repoName, String imageName) {
 System.out.println("Pushing " + imageName + " to Amazon ECR will take a
few seconds.");
 CompletableFuture<AuthConfig> authResponseFuture =
getAsyncClient().getAuthorizationToken()
 .thenApply(response -> {
 String token =
response.authorizationData().get(0).authorizationToken();
 String decodedToken = new
String(Base64.getDecoder().decode(token));
 String password = decodedToken.substring(4);

 DescribeRepositoriesResponse descrRepoResponse =
getAsyncClient().describeRepositories(b -> b.repositoryNames(repoName)).join();
 Repository repoData =
descrRepoResponse.repositories().stream().filter(r ->
r.repositoryName().equals(repoName)).findFirst().orElse(null);
 assert repoData != null;
 String registryURL = repoData.repositoryUri().split("/")[0];

 AuthConfig authConfig = new AuthConfig()
 .withUsername("AWS")
 .withPassword(password)
 .withRegistryAddress(registryURL);
 return authConfig;
 })
 .thenCompose(authConfig -> {
 DescribeRepositoriesResponse descrRepoResponse =
getAsyncClient().describeRepositories(b -> b.repositoryNames(repoName)).join();
 Repository repoData =
descrRepoResponse.repositories().stream().filter(r ->
r.repositoryName().equals(repoName)).findFirst().orElse(null);
 getDockerClient().tagImageCmd(imageName + ":latest",
repoData.repositoryUri() + ":latest", imageName).exec();
 try {

getDockerClient().pushImageCmd(repoData.repositoryUri()).withTag("echo-
text").withAuthConfig(authConfig).start().awaitCompletion();
 System.out.println("The " + imageName + " was pushed to
ECR");

 } catch (InterruptedException e) {

```

```

 throw (RuntimeException) e.getCause();
 }
 return CompletableFuture.completedFuture(authConfig);
});

authResponseFuture.join();
}

```

- Pour plus de détails sur l'API, reportez-vous [PushImageCmd](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

/**
 * Pushes a Docker image to an Amazon Elastic Container Registry (ECR)
 repository.
 *
 * @param repoName the name of the ECR repository to push the image to.
 * @param imageName the name of the Docker image.
 */
suspend fun pushDockerImage(
 repoName: String,
 imageName: String,
) {
 println("Pushing $imageName to $repoName will take a few seconds")
 val authConfig = getAuthConfig(repoName)

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val describeRequest =
 DescribeRepositoriesRequest {
 repositoryNames = listOf(repoName)
 }
 }
}

```

```

 }

 val describeRepoResponse = ecrClient.describeRepositories(desRequest)
 val repoData =
 describeRepoResponse.repositories?.firstOrNull
 { it.repositoryName == repoName }
 ?: throw RuntimeException("Repository not found: $repoName")

 val tagImageCmd = getDockerClient()?.tagImageCmd("$imageName",
 "${repoData.repositoryUri}", imageName)
 if (tagImageCmd != null) {
 tagImageCmd.exec()
 }
 val pushImageCmd =
 repoData.repositoryUri?.let {
 dockerClient?.pushImageCmd(it)
 // ?.withTag("latest")
 ?.withAuthConfig(authConfig)
 }

 try {
 if (pushImageCmd != null) {
 pushImageCmd.start().awaitCompletion()
 }
 println("The $imageName was pushed to Amazon ECR")
 } catch (e: IOException) {
 throw RuntimeException(e)
 }
}
}

```

- Pour plus de détails sur l'API, consultez [PushImageCmd](#) la section AWS SDK pour la référence de l'API Kotlin.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **PutLifecyclePolicy** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser **PutLifecyclePolicy**.

## CLI

## AWS CLI

Pour créer une politique de cycle de vie

L'exemple `put-lifecycle-policy` suivant crée une politique de cycle de vie pour le référentiel spécifié dans le registre par défaut d'un compte.

```
aws ecr put-lifecycle-policy \
 --repository-name "project-a/amazon-ecs-sample" \
 --lifecycle-policy-text "file://policy.json"
```

Contenu de `policy.json` :

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images older than 14 days",
 "selection": {
 "tagStatus": "untagged",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 14
 },
 "action": {
 "type": "expire"
 }
 }
]
}
```

Sortie :

```
{
 "registryId": "<aws_account_id>",
 "repositoryName": "project-a/amazon-ecs-sample",
 "lifecyclePolicyText": "{\"rules\": [{\"rulePriority\":1,\"description\":
 \"Expire images older than 14 days\", \"selection\": {\"tagStatus\": \"untagged\",
 \"countType\": \"sinceImagePushed\", \"countUnit\": \"days\", \"countNumber\":14},
 \"action\": {\"type\": \"expire\"}}]}"
```

```
}
```

Pour plus d'informations, consultez [Politiques de cycle de vie](#) dans le Guide de l'utilisateur Amazon ECR.

- Pour plus de détails sur l'API, reportez-vous [PutLifecyclePolicy](#) à la section Référence des AWS CLI commandes.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

 def put_lifecycle_policy(self, repository_name: str, lifecycle_policy_text:
str):
 """
 Puts a lifecycle policy for an ECR repository.

 :param repository_name: The name of the repository to put the lifecycle
 policy for.
 :param lifecycle_policy_text: The lifecycle policy text to put.
```

```
"""
try:
 self.ecr_client.put_lifecycle_policy(
 repositoryName=repository_name,
 lifecyclePolicyText=lifecycle_policy_text,
)
 print(f"Put lifecycle policy for repository {repository_name}.")
except ClientError as err:
 logger.error(
 "Couldn't put lifecycle policy for repository %s. Here's why %s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise
```

Exemple qui définit une politique de date d'expiration.

```
def put_expiration_policy(self):
 """
 Puts an expiration policy on the ECR repository.
 """
 policy_json = {
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images older than 14 days",
 "selection": {
 "tagStatus": "any",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 14,
 },
 "action": {"type": "expire"},
 }
]
 }

 self.ecr_wrapper.put_lifecycle_policy(
 self.repository_name, json.dumps(policy_json)
)
```

- Pour plus de détails sur l'API, consultez [PutLifecyclePolicy](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
TRY.
 " iv_repository_name = 'my-repository'
 " iv_lifecycle_policy_text = '{"rules":
[{"rulePriority":1,"description":"Expire images older than 14 days",...}]}'
 lo_ecr->putlifecyclepolicy(
 iv_repositoryname = iv_repository_name
 iv_lifecyclepolicytext = iv_lifecycle_policy_text).
 MESSAGE |Lifecycle policy set for repository { iv_repository_name }.|
TYPE 'I'.
CATCH /aws1/cx_ecrrepositorynotfound.
 MESSAGE 'Repository not found.' TYPE 'I'.
CATCH /aws1/cx_ecrvalidationex.
 MESSAGE 'Invalid lifecycle policy format.' TYPE 'I'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous [PutLifecyclePolicy](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **SetRepositoryPolicy** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `SetRepositoryPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- [Principes de base](#)

### CLI

#### AWS CLI

Pour définir la politique de référentiel d'un référentiel

L'exemple `set-repository-policy` suivant joint au référentiel `cluster-autoscaler` une politique de référentiel contenue dans un fichier.

```
aws ecr set-repository-policy \
 --repository-name cluster-autoscaler \
 --policy-text file://my-policy.json
```

Contenu de `my-policy.json` :

```
{
 "Version": "2012-10-17",
 "Statement" : [
 {
 "Sid" : "allow public pull",
 "Effect" : "Allow",
 "Principal" : "*",
 "Action" : [
 "ecr:BatchCheckLayerAvailability",
 "ecr:BatchGetImage",
 "ecr:GetDownloadUrlForLayer"
]
 }
]
}
```

Sortie :

```
{
 "registryId": "012345678910",
 "repositoryName": "cluster-autoscaler",
 "policyText": "{\n \"Version\" : \"2008-10-17\",\n \"Statement\" :
[{\n \"Sid\" : \"allow public pull\",\n \"Effect\" : \"Allow\",\n \"Principal\" : \"*\",\n \"Action\" : [\"ecr:BatchCheckLayerAvailability\",\n \"ecr:BatchGetImage\", \"ecr:GetDownloadUrlForLayer\"]\n }]\n}"
}
```

- Pour plus de détails sur l'API, reportez-vous [SetRepositoryPolicy](#) à la section Référence des AWS CLI commandes.

## Java

### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Sets the repository policy for the specified ECR repository.
 *
 * @param repoName the name of the ECR repository.
 * @param iamRole the IAM role to be granted access to the repository.
 * @throws RepositoryPolicyNotFoundException if the repository policy does
not exist.
 * @throws EcrException if there is an unexpected error
setting the repository policy.
 */
public void setRepoPolicy(String repoName, String iamRole) {
 /**
 This example policy document grants the specified AWS principal the
permission to perform the
`ecr:BatchGetImage` action. This policy is designed to allow the
specified principal
to retrieve Docker images from the ECR repository.
 */
 String policyDocumentTemplate = ""
```

```
 {
 "Version": "2012-10-17",
 "Statement" : [{
 "Sid" : "new statement",
 "Effect" : "Allow",
 "Principal" : {
 "AWS" : "%s"
 },
 "Action" : "ecr:BatchGetImage"
 }]
 }
 }
 """;

 String policyDocument = String.format(policyDocumentTemplate, iamRole);
 SetRepositoryPolicyRequest setRepositoryPolicyRequest =
SetRepositoryPolicyRequest.builder()
 .repositoryName(repoName)
 .policyText(policyDocument)
 .build();

 CompletableFuture<SetRepositoryPolicyResponse> response =
getAsyncClient().setRepositoryPolicy(setRepositoryPolicyRequest);
 response.whenComplete((resp, ex) -> {
 if (resp != null) {
 System.out.println("Repository policy set successfully.");
 } else {
 Throwable cause = ex.getCause();
 if (cause instanceof RepositoryPolicyNotFoundException) {
 throw (RepositoryPolicyNotFoundException) cause;
 } else if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 String errorMessage = "Unexpected error: " +
cause.getMessage();
 throw new RuntimeException(errorMessage, cause);
 }
 }
 });
 response.join();
}
```

- Pour plus de détails sur l'API, reportez-vous [SetRepositoryPolicy](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

### SDK pour Kotlin

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Sets the repository policy for the specified ECR repository.
 *
 * @param repoName the name of the ECR repository.
 * @param iamRole the IAM role to be granted access to the repository.
 */
suspend fun setRepoPolicy(
 repoName: String?,
 iamRole: String?,
) {
 val policyDocumentTemplate =
 """
 {
 "Version": "2012-10-17",
 "Statement" : [{
 "Sid" : "new statement",
 "Effect" : "Allow",
 "Principal" : {
 "AWS" : "$iamRole"
 },
 "Action" : "ecr:BatchGetImage"
 }]
 }
 """.trimIndent()
 val setRepositoryPolicyRequest =
 SetRepositoryPolicyRequest {
```

```

 repositoryName = repoName
 policyText = policyDocumentTemplate
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val response =
 ecrClient.setRepositoryPolicy(setRepositoryPolicyRequest)
 if (response != null) {
 println("Repository policy set successfully.")
 }
 }
}

```

- Pour plus de détails sur l'API, consultez [SetRepositoryPolicy](#) la section AWS SDK pour la référence de l'API Kotlin.

## Python

### Kit SDK for Python (Boto3)

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

class ECRWrapper:
 def __init__(self, ecr_client: client):
 self.ecr_client = ecr_client

 @classmethod
 def from_client(cls) -> "ECRWrapper":
 """
 Creates a ECRWrapper instance with a default Amazon ECR client.

 :return: An instance of ECRWrapper initialized with the default Amazon
 ECR client.
 """
 ecr_client = boto3.client("ecr")
 return cls(ecr_client)

```

```

def set_repository_policy(self, repository_name: str, policy_text: str):
 """
 Sets the policy for an ECR repository.

 :param repository_name: The name of the repository to set the policy for.
 :param policy_text: The policy text to set.
 """
 try:
 self.ecr_client.set_repository_policy(
 repositoryName=repository_name, policyText=policy_text
)
 print(f"Set repository policy for repository {repository_name}.")
 except ClientError as err:
 if err.response["Error"]["Code"] ==
"RepositoryPolicyNotFoundException":
 logger.error("Repository does not exist. %s.", repository_name)
 raise
 else:
 logger.error(
 "Couldn't set repository policy for repository %s. Here's why
%s",
 repository_name,
 err.response["Error"]["Message"],
)
 raise

```

Exemple qui accorde à un rôle IAM l'accès au téléchargement.

```

def grant_role_download_access(self, role_arn: str):
 """
 Grants the specified role access to download images from the ECR
 repository.

 :param role_arn: The ARN of the role to grant access to.
 """
 policy_json = {
 "Version": "2012-10-17",
 "Statement": [
 {

```

```

 "Sid": "AllowDownload",
 "Effect": "Allow",
 "Principal": {"AWS": role_arn},
 "Action": ["ecr:BatchGetImage"],
 }
],
}

self.ecr_wrapper.set_repository_policy(
 self.repository_name, json.dumps(policy_json)
)

```

- Pour plus de détails sur l'API, consultez [SetRepositoryPolicy](#) le AWS manuel de référence de l'API SDK for Python (Boto3).

## SAP ABAP

### Kit SDK pour SAP ABAP

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

TRY.
 " iv_repository_name = 'my-repository'
 " iv_policy_text = '{"Version":"2012-10-17", "Statement":[...]}'
 lo_ecr->setrepositorypolicy(
 iv_repositoryname = iv_repository_name
 iv_policytext = iv_policy_text).
 MESSAGE |Policy set for repository { iv_repository_name }.| TYPE 'I'.
CATCH /aws1/cx_ecrrepositorynotfound.
 MESSAGE 'Repository not found.' TYPE 'I'.
ENDTRY.

```

- Pour plus de détails sur l'API, reportez-vous [SetRepositoryPolicy](#) à la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Utilisation **StartLifecyclePolicyPreview** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `StartLifecyclePolicyPreview`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- [Principes de base](#)

### CLI

#### AWS CLI

Pour créer un aperçu d'une politique de cycle de vie

L'exemple `start-lifecycle-policy-preview` suivant crée un aperçu d'une politique de cycle de vie défini par un fichier JSON pour le référentiel spécifié.

```
aws ecr start-lifecycle-policy-preview \
 --repository-name "project-a/amazon-ecs-sample" \
 --lifecycle-policy-text "file://policy.json"
```

Contenu de `policy.json` :

```
{
 "rules": [
 {
 "rulePriority": 1,
 "description": "Expire images older than 14 days",
 "selection": {
 "tagStatus": "untagged",
 "countType": "sinceImagePushed",
 "countUnit": "days",
 "countNumber": 14
 }
 },
],
}
```

```

 "action": {
 "type": "expire"
 }
]
}

```

Sortie :

```

{
 "registryId": "012345678910",
 "repositoryName": "project-a/amazon-ecs-sample",
 "lifecyclePolicyText": "{\n \"rules\": [\n {\n\n \"rulePriority\": 1,\n \"description\": \"Expire images older than 14 days\",\n \"selection\": {\n \"tagStatus\": \"untagged\",\n \"countType\": \"sinceImagePushed\",\n \"countUnit\": \"days\",\n \"countNumber\": 14\n },\n \"action\": {\n \"type\": \"expire\"\n }\n }\n]\n}",
 "status": "IN_PROGRESS"
}

```

- Pour plus de détails sur l'API, reportez-vous [StartLifecyclePolicyPreview](#) à la section Référence des AWS CLI commandes.

## Java

### SDK pour Java 2.x

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```

/**
 * Verifies the existence of an image in an Amazon Elastic Container Registry
 (Amazon ECR) repository asynchronously.
 *
 * @param repositoryName The name of the Amazon ECR repository.

```

```
* @param imageTag The tag of the image to verify.
* @throws EcrException if there is an error retrieving the image
information from Amazon ECR.
* @throws CompletionException if the asynchronous operation completes
exceptionally.
*/
public void verifyImage(String repositoryName, String imageTag) {
 DescribeImagesRequest request = DescribeImagesRequest.builder()
 .repositoryName(repositoryName)
 .imageIds(ImageIdentifier.builder().imageTag(imageTag).build())
 .build();


 CompletableFuture<DescribeImagesResponse> response =
getAsyncClient().describeImages(request);
 response.whenComplete((describeImagesResponse, ex) -> {
 if (ex != null) {
 if (ex instanceof CompletionException) {
 Throwable cause = ex.getCause();
 if (cause instanceof EcrException) {
 throw (EcrException) cause;
 } else {
 throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
 }
 } else {
 throw new RuntimeException("Unexpected error: " +
ex.getCause());
 }
 } else if (describeImagesResponse != null && !
describeImagesResponse.imageDetails().isEmpty()) {
 System.out.println("Image is present in the repository.");
 } else {
 System.out.println("Image is not present in the repository.");
 }
 });

 // Wait for the CompletableFuture to complete.
 response.join();
}
```

- Pour plus de détails sur l'API, reportez-vous [StartLifecyclePolicyPreview](#) à la section Référence des AWS SDK for Java 2.x API.

## Kotlin

## SDK pour Kotlin

 Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code AWS](#).

```
/**
 * Verifies the existence of an image in an Amazon Elastic Container Registry
 (Amazon ECR) repository asynchronously.
 *
 * @param repositoryName The name of the Amazon ECR repository.
 * @param imageTag The tag of the image to verify.
 */
suspend fun verifyImage(
 repoName: String?,
 imageTagVal: String?,
) {
 require(!(repoName == null || repoName.isEmpty())) { "Repository name
cannot be null or empty" }
 require(!(imageTagVal == null || imageTagVal.isEmpty())) { "Image tag
cannot be null or empty" }

 val imageId =
 ImageIdentifier {
 imageTag = imageTagVal
 }
 val request =
 DescribeImagesRequest {
 repositoryName = repoName
 imageIds = listOf(imageId)
 }

 EcrClient.fromEnvironment { region = "us-east-1" }.use { ecrClient ->
 val describeImagesResponse = ecrClient.describeImages(request)
 if (describeImagesResponse != null && !
describeImagesResponse.imageDetails?.isEmpty()!!) {
 println("Image is present in the repository.")
 }
 }
}
```

```
 } else {
 println("Image is not present in the repository.")
 }
 }
}
```

- Pour plus de détails sur l'API, consultez [StartLifecyclePolicyPreview](#) la section AWS SDK pour la référence de l'API Kotlin.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Scénarios pour Amazon ECR utilisant des SDK AWS

Les exemples de code suivants vous montrent comment implémenter des scénarios courants dans Amazon ECR avec des AWS SDK. Ces scénarios vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions au sein d'Amazon ECR ou en les combinant à d'autres Services AWS. Chaque exemple inclut un lien vers le code source complet, où vous trouverez des instructions sur la configuration et l'exécution du code.

Les scénarios ciblent un niveau d'expérience intermédiaire pour vous aider à comprendre les actions de service dans leur contexte.

### Exemples

- [Commencer à utiliser Amazon ECR](#)

## Commencer à utiliser Amazon ECR

L'exemple de code suivant illustre comment :

- Créer une image Docker
- Création d'un référentiel Amazon ECR
- Supprimer des ressources

## Bash

### AWS CLI avec le script Bash

#### Note

Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le [référentiel d'exemples de code](#).

```
#!/bin/bash

Amazon ECR Getting Started Script
This script demonstrates the lifecycle of a Docker image in Amazon ECR

Set up logging
LOG_FILE="ecr-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "======"
echo "Amazon ECR Getting Started Tutorial"
echo "======"
echo "This script will:"
echo "1. Create a Docker image"
echo "2. Create an Amazon ECR repository"
echo "3. Authenticate to Amazon ECR"
echo "4. Push the image to Amazon ECR"
echo "5. Pull the image from Amazon ECR"
echo "6. Clean up resources (optional)"
echo "======"

Check prerequisites
echo "Checking prerequisites..."

Check if AWS CLI is installed
if ! command -v aws &> /dev/null; then
 echo "ERROR: AWS CLI is not installed. Please install it before running this
 script."
 echo "Visit https://docs.aws.amazon.com/cli/latest/userguide/install-
 cliv2.html for installation instructions."
 exit 1
fi
```

```
Check if AWS CLI is configured
if ! aws sts get-caller-identity &> /dev/null; then
 echo "ERROR: AWS CLI is not configured properly. Please run 'aws configure'
 to set up your credentials."
 exit 1
fi

Check if Docker is installed
if ! command -v docker &> /dev/null; then
 echo "ERROR: Docker is not installed. Please install Docker before running
 this script."
 echo "Visit https://docs.docker.com/get-docker/ for installation
 instructions."
 exit 1
fi

Check if Docker daemon is running
if ! docker info &> /dev/null; then
 echo "ERROR: Docker daemon is not running. Please start Docker and try
 again."
 exit 1
fi

echo "All prerequisites met."

Initialize variables
REPO_URI=""
TIMEOUT_CMD="timeout 300" # 5-minute timeout for long-running commands

Function to handle errors
handle_error() {
 echo "ERROR: $1"
 echo "Check the log file for details: $LOG_FILE"

 echo "======"
 echo "Resources created:"
 echo "- Docker image: hello-world (local)"
 if [-n "$REPO_URI"]; then
 echo "- ECR Repository: hello-repository"
 echo "- ECR Image: $REPO_URI:latest"
 fi
 echo "======"
}
```

```
 echo "Attempting to clean up resources..."
 cleanup
 exit 1
}

Function to clean up resources
cleanup() {
 echo "======"
 echo "Cleaning up resources..."

 # Delete the image from ECR if it exists
 if [-n "$REPO_URI"]; then
 echo "Deleting image from ECR repository..."
 aws ecr batch-delete-image --repository-name hello-repository --image-ids
imageTag=latest || echo "Failed to delete image, it may not exist or may have
already been deleted."
 fi

 # Delete the ECR repository if it exists
 if [-n "$REPO_URI"]; then
 echo "Deleting ECR repository..."
 aws ecr delete-repository --repository-name hello-repository --force ||
echo "Failed to delete repository, it may not exist or may have already been
deleted."
 fi

 # Remove local Docker image
 echo "Removing local Docker image..."
 docker rmi hello-world:latest 2>/dev/null || echo "Failed to remove local
image, it may not exist or may have already been deleted."
 if [-n "$REPO_URI"]; then
 docker rmi "$REPO_URI:latest" 2>/dev/null || echo "Failed to remove
tagged image, it may not exist or may have already been deleted."
 fi

 echo "Cleanup completed."
 echo "======"
}

Step 1: Create a Docker image
echo "Step 1: Creating a Docker image"

Create Dockerfile
echo "Creating Dockerfile..."
```

```
cat > Dockerfile << 'EOF'
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

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

Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

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

EXPOSE 80

CMD /root/run_apache.sh
EOF

Build Docker image
echo "Building Docker image..."
$TIMEOUT_CMD docker build -t hello-world . || handle_error "Failed to build
 Docker image or operation timed out after 5 minutes"

Verify image was created
echo "Verifying Docker image..."
docker images --filter reference=hello-world || handle_error "Failed to list
 Docker images"

echo "Docker image created successfully."

Step 2: Create an Amazon ECR repository
echo "Step 2: Creating an Amazon ECR repository"

Get AWS account ID
echo "Getting AWS account ID..."
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[-z "$AWS_ACCOUNT_ID" || "$AWS_ACCOUNT_ID" == *"error"*]]; then
 handle_error "Failed to get AWS account ID. Make sure your AWS credentials
 are configured correctly."
fi
echo "AWS Account ID: $AWS_ACCOUNT_ID"
```

```
Get current region
AWS_REGION=$(aws configure get region)
if [[-z "$AWS_REGION"]]; then
 AWS_REGION="us-east-1" # Default to us-east-1 if no region is configured
 echo "No AWS region configured, defaulting to $AWS_REGION"
else
 echo "Using AWS region: $AWS_REGION"
fi

Create ECR repository
echo "Creating ECR repository..."
REPO_RESULT=$(aws ecr create-repository --repository-name hello-repository --
tags key=project,value=doc-smith key=tutorial,value=amazon-elastic-container-
registry-gs)
if [[-z "$REPO_RESULT" || "$REPO_RESULT" == *"error"*]]; then
 handle_error "Failed to create ECR repository"
fi

Extract repository URI
REPO_URI="$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/hello-repository"
echo "Repository URI: $REPO_URI"

Step 3: Authenticate to Amazon ECR
echo "Step 3: Authenticating to Amazon ECR"

echo "Getting ECR login password..."
aws ecr get-login-password --region "$AWS_REGION" | docker login --username
AWS --password-stdin "$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com" ||
handle_error "Failed to authenticate to ECR"

echo "Successfully authenticated to ECR."

Step 4: Push the image to Amazon ECR
echo "Step 4: Pushing the image to Amazon ECR"

Tag the image
echo "Tagging Docker image..."
docker tag hello-world:latest "$REPO_URI:latest" || handle_error "Failed to tag
Docker image"

Push the image with timeout
echo "Pushing image to ECR..."
```

```
$TIMEOUT_CMD docker push "$REPO_URI:latest" || handle_error "Failed to push image
to ECR or operation timed out after 5 minutes"

echo "Successfully pushed image to ECR."

Step 5: Pull the image from Amazon ECR
echo "Step 5: Pulling the image from Amazon ECR"

Remove local image to ensure we're pulling from ECR
echo "Removing local tagged image..."
docker rmi "$REPO_URI:latest" || echo "Warning: Failed to remove local tagged
image"

Pull the image with timeout
echo "Pulling image from ECR..."
$TIMEOUT_CMD docker pull "$REPO_URI:latest" || handle_error "Failed to pull image
from ECR or operation timed out after 5 minutes"

echo "Successfully pulled image from ECR."

List resources created
echo "======"
echo "Resources created:"
echo "- Docker image: hello-world (local)"
echo "- ECR Repository: hello-repository"
echo "- ECR Image: $REPO_URI:latest"
echo "======"

Ask user if they want to clean up resources
echo ""
echo "======"
echo "CLEANUP CONFIRMATION"
echo "======"
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [["$CLEANUP_CHOICE" =~ ^[Yy]$]]; then
 # Step 6: Delete the image from ECR
 echo "Step 6: Deleting the image from ECR"

 DELETE_IMAGE_RESULT=$(aws ecr batch-delete-image --repository-name hello-
repository --image-ids imageTag=latest)
 if [[-z "$DELETE_IMAGE_RESULT" || "$DELETE_IMAGE_RESULT" == *"error"*]];
then
```

```

 echo "Warning: Failed to delete image from ECR"
 else
 echo "Successfully deleted image from ECR."
 fi

 # Step 7: Delete the ECR repository
 echo "Step 7: Deleting the ECR repository"

 DELETE_REPO_RESULT=$(aws ecr delete-repository --repository-name hello-
repository --force)
 if [[-z "$DELETE_REPO_RESULT" || "$DELETE_REPO_RESULT" == *"error"*]]; then
 echo "Warning: Failed to delete ECR repository"
 else
 echo "Successfully deleted ECR repository."
 fi

 # Remove local Docker images
 echo "Removing local Docker images..."
 docker rmi hello-world:latest 2>/dev/null || echo "Warning: Failed to remove
local image"

 echo "All resources have been cleaned up."
else
 echo "Resources were not cleaned up. You can manually clean up later with:"
 echo "aws ecr batch-delete-image --repository-name hello-repository --image-
ids imageTag=latest"
 echo "aws ecr delete-repository --repository-name hello-repository --force"
 echo "docker rmi hello-world:latest"
 echo "docker rmi $REPO_URI:latest"
fi

echo "======"
echo "Tutorial completed!"
echo "Log file: $LOG_FILE"
echo "======"

```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - [BatchDeleteImage](#)
  - [CreateRepository](#)
  - [DeleteRepository](#)

- [GetCallerIdentity](#)
- [GetLoginPassword](#)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultez [Utilisation d'Amazon ECR avec un AWS Kit SDK](#). Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

## Service Quotas Amazon ECR.

Le tableau suivant fournit les quotas de service par défaut pour Amazon Elastic Container Registry (Amazon ECR).

| Nom                                                     | Par défaut                              | Ajusté              | Description                                                                                                                                                                                                             |
|---------------------------------------------------------|-----------------------------------------|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Numérisation d'images de base par 24 heures             | Chaque région prise en charge : 100 000 | Non                 | Nombre maximal d'images pouvant être numérisées sur une période de 24 heures dans le compte courant et la région à l'aide de la numérisation de base. Cette limite inclut à la fois le scan sur push et le scan manuel. |
| Filtres par règle dans une configuration de réplication | Chaque région prise en charge : 100     | Non                 | Nombre maximum de filtres par règle dans une configuration de réplication.                                                                                                                                              |
| Images par référentiel                                  | Chaque région prise en charge : 100 000 | <a href="#">Oui</a> | Nombre maximal d'images par référentiel                                                                                                                                                                                 |
| Parties de couche                                       | Chaque Région prise en charge : 4 200   | Non                 | Nombre maximal de parties de couche Ceci s'applique uniquement si vous utilisez directement des actions d'API Amazon ECR pour lancer des chargements de plusieurs parties pour des opérations de transfert d'images.    |

| Nom                                                 | Par défaut                             | Ajusté | Description                                                                                                                                                                                                                    |
|-----------------------------------------------------|----------------------------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Durée de la stratégie de cycle de vie               | Chaque Région prise en charge : 30 720 | Non    | Nombre maximal de caractères dans une stratégie de cycle de vie.                                                                                                                                                               |
| Taille maximale de la partie de couche              | Chaque Région prise en charge : 10     | Non    | Taille maximale (Mio) d'une partie de couche. Ceci s'applique uniquement si vous utilisez directement des actions d'API Amazon ECR pour lancer des chargements de plusieurs parties pour des opérations de transfert d'images. |
| Taille maximale de couche                           | Chaque Région prise en charge : 52 000 | Non    | Taille maximale (Mio) d'une couche.                                                                                                                                                                                            |
| Taille minimale de la partie de couche              | Chaque région prise en charge : 5      | Non    | Taille minimale (Mio) d'une partie de couche. Ceci s'applique uniquement si vous utilisez directement des actions d'API Amazon ECR pour lancer des chargements de plusieurs parties pour des opérations de transfert d'images. |
| Règles de mise en cache par extraction par registre | Chaque région prise en charge : 50     | Non    | Nombre maximum de règles de mise en cache par extraction.                                                                                                                                                                      |

| Nom                                          | Par défaut                                           | Ajuste              | Description                                                                                                                                                                                                                                                                                                                                        |
|----------------------------------------------|------------------------------------------------------|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Taux de BatchCheckLayerAvailability demandes | Chaque Région prise en charge :<br>1 000 par seconde | <a href="#">Oui</a> | Le nombre maximum de BatchCheckLayerAvailability demandes que vous pouvez effectuer par seconde dans la région actuelle. Lorsqu'une image est transférée vers un référentiel, chaque couche d'image est examinée afin de vérifier si elle a déjà été chargée. Si c'est le cas, la couche d'image est ignorée.                                      |
| Taux de BatchGetImage demandes               | Chaque Région prise en charge :<br>2 000 par seconde | <a href="#">Oui</a> | Le nombre maximum de BatchGetImage demandes que vous pouvez effectuer par seconde dans la région actuelle. Lorsqu'une image est extraite, l'BatchGetImage API est appelée une fois pour récupérer le manifeste de l'image. Si vous demandez une augmentation du quota pour cette API, vérifiez également votre GetDownloadUrlForLayer utilisation. |

| Nom                                    | Par défaut                                      | Ajuste              | Description                                                                                                                                                                                                                                                                         |
|----------------------------------------|-------------------------------------------------|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Taux de CompleteLayerUpload demandes   | Chaque Région prise en charge : 100 par seconde | <a href="#">Oui</a> | Le nombre maximum de CompleteLayerUpload demandes que vous pouvez effectuer par seconde dans la région actuelle. Lorsqu'une image est envoyée, l' CompleteLayerUpload API est appelée une fois pour chaque nouvelle couche d'image pour vérifier que le téléchargement est terminé. |
| Taux de GetAuthorizationToken demandes | Chaque Région prise en charge : 500 par seconde | <a href="#">Oui</a> | Le nombre maximum de GetAuthorizationToken demandes que vous pouvez effectuer par seconde dans la région actuelle.                                                                                                                                                                  |

| Nom                                     | Par défaut                                        | Ajuste              | Description                                                                                                                                                                                                                                                                                                                                                                 |
|-----------------------------------------|---------------------------------------------------|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Taux de GetDownloadUrlForLayer demandes | Chaque Région prise en charge : 3 000 par seconde | <a href="#">Oui</a> | Le nombre maximum de GetDownloadUrlForLayer demandes que vous pouvez effectuer par seconde dans la région actuelle. Lorsqu'une image est extraite, l' GetDownloadUrlForLayer API est appelée une fois par couche d'image qui n'est pas encore mise en cache. Si vous demandez une augmentation du quota pour cette API, vérifiez également votre BatchGetImage utilisation. |
| Taux de InitiateLayerUpload demandes    | Chaque Région prise en charge : 100 par seconde   | <a href="#">Oui</a> | Le nombre maximum de InitiateLayerUpload demandes que vous pouvez effectuer par seconde dans la région actuelle. Lorsqu'une image est envoyée, l' InitiateLayerUpload API est appelée une fois par couche d'image qui n'a pas encore été téléchargée. L'action de l' BatchCheckLayerAvailability API détermine si une couche d'image a été téléchargée ou non.              |

| Nom                              | Par défaut                                      | Ajuste              | Description                                                                                                                                                                                                                                                                                                                             |
|----------------------------------|-------------------------------------------------|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Taux de PutImage demandes        | Chaque Région prise en charge : 10 par seconde  | <a href="#">Oui</a> | Le nombre maximum de PutImage demandes que vous pouvez effectuer par seconde dans la région actuelle. Lorsqu'une image est envoyée et que toutes les nouvelles couches d'image ont été téléchargées, l' PutImage API est appelée une seule fois pour créer ou mettre à jour le manifeste de l'image et les balises associées à l'image. |
| Taux de UploadLayerPart demandes | Chaque Région prise en charge : 500 par seconde | <a href="#">Oui</a> | Le nombre maximum de UploadLayerPart demandes que vous pouvez effectuer par seconde dans la région actuelle. Lorsqu'une image est poussée, chaque nouvelle couche d'image est téléchargée en plusieurs parties et l' UploadLayerPart API est appelée une fois pour chaque nouvelle partie de couche d'image.                            |
| Taux de numérisation d'images    | Chaque Région prise en charge : 1               | Non                 | Nombre maximal d'images numérisées par image et par 24 heures.                                                                                                                                                                                                                                                                          |

| Nom                                                                               | Par défaut                              | Ajusté              | Description                                                                                             |
|-----------------------------------------------------------------------------------|-----------------------------------------|---------------------|---------------------------------------------------------------------------------------------------------|
| Référentiels enregistrés                                                          | Chaque région prise en charge : 100 000 | <a href="#">Oui</a> | Le nombre maximum de référentiels que vous pouvez créer dans ce compte dans la région actuelle.         |
| Règles par politique de cycle de vie                                              | Chaque région prise en charge : 50      | Non                 | Nombre maximal de règles dans une politique de cycle de vie                                             |
| Règles par configuration de réplication                                           | Chaque Région prise en charge : 10      | Non                 | Le nombre maximum de règles dans une configuration de réplication.                                      |
| Balises par image                                                                 | Chaque Région prise en charge : 1 000   | Non                 | Nombre maximal de balises par image.                                                                    |
| Destinations uniques pour toutes les règles dans une configuration de réplication | Chaque région prise en charge : 25      | Non                 | Le nombre maximum de destinations uniques pour toutes les règles dans une configuration de réplication. |

## Gérer vos quotas de service Amazon ECR dans AWS Management Console

Amazon ECR a intégré Service Quotas, un AWS service qui vous permet de consulter et de gérer vos quotas à partir d'un emplacement central. Pour en savoir plus sur Service Quotas, consultez [Qu'est-ce que Service Quotas ?](#) dans le Guide de l'utilisateur Service Quotas.

Service Quotas facilite la recherche de la valeur de tous les quotas de service Amazon ECR.

Afficher les quotas de service Amazon ECR (AWS Management Console)

- Ouvrez la console Service Quotas sur <https://console.aws.amazon.com/servicequotas/>.

2. Dans le panneau de navigation, choisissez Services AWS .
3. Dans la liste AWS Services, recherchez et sélectionnez Amazon Elastic Container Registry (Amazon ECR).

Dans la liste des quotas de service, vous pouvez voir le nom du quota de service, la valeur appliquée (si elle est disponible), le quota AWS par défaut et si la valeur du quota est ajustable.

4. Pour afficher des informations supplémentaires sur un quota de service, notamment la description, choisissez le nom du quota.

Pour demander une augmentation de quota, consultez [Demande d'augmentation de quota](#) dans le guide de l'utilisateur Service Quotas.

## Création d'une CloudWatch alarme pour surveiller les métriques d'utilisation de l'API

Amazon ECR fournit des métriques CloudWatch d'utilisation qui correspondent aux quotas de AWS service pour chacune des entités APIs impliquées dans les actions d'authentification du registre, de transfert d'image et d'extraction d'images. Dans la console Service Quotas, vous pouvez visualiser votre utilisation sur un graphique et configurer des alarmes qui vous alertent lorsque votre utilisation approche un quota de service. Pour de plus amples informations, veuillez consulter [Métriques d'utilisation Amazon ECR](#).

Suivez les étapes ci-dessous pour créer une CloudWatch alarme basée sur l'une des métriques d'utilisation de l'API Amazon ECR.

Pour créer une alarme en fonction de vos quotas d'utilisation Amazon ECR (AWS Management Console)

1. Ouvrez la console Service Quotas sur <https://console.aws.amazon.com/servicequotas/>.
2. Dans le panneau de navigation, choisissez Services AWS .
3. Dans la liste Services AWS , recherchez et sélectionnez Amazon Elastic Container Registry (Amazon ECR).
4. Dans la listeService quotas, sélectionnez le quota d'utilisation Amazon ECR pour lequel vous souhaitez créer une alarme.
5. Dans la section Alarmes Amazon CloudWatch Events, choisissez Create.

6. Pour Alarm threshold (Seuil d'alarme), choisissez le pourcentage de la valeur de quota appliquée que vous souhaitez définir comme valeur d'alarme.
7. Pour Nom de l'alarme, saisissez un nom pour l'alarme, puis choisissez Créer.

# Dépannage d'Amazon ECR

Ce chapitre vous aide à trouver des informations de diagnostic pour Amazon ECR et fournit des étapes de résolution des problèmes courants et des messages d'erreur.

## Rubriques

- [Résolution des commandes Docker et des problèmes liés à l'utilisation d'Amazon ECR](#)
- [Dépannage des messages d'erreur Amazon ECR](#)

## Résolution des commandes Docker et des problèmes liés à l'utilisation d'Amazon ECR

Dans certains cas, l'exécution d'une commande Docker sur Amazon ECR peut générer un message d'erreur. Vous trouverez ci-après la présentation de certains messages d'erreur courants et des résolutions potentielles.

## Rubriques

- [Les journaux Docker ne contiennent pas les messages d'erreur attendus](#)
- [Erreur : « Filesystem Verification Failed » ou « 404: Image Not Found » lors de l'extraction d'une image d'un référentiel Amazon ECR](#)
- [Erreur : « Filesystem Layer Verification Failed » lors de l'extraction d'images d'Amazon ECR](#)
- [Erreurs HTTP 403 ou « no basic auth credentials » lors de la transmission au référentiel](#)

## Les journaux Docker ne contiennent pas les messages d'erreur attendus

Pour commencer à déboguer tout problème lié à Docker, commencez par activer la sortie de débogage Docker sur le démon Docker exécuté sur vos instances hôtes. Si vous utilisez des images extraites d'Amazon ECR sur des instances de conteneur Amazon ECS, consultez la [section Configuration de la sortie détaillée du démon Docker dans le manuel](#) Amazon Elastic Container Service Developer Guide.

## Erreur : « Filesystem Verification Failed » ou « 404: Image Not Found » lors de l'extraction d'une image d'un référentiel Amazon ECR

Il est possible que vous receviez l'erreur `filesystem verification failed` lorsque vous utilisez la commande `docker pull` afin d'extraire une image d'un référentiel Amazon ECR avec Docker 1.9 ou une version ultérieure. Il est possible que vous receviez l'erreur `404: Image not found` lorsque vous utilisez des versions Docker antérieures à 1.9.

Quelques motifs possibles et leurs explications figurent ci-dessous :

### Le disque local est plein

Si le disque local sur lequel vous exécutez `docker pull` est plein, le hachage SHA-1 qui est calculé sur le fichier local peut être différent de celui calculé par Amazon ECR. Vérifiez que le disque local dispose d'un espace libre suffisant pour stocker l'image Docker transmise. Vous pouvez également supprimer d'anciennes images afin de libérer de l'espace pour les nouvelles. Utilisez la commande `docker images` pour afficher une liste de toutes les images Docker téléchargées localement, ainsi que de leurs tailles.

### Le client ne peut pas se connecter au référentiel distant en raison d'une erreur de réseau

Les appels à un référentiel Amazon ECR requièrent une connexion à Internet fonctionnelle. Vérifiez vos paramètres réseau et assurez-vous que les autres outils et applications peuvent accéder aux ressources sur Internet. Si vous exécutez `docker pull` sur une instance Amazon EC2 dans un sous-réseau privé, vérifiez que le sous-réseau offre un acheminement vers Internet. Utilisez un serveur de traduction d'adresses réseau (NAT) ou une passerelle NAT gérée.

Pour l'instant, les appels d'un référentiel Amazon ECR ont également besoin d'un accès réseau via le pare-feu de votre entreprise à Amazon Simple Storage Service (Amazon S3). Si votre organisation utilise un logiciel de pare-feu ou un appareil NAT qui autorise les points de terminaison de service, vérifiez que les points de terminaison de service Amazon S3 de votre région actuelle sont autorisés.

Si vous utilisez Docker derrière un proxy HTTP, vous pouvez configurer Docker avec les paramètres de proxy appropriés. Pour en savoir plus, consultez [Proxy HTTP](#) dans la documentation Docker.

## Erreur : « Filesystem Layer Verification Failed » lors de l'extraction d'images d'Amazon ECR

Il est possible que vous receviez l'erreur `image image-name not found` lors de l'extraction d'images à l'aide de la commande `docker pull`. Si vous inspectez les journaux de Docker, vous verrez peut-être une erreur similaire à ce qui suit :

```
filesystem layer verification failed for digest sha256:2b96f...
```

Cette erreur indique qu'une ou plusieurs couches de votre image n'ont pas pu être téléchargées. Quelques motifs possibles et leurs explications figurent ci-dessous :

**Vous utilisez une version plus ancienne de Docker**

Cette erreur peut se produire dans un petit pourcentage des cas lors de l'utilisation d'une version de Docker antérieure à 1.10. Dans ce cas, mettez à niveau le client Docker vers la version 1.10 ou une version ultérieure.

**Votre client a rencontré une erreur de réseau ou de disque**

Un disque plein ou un problème de réseau peuvent empêcher le téléchargement d'une ou de plusieurs couches, comme nous l'avons vu pour le message `Filesystem verification failed`. Suivez les recommandations ci-dessus pour veiller à ce que votre système de fichiers ne soit pas plein et à autoriser l'accès à Amazon S3 depuis votre réseau.

## Erreurs HTTP 403 ou « no basic auth credentials » lors de la transmission au référentiel

Vous êtes susceptible de recevoir une erreur HTTP `403 (Forbidden)` ou le message d'erreur `no basic auth credentials` des commandes `docker push` ou `docker pull`, même si vous vous êtes authentifié correctement auprès de Docker à l'aide de la commande `aws ecr get-login-password`. Voici quelques causes connues de ce problème :

**Vous vous êtes authentifié auprès d'une région différente**

Les demandes d'authentification sont liées à des régions spécifiques et ne peuvent pas être utilisées d'une région à l'autre. Par exemple, si vous obtenez un jeton d'autorisation de la région USA Ouest (Oregon), vous ne pourrez pas l'utiliser pour vous authentifier auprès des référentiels

de la région USA Est (Virginie du Nord). Pour résoudre le problème, vérifiez que vous avez récupéré un jeton d'authentification à partir de la région dans laquelle votre référentiel se trouve. Pour de plus amples informations, veuillez consulter [the section called "Authentication de registre"](#).

Vous vous êtes authentifié pour effectuer une transmission vers un référentiel pour lequel vous n'avez pas d'autorisations

Vous n'avez pas les autorisations nécessaires pour effectuer une transmission vers le référentiel. Pour de plus amples informations, veuillez consulter [Politiques relatives aux référentiels privés dans Amazon ECR](#).

Votre jeton a expiré

La période d'expiration du jeton d'autorisation par défaut pour les jetons obtenus à l'aide de l'opération `GetAuthorizationToken` est de 12 heures.

Bogue dans le gestionnaire des informations d'identification `wincrd`

Certaines versions de Docker pour Windows utilisent un gestionnaire d'informations d'identification appelé `wincrd`, qui ne gère pas correctement la commande de connexion Docker produite par `aws ecr get-login-password` (pour plus d'informations, voir [CredsStoreÉchec avec les référentiels privés](#)). Vous pouvez exécuter la commande de connexion de Docker qui est générée, mais lorsque vous tentez de transmettre ou d'extraire des images, ces commandes échouent. Vous pouvez contourner ce bogue en supprimant le schéma `https://` de l'argument de registre dans la commande de connexion de Docker sortie de `aws ecr get-login-password`. Voici un exemple de commande de connexion Docker sans le schéma HTTPS.

```
docker login -u AWS -p <password> <aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

## Dépannage des messages d'erreur Amazon ECR

Dans certains cas, un appel d'API que vous avez lancé par le biais de la console Amazon ECR se termine AWS CLI par un message d'erreur. Vous trouverez ci-après la présentation de certains messages d'erreur courants et des résolutions potentielles.

### HTTP 429 : trop de requêtes ou `ThrottlingException`

Vous pouvez recevoir une 429: Too Many Requests erreur ou une erreur suite à une `ThrottlingException` ou plusieurs actions Amazon ECR ou à un ou plusieurs appels d'API. Cela

indique que vous avez appelé un seul point de terminaison dans Amazon ECR à plusieurs reprises au cours d'une période de temps limitée et que vos demandes se retrouvent limitées. La limitation se produit lorsque des appels d'un seul point de terminaison par un seul utilisateur dépassent un seuil donné au cours d'une période de temps donnée.

Chaque opération d'API dans Amazon ECR est associée à des limitations de débit. Par exemple, la limitation de l'action [GetAuthorizationToken](#) est fixée à 20 transactions par seconde (TPS), avec une rafale pouvant atteindre 200 TPS autorisée. Dans chaque région, chaque compte reçoit un compartiment qui peut stocker jusqu'à 200 crédits `GetAuthorizationToken`. Ces crédits sont réapprovisionnés au rythme de 20 par seconde. Si votre compartiment contient 200 crédits, vous pouvez réaliser 200 transactions d'API `GetAuthorizationToken` par seconde pendant une seconde, puis soutenir 20 transactions par seconde indéfiniment. Pour plus d'informations sur les limites de débit pour Amazon ECR APIs, consultez [Service Quotas Amazon ECR](#).

Pour gérer les erreurs de limitation, implémentez une fonction de nouvelle tentative avec une interruption incrémentielle dans votre code. Pour plus d'informations, consultez la section [Comportement](#) des tentatives dans le guide de référence AWS SDKs et Tools. Une autre option consiste à demander une augmentation de la limite de débit, ce que vous pouvez faire à l'aide de la console Service Quotas. Pour plus d'informations, consultez [Gérer vos quotas de service Amazon ECR dans AWS Management Console](#).

## HTTP 403 : « User [arn] is not authorized to perform [operation] »

Vous êtes susceptible de recevoir l'erreur suivante lorsque vous tentez d'effectuer une action avec Amazon ECR :

```
$ aws ecr get-login-password
```

```
A client error (AccessDeniedException) occurred when calling the GetAuthorizationToken operation:
```

```
User: arn:aws:iam::account-number:user/username is not authorized to perform:
ecr:GetAuthorizationToken on resource: *
```

Cette erreur indique que l'utilisateur n'est pas autorisé à utiliser Amazon ECR ou que les autorisations dont il dispose n'ont pas été configurées correctement. Si vous effectuez des actions liées au référentiel Amazon ECR, vérifiez plus particulièrement si l'utilisateur dispose des autorisations nécessaires pour accéder à ce référentiel. Pour en savoir plus sur la création et la vérification des autorisations pour Amazon ECR, consultez [Gestion des identités et des accès au registre de conteneur Amazon Elastic](#).

## HTTP 404 : « Repository Does Not Exist »

Si vous spécifiez un référentiel Docker Hub qui n'existe pas, Docker Hub le créera automatiquement. Avec Amazon ECR, les nouveaux référentiels doivent être créés explicitement avant de pouvoir être utilisés. Cela évite que de nouveaux référentiels soient créés accidentellement (par exemple, en raison de fautes de frappe) et vous permet également de veiller à ce qu'une politique d'accès de sécurité appropriée soit attribuée explicitement à tout nouveau référentiel. Pour plus d'informations sur la création des référentiels, consultez [Référentiels privés Amazon ECR](#).

## Erreur : impossible d'effectuer une connexion interactive à partir d'un appareil autre que TTY

Si vous recevez le message d'erreur `Cannot perform an interactive login from a non TTY device`, les étapes de dépannage suivantes devraient vous aider.

- Vérifiez que vous utilisez AWS CLI la version 2 et que vous ne disposez pas d'une version conflictuelle de la AWS CLI version 1 sur votre système. Pour plus d'informations, consultez [Installation ou mise à jour de la version la plus récente de l' AWS CLI](#).
- Vérifiez que vous avez configuré votre compte AWS CLI avec des informations d'identification valides. Pour plus d'informations, consultez [Installation ou mise à jour de la version la plus récente de l' AWS CLI](#).
- Vérifiez que la syntaxe de votre AWS CLI commande est correcte.

# Utilisation de Podman avec Amazon ECR

L'utilisation Podman d'Amazon ECR permet aux entreprises de tirer parti de la sécurité et de la simplicité d'Amazon ECR Podman tout en bénéficiant de l'évolutivité et de la fiabilité d'Amazon ECR pour la gestion des images de conteneurs. En suivant les étapes et les commandes décrites, les développeurs et les administrateurs peuvent rationaliser leurs flux de travail liés aux conteneurs, renforcer la sécurité et optimiser l'utilisation des ressources. Alors que la conteneurisation continue de prendre de l'ampleur, l'utilisation d'PodmanAmazon ECR fournit une solution robuste et flexible pour gérer et déployer des applications conteneurisées.

## Utiliser Podman pour s'authentifier auprès d'Amazon ECR

Avant d'interagir avec Amazon ECR à l'aide d'Amazon ECRPodman, une authentification est requise. Cela peut être réalisé en exécutant la `aws ecr get-login-password` commande pour récupérer un jeton d'authentification, puis en utilisant ce jeton avec la `podman login` commande pour s'authentifier auprès d'Amazon ECR.

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

## Utilisation de l'assistant d'identification Amazon ECR avec Podman

Amazon ECR fournit un assistant d'identification Docker qui fonctionne avec Podman. L'assistant d'identification facilite le stockage et l'utilisation des informations d'identification Docker lors du transfert et de l'extraction d'images vers Amazon ECR. Pour connaître les étapes d'installation et de configuration, consultez [Assistant d'informations d'identification Amazon ECR Docker](#).

### Important

Podman ne prend en charge que partiellement la `docker-creds-helper` spécification. Podman prend en charge le `credHelpers` mot clé dans la configuration de Docker mais ne le prend pas en charge. `credsStore`

Pour utiliser l'assistant d'identification Amazon ECR avec Podman, configurez votre fichier de configuration Docker au format suivant : `credHelpers`

```
{
 "credHelpers": {
```

```
"public.ecr.aws": "ecr-login",
"<aws_account_id>.dkr.ecr.<region>.amazonaws.com": "ecr-login"
}
}
```

La credsStore configuration suivante n'est pas prise en charge par Podman :

```
{
 "credsStore": "ecr-login"
}
```

### Note

L'assistant d'identification Docker d'Amazon ECR ne prend pas actuellement en charge l'authentification multifactorielle (MFA).

## Extraire des images depuis Amazon ECR avec Podman

Une fois l'authentification réussie, les images du conteneur peuvent être extraites d'Amazon ECR à l'aide de la `podman pull` commande avec l'URI complet du référentiel Amazon ECR.

```
podman pull aws_account_id.dkr.ecr.region.amazonaws.com/repository_name:tag
```

## Exécution de conteneurs pour Amazon ECR avec Podman

Une fois que l'image souhaitée a été extraite, un conteneur peut être instancié à l'aide de la `podman run` commande.

```
podman run -d aws_account_id.dkr.ecr.region.amazonaws.com/repository_name:tag
```

## Transférer des images vers Amazon ECR avec Podman

Pour envoyer une image locale vers Amazon ECR, l'image doit d'abord être étiquetée avec l'URI du référentiel Amazon ECR en utilisant `podman tag`, puis la `podman push` commande peut être utilisée pour télécharger l'image sur Amazon ECR.

**podman**

```
tag local_image:tag aws_account_id.dkr.ecr.region.amazonaws.com/repository_name:tag
```

```
podman push aws_account_id.dkr.ecr.region.amazonaws.com/repository_name:tag
```

## Historique du document

Le tableau suivant décrit les modifications importantes apportées à la documentation depuis la dernière version d'Amazon ECR. Nous mettons aussi la documentation à jour régulièrement pour prendre en compte les commentaires qui nous sont envoyés.

| Modifier                                                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                | Date             |
|---------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| La dépréciation du scan basé sur Clair est terminée           | Depuis le 2 février 2026, la numérisation d'images basée sur Clair est devenue obsolète et tous les comptes ECR ont été migrés vers la numérisation de base native. AWS La documentation a été mise à jour pour supprimer le contenu spécifique à Clair et refléter le fait que le AWS mode natif est désormais la seule implémentation de la numérisation de base.                                                                                                        | 2 février 2026   |
| Signature gérée par ECR                                       | Amazon ECR prend désormais en charge la signature d'images de conteneurs gérés afin d'améliorer votre niveau de sécurité et d'éliminer les frais opérationnels liés à la configuration de la signature. La signature d'images de conteneurs vous permet de vérifier que les images proviennent de sources fiables. Pour de plus amples informations, veuillez consulter <a href="#">Signature gérée</a> .                                                                  | 21 novembre 2025 |
| IPv6 support pour AWS PrivateLink (points de terminaison VPC) | Ajout de la prise en charge de la connectivité à double pile (IPv4 et IPv6) pour les points de terminaison Amazon ECR VPC alimentés par. AWS PrivateLink Vous pouvez désormais créer des points de terminaison VPC à double pile qui gèrent le trafic à la IPv4 fois sur des adresses IP privées IPv6 et sur des adresses IP. Pour de plus amples informations, veuillez consulter <a href="#">Points de terminaison VPC de l'interface Amazon ECR (AWS PrivateLink)</a> . | 21 novembre 2025 |

| Modifier                                                                                               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Date             |
|--------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| Fonction d'archivage ECR                                                                               | Amazon ECR a ajouté la prise en charge de l'archivage des images pour une conservation à long terme. Pour de plus amples informations, veuillez consulter <a href="#">Archivage d'une image dans Amazon ECR</a> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 19 novembre 2025 |
| Mise à jour pour inclure la prise en charge des modèles d'exclusion d'immutabilité des balises d'image | Amazon ECR a mis à jour ses fonctionnalités de balisage d'image afin d'inclure des modèles d'exclusion d'immutabilité des balises d'image lors de la création et de la mise à jour de référentiels. Vous pouvez désormais spécifier des balises qui peuvent être mises à jour même lorsque l'immutabilité des balises est activée dans un référentiel en définissant des modèles génériques (tels que <code>latest</code> «,v*.beta, » <code>dev*</code> ) afin d'exclure des balises spécifiques des règles d'immutabilité tout en préservant l'immutabilité de toutes les autres balises. Pour de plus amples informations, veuillez consulter <a href="#">Création d'un référentiel privé Amazon ECR pour stocker des images</a> . | 23 juillet 2025  |
| Numérisation d'images améliorée mise à jour pour fournir des informations sur l'utilisation des images | Amazon ECR a mis à jour les fonctionnalités améliorées de numérisation d'images afin d'inclure une visibilité sur la manière dont les images sont utilisées sur Amazon EKS et Amazon ECS. Pour plus d'informations, consultez <a href="#">Scanner des images pour détecter les vulnérabilités du système d'exploitation et des packages de langage de programmation dans Amazon ECR</a> .                                                                                                                                                                                                                                                                                                                                             | 16 juin 2025     |
| IPv6 soutien                                                                                           | Ajout de la prise en charge des demandes adressées aux registres Amazon ECR en utilisant à la fois des points de terminaison à double pile et des IPv4 points de terminaison à double pile (IPv4 et). IPv6 Pour de plus amples informations, veuillez consulter <a href="#">Envoyer des demandes aux registres Amazon ECR</a> .                                                                                                                                                                                                                                                                                                                                                                                                       | 30 avril 2025    |

| Modifier                                                                                                                        | Description                                                                                                                                                                                                                                                                                                                                                                                                                                      | Date             |
|---------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| Ajout de la prise en charge du registre privé Amazon ECR pour extraire le cache                                                 | Amazon ECR a ajouté la prise en charge de la création de règles de cache d'extraction pour le registre privé Amazon ECR. Pour plus d'informations, consultez <a href="#">Synchroniser un registre en amont avec un registre privé Amazon ECR</a> et <a href="#">Rôle lié à un service Amazon ECR pour la mise en cache par extraction</a> .                                                                                                      | 12 mars 2025     |
| Ajout du support pour définir le champ d'application de la politique de registre                                                | Amazon ECR a ajouté la prise en charge de la configuration de l'étendue de la politique de registre pour votre registre privé. Pour plus d'informations, consultez <a href="#">Autorisations de registre privé dans Amazon ECR</a> et Registre <a href="#">privé Amazon ECR</a> .                                                                                                                                                                | 23 décembre 2024 |
| <a href="#">Amazon EC2 Container RegistryPullOnly</a> — Nouvelle politique                                                      | Amazon ECR a ajouté une nouvelle politique qui accorde des autorisations d'extraction uniquement à Amazon ECR.                                                                                                                                                                                                                                                                                                                                   | 10 octobre 2024  |
| Les opérations proxy du client Docker/OCI dans les événements pointent désormais vers CloudTrail <code>ecr.amazonaws.com</code> | La valeur <code>ecr.amazonaws.com</code> remplacée <code>AWS_Internal</code> dans les champs Agent utilisateur ( <code>userAgent</code> ) et Adresse IP source ( <code>sourceIPAddress</code> ) pour les CloudTrail événements associés aux points de terminaison Docker/OCI du client. Pour obtenir des exemples, consultez <a href="#">Exemple : Action d'extraction d'image</a> et <a href="#">Exemple : Action de transmission d'image</a> . | 1er juillet 2024 |
| Ajout de la description du nouveau rôle lié au service Amazon ECR pour les modèles de création de référentiels.                 | Amazon ECR utilise un rôle lié à un service nommé <code>AWSServiceRoleForECRTemplate</code> qui autorise Amazon ECR à effectuer des actions en votre nom afin de terminer les actions de création de modèles de référentiel. Pour de plus amples informations, veuillez consulter <a href="#">Rôle lié au service Amazon ECR pour les modèles de création de référentiels</a> .                                                                  | 20 juin 2024     |

| Modifier                                                                                                                           | Description                                                                                                                                                                                                                                                                                                                                                                  | Date             |
|------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| Le rôle ECRTemplateServiceRolePolicy lié au service a été ajouté.                                                                  | Le rôle ECRTemplateServiceRolePolicy lié au service a été ajouté. Pour de plus amples informations, consultez <a href="#">ECRTemplateServiceRolePolicy</a> .                                                                                                                                                                                                                 | 20 juin 2024     |
| Ajout de la réplication entre régions et entre comptes dans les régions de Chine.                                                  | Amazon ECR a ajouté la prise en charge de la région de Chine pour le filtrage des référentiels répliqués. Pour de plus amples informations, consultez <a href="#">Réplication d'images privées sur Amazon ECR</a> .                                                                                                                                                          | 15 mai 2024      |
| Ajout d' GitLab un registre de conteneurs pour parcourir les règles de cache                                                       | Amazon ECR a ajouté la prise en charge de la création de règles de cache d'extraction pour le registre des GitLab conteneurs. Pour de plus amples informations, veuillez consulter <a href="#">Synchroniser un registre en amont avec un registre privé Amazon ECR</a> .                                                                                                     | 8 mai 2024       |
| Mise à jour de la politique de cycle de vie d'Amazon ECR pour ajouter la prise en charge de l'utilisation de caractères génériques | Amazon ECR a ajouté une prise en charge pour les caractères génériques dans une politique de cycle de vie grâce à l'utilisation du paramètre tagPatternList dans une règle de politique de cycle de vie. Pour de plus amples informations, veuillez consulter <a href="#">Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR</a> . | 18 décembre 2023 |
| Modèles de création de référentiels Amazon ECR                                                                                     | Amazon ECR a ajouté une prise en charge pour les modèles de création de référentiels. Pour de plus amples informations, veuillez consulter <a href="#">Modèles pour contrôler les référentiels créés lors d'une opération d'extraction du cache, d'une création push ou d'une action de réplication</a> .                                                                    | 15 novembre 2023 |

| Modifier                                                                                                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                | Date             |
|---------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| Ajout d'une mise en cache par extraction d'Amazon ECR pris en charge pour les registres en amont authentifiés | Amazon ECR a ajouté une prise en charge pour l'utilisation de registres en amont qui nécessitent une authentification pour vos règles de mise en cache par extraction. Pour de plus amples informations, veuillez consulter <a href="#">Synchroniser un registre en amont avec un registre privé Amazon ECR</a> .                                                                                                          | 15 novembre 2023 |
| <a href="#">AWSECRPullThroughCache_ServiceRolePolicy</a> — Mise à jour d'une politique existante              | Amazon ECR a ajouté de nouvelles autorisations à la politique <code>AWSECRPullThroughCache_ServiceRolePolicy</code> . Ces autorisations permettent à Amazon ECR de récupérer le contenu chiffré d'un secret de Secrets Manager. Cela est nécessaire lors de l'utilisation d'une règle de mise en cache par extraction pour mettre en cache des images provenant d'un registre en amont qui nécessite une authentification. | 15 novembre 2023 |
| Signature d'image Amazon ECR                                                                                  | Amazon ECR et AWS Signer ajout de la prise en charge de la création et de l'envoi de signatures d'images de conteneurs à l'aide du client Notary. Pour de plus amples informations, veuillez consulter <a href="#">Signer des images dans Amazon ECR</a> .                                                                                                                                                                 | 6 juin 2023      |
| Ajout d'un registre de conteneurs Kubernetes pour consulter les règles de mise en cache par extraction        | Amazon ECR a ajouté la prise en charge de la création de règles de mise en cache par extraction pour le registre de conteneurs Kubernetes. Pour de plus amples informations, veuillez consulter <a href="#">Synchroniser un registre en amont avec un registre privé Amazon ECR</a> .                                                                                                                                      | 1 juin 2023      |

| Modifier                                                                                      | Description                                                                                                                                                                                                                                                                                                                                | Date              |
|-----------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| Prise en charge de la durée de numérisation améliorée Amazon ECR                              | Amazon Inspector a ajouté la prise en charge de la définition de la durée pendant laquelle vos référentiels sont surveillés lorsque la numérisation améliorée est activée. Pour de plus amples informations, veuillez consulter <a href="#">Modification de la durée de numérisation améliorée pour les images dans Amazon Inspector</a> . | 28 juin 2022      |
| Amazon ECR envoie les statistiques du nombre d'extractions du référentiel à Amazon CloudWatch | Amazon ECR envoie les statistiques du nombre d'appels du référentiel à Amazon CloudWatch. Pour de plus amples informations, veuillez consulter <a href="#">Métriques de référentiel Amazon ECR</a> .                                                                                                                                       | 6 janvier 2022    |
| Prise en charge étendue de la réplication                                                     | Amazon ECR a ajouté une prise en charge pour le filtrage des référentiels répliqués. Pour de plus amples informations, veuillez consulter <a href="#">Réplication d'images privées sur Amazon ECR</a> .                                                                                                                                    | 21 septembre 2021 |
| AWS politiques gérées pour Amazon ECR                                                         | Amazon ECR a ajouté de la documentation sur les politiques AWS gérées. Pour de plus amples informations, veuillez consulter <a href="#">AWS politiques gérées pour Amazon Elastic Container Registry</a> .                                                                                                                                 | 24 juin 2021      |
| Réplication inter-régions et inter-comptes                                                    | Amazon ECR a ajouté une prise en charge pour la configuration des paramètres de réplication de votre registre privé. Pour de plus amples informations, veuillez consulter <a href="#">Paramètres du registre privé dans Amazon ECR</a> .                                                                                                   | 8 décembre 2020   |

| Modifier                          | Description                                                                                                                                                                                                                                                                                                                                                                                                              | Date            |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| Prise en charge des artefacts OCI | <p>Amazon ECR a ajouté une prise en charge pour pousser et tirer les artefacts OCI (Open Container Initiative). Un nouveau paramètre <code>artifactMediaTypes</code> a été ajouté à la réponse d'API <code>DescribeImages</code> pour indiquer le type d'artefact.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Transférer un graphique de Helm vers un référentiel privé Amazon ECR</a>.</p> | 24 août 2020    |
| Chiffrement au repos              | <p>Amazon ECR a ajouté une prise en charge pour la configuration du chiffrement de vos référentiels à l'aide du chiffrement côté serveur avec des clés gérées par le client stockées dans AWS Key Management Service (AWS KMS).</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Chiffrement au repos</a>.</p>                                                                                    | 29 juillet 2020 |
| Images multi-architecture         | <p>Amazon ECR a ajouté une prise en charge pour la création et la transmission des listes manifeste Docker utilisées pour les images multi-architecture.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Transmission d'une image multi-architecture vers un référentiel privé Amazon ECR</a>.</p>                                                                                               | 28 avril 2020   |

| Modifier                             | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | Date            |
|--------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| Métriques d'utilisation Amazon ECR   | <p>Amazon ECR a ajouté des statistiques CloudWatch d'utilisation qui fournissent une visibilité sur l'utilisation des ressources de votre compte. Vous avez également la possibilité de créer des CloudWatch alarmes à partir de la console Service Quotas CloudWatch et de la console Service Quotas pour recevoir des alertes lorsque votre utilisation approche le quota de service appliqué.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Métriques d'utilisation Amazon ECR</a>.</p> | 28 février 2020 |
| Service Quotas Amazon ECR mis à jour | <p>Mise à jour des quotas de service Amazon ECR pour inclure des quotas par API.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Service Quotas Amazon ECR</a>.</p>                                                                                                                                                                                                                                                                                                                          | 19 février 2020 |
| Commande get-login -password ajoutée | <p>Prise en charge ajoutée pour la get-login-password, qui fournit une méthode simple et sécurisée permettant de récupérer un jeton d'autorisation.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Utiliser un jeton d'autorisation</a>.</p>                                                                                                                                                                                                                                                | 4 février 2020  |
| Numérisation d'images                | <p>Prise en charge ajoutée pour la numérisation d'images, ce qui permet d'identifier les vulnérabilités logicielles dans vos images de conteneur. Amazon ECR utilise la base de données Common Vulnerabilities and Exposures (CVEs) du projet open source CoreOS Clair et vous fournit une liste des résultats d'analyse.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Scannez les images pour détecter les vulnérabilités logicielles dans Amazon ECR</a>.</p>                           | 24 oct. 2019    |

| Modifier                                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                       | Date              |
|--------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| Politique de point de terminaison d'un VPC                   | <p>Prise en charge ajoutée pour la définition d'une politique IAM relative aux points de terminaison d'un VPC d'interface Amazon ECR.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Créer une politique de point de terminaison pour vos points de terminaison d'un VPC Amazon ECR</a>.</p>                                                                                                                                             | 26 septembre 2019 |
| Caractère immuable des étiquettes d'image                    | <p>Prise en charge ajoutée pour la configuration du caractère immuable d'un référentiel afin qu'il ne soit pas possible d'écraser les étiquettes d'image.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Empêcher le remplacement des balises d'image dans Amazon ECR</a>.</p>                                                                                                                                                           | 25 juillet 2019   |
| Points de terminaison d'un VPC d'interface (AWS PrivateLink) | <p>Ajout de la prise en charge de la configuration des points de terminaison VPC d'interface alimentés par AWS PrivateLink. Cela vous permet de créer une connexion privée entre votre VPC et Amazon ECR sans avoir besoin d'un accès Internet, via une instance NAT, une connexion VPN ou Direct Connect.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Points de terminaison VPC de l'interface Amazon ECR (AWS PrivateLink)</a>.</p> | 25 janvier 2019   |
| Étiquette des ressources                                     | <p>Amazon ECR a ajouté une prise en charge pour l'ajout d'étiquettes de métadonnées dans vos référentiels.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Marquage d'un référentiel privé dans Amazon ECR</a>.</p>                                                                                                                                                                                                                       | 18 décembre 2018  |
| Changement de nom Amazon ECR                                 | <p>Le registre de conteneur Amazon Elastic est renommé (anciennement, Amazon EC2 Container Registry).</p>                                                                                                                                                                                                                                                                                                                                                         | 21 novembre 2017  |

| Modifier                                                           | Description                                                                                                                                                                                                                                                                                                               | Date             |
|--------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| Politiques de cycle de vie                                         | <p>Les politiques de cycle de vie Amazon ECR vous permettent de préciser la gestion du cycle de vie des images dans un référentiel.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Automatisez le nettoyage des images en utilisant les politiques de cycle de vie d'Amazon ECR</a>.</p>         | 11 octobre 2017  |
| Amazon ECR prend en charge le manifeste 2, schéma 2 d'image Docker | <p>Amazon ECR prend désormais en charge le manifeste V2, schéma 2 d'image Docker (utilisé avec la version 1.10 de Docker et les versions les plus récentes).</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Prise en charge du format de manifeste d'image de conteneur dans Amazon ECR</a>.</p> | 27 janvier 2017  |
| Disponibilité générale d'Amazon ECR                                | <p>Amazon Elastic Container Registry (Amazon ECR) est un service de registre Docker AWS géré qui est sécurisé, évolutif et fiable.</p>                                                                                                                                                                                    | 21 décembre 2015 |

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.