Guide de l’utilisateur

AWS Service d'injection de défauts

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

AWS Service d'injection de défauts: Guide de l’utilisateur

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

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

Table of Contents

Qu'est-ce que AWS le FIS ? 1
Concepts 1
Actions 2
Cibles 2
Conditions d'arrêt 2
Soutenu Services AWS 3
Accédez au AWS FIS 3
Tarification 4
Planification de vos expériences 5
Principes de base et directives 5
Directives de planification des expériences 7
Composants du modèle d'expérience 9
Syntaxe du modèle 10
Mise en route 10
Actions 10
Syntaxe des actions 11
Identifiants d'action 12
Paramètres d'action 12
Objectifs d'action 13
Durée de l'action 14
Exemples d'actions 14
Cibles 17
Syntaxe cible 18
Types de ressources 19
Identifier les ressources cibles 20
Mode de sélection 24
Exemples de cibles 25
Exemples de filtres 26
Conditions d'arrêt 31
Syntaxe de la condition d'arrêt 32
En savoir plus 32
Rôle d'expérience 33
Prérequis 33
Option 1 : créer un rôle d'essai et associer une politique AWS gérée 35

Option 2 : créer un rôle d'essai et ajouter un document de politique intégré 36
Configuration du rapport d'expérimentation 37
Syntaxe de configuration du rapport d'expérimentation 40
Autorisations relatives aux rapports d'expérimentation 42
Meilleures pratiques en matière de rapports d'expérimentation 44
Options d'expérimentation 44
Ciblage des comptes 45
Mode de résolution cible vide 47
Mode actions 47
Référence des actions 49
Actions d'injection de défauts 50
aws:fis:inject-api-internal-error 50
aws:fis:inject-api-throttle-error 51
aws:fis:inject-api-unavailable-error 51
Action de rétablissement 52
aws:arc:start-zonal-autoshift 52
Attendre une action 54
aws:fis:wait 54
CloudWatch Actions Amazon 54
aws:cloudwatch:assert-alarm-state 54
Actions Amazon DynamoDB 55
aws:dynamodb:global-table-pause-replication 55
Actions Amazon EBS 57
aws:ebs:pause-volume-io 57
EC2 Actions Amazon 58
aws:ec2:api-insufficient-instance-capacity-error 59
aws:ec2:asg-insufficient-instance-capacity-error 59
aws:ec2:reboot-instances 60
aws:ec2:send-spot-instance-interruptions 61
aws:ec2:stop-instances 62
aws:ec2:terminate-instances 63
Actions d'Amazon ECS 63
aws:ecs:drain-container-instances 64
aws:ecs:stop-task 64
aws:ecs:task-cpu-stress 65
aws:ecs:task-io-stress 66

aws:ecs:task-kill-process 67
aws:ecs:task-network-blackhole-port 67
aws:ecs:task-network-latency 68
aws:ecs:task-network-packet-loss 69
Actions d'Amazon EKS 70
aws:eks:inject-kubernetes-custom-resource 71
aws:eks:pod-cpu-stress 72
aws:eks:pod-delete 73
aws:eks:pod-io-stress 75
aws:eks:pod-memory-stress 76
aws:eks:pod-network-blackhole-port 77
aws:eks:pod-network-latency 78
aws:eks:pod-network-packet-loss 79
aws:eks:terminate-nodegroup-instances 81
ElastiCache Actions Amazon 81
aws:elasticache:replicationgroup-interrupt-az-power 81
AWS Lambda actions 82
aws:lambda:invocation-add-delay 83
aws:lambda:invocation-error 83
aws:lambda:invocation-http-integration-response 84
Actions du réseau 85
aws:network:disrupt-connectivity 85
aws:network:route-table-disrupt-cross-region-connectivity 87
aws:network:transit-gateway-disrupt-cross-region-connectivity 88
Actions Amazon RDS 89
aws:rds:failover-db-cluster 89
aws:rds:reboot-db-instances 90
Actions Amazon S3 90
aws:s3:bucket-pause-replication 91
Actions de Systems Manager 92
aws:ssm:send-command 92
aws:ssm:start-automation-execution 93
Actions relatives aux documents SSM 94
Utilisez l'aws:ssm:send-commandaction 94
Documents AWS FIS SSM préconfigurés 96
Exemples 105

Résolution des problèmes 105
Actions de tâches ECS 105
Actions 106
Limites 107
Prérequis 107
Version de référence du script 110
Exemple de modèle d'expérience 113
Actions du EKS Pod 113
Actions 115
Limites 115
Prérequis 116
Création d'un rôle d'expérience 116
Configuration du compte de service Kubernetes 117
Accorder aux utilisateurs et aux rôles IAM l'accès à Kubernetes APIs 118
Images du conteneur Pod 119
Exemple de modèle d'expérience 121
AWS Lambda actions 122
Actions 123
Limites 123
Prérequis 123
Configuration des fonctions Lambda 125
Configuration d'une AWS FIS expérience 125
Journalisation 126
Rubriques avancées 127
AWS FIS Versions de l'extension Lambda 134
Gestion des modèles d'expériences 138
Création d'un modèle d'expérience 138
Afficher les modèles d'expériences 141
Génération d'un aperçu de la cible 142
Lancer une expérience à partir d'un modèle 143
Mettre à jour un modèle d'expérience 143
Modèles d'expériences de tags 144
Supprimer un modèle d'expérience 145
Exemple de modèles 145
Arrêter EC2 les instances en fonction de filtres 146
Arrêter un nombre spécifié d' EC2 instances 147

Exécuter un document AWS FIS SSM préconfiguré 148
Exécuter un runbook d'automatisation prédéfini 149
Limiter les actions d'API sur les EC2 instances dotées du rôle IAM cible 150
Test de résistance du processeur des pods dans un cluster Kubernetes 151
Gestion des expériences 154
Lancer une expérience 154
Afficher vos expériences 155
États de l'expérience 156
États d'action 156
Marquer une expérience 157
Arrêt d'une expérience 157
Lister les cibles résolues 158
Didacticiels 159
Arrêt et démarrage de l'instance de test 159
Prérequis 159
Étape 1 : Création d'un modèle d'expérience 160
Étape 2 : démarrer l'expérience 163
Étape 3 : suivre la progression de l'expérience 163
Étape 4 : vérifier le résultat de l'expérience 164
Étape 5 : nettoyer 164
Exécuter le stress du processeur sur une instance 165
Prérequis 165
Étape 1 : créer une CloudWatch alarme pour une condition d'arrêt 166
Étape 2 : Création d'un modèle d'expérience 167
Étape 3 : démarrer l'expérience 169
Étape 4 : suivre la progression de l'expérience 170
Étape 5 : Vérifiez les résultats de l'expérience 170
Étape 6 : Nettoyer 164
Interruptions des instances de test Spot 172
Prérequis 173
Étape 1 : Création d'un modèle d'expérience 174
Étape 2 : démarrer l'expérience 177
Étape 3 : suivre la progression de l'expérience 177
Étape 4 : vérifier le résultat de l'expérience 178
Étape 5 : nettoyer 178
Simuler un événement de connectivité 179

Prérequis 180
Étape 1 : Création d'un AWS modèle d'expérience FIS 181
Étape 2 : envoyer un ping à un point de terminaison Amazon S3 182
Étape 3 : Commencez votre AWS expérience FIS 183
Étape 4 : Suivez la progression AWS de votre expérience FIS 183
Étape 5 : vérifier l'interruption du réseau Amazon S3 184
Étape 5 : nettoyer 184
Planifier une expérience récurrente 185
Prérequis 185
Étape 1 : Création d'un rôle et d'une politique IAM 186
Étape 2 : Création d'un Amazon EventBridge planificateur 188
Étape 3 : Vérifiez votre expérience 189
Étape 4 : nettoyer 189
Utilisation de la bibliothèque de scénarios 190
Visualisation d'un scénario 190
Utilisation d'un scénario 191
Exporter un scénario 192
Référence de scénarios 192
AZ Availability: Power Interruption 195
Cross-Region: Connectivity 209
Travailler avec des expériences multi-comptes 222
Concepts 223
Bonnes pratiques 223
Prérequis 224
Autorisations 224
Conditions d'arrêt (facultatif) 227
Leviers de sécurité pour les expériences multi-comptes (en option) 227
Création d'un modèle d'expérience multi-comptes 227
Mettre à jour la configuration d'un compte cible 229
Supprimer une configuration de compte cible 230
Planification des expériences 231
Création d'un rôle de planificateur 231
Création d'un calendrier d'expériences 235
Pour mettre à jour le calendrier à l'aide de la console 236
Mettre à jour le calendrier d'une expérience 236
Désactiver ou supprimer un calendrier d'expérimentation 237

Leviers de sécurité 238
Concepts de leviers de sécurité 238
Ressources sur les leviers de sécurité 239
Utilisation de leviers de sécurité 239
Visualisation d'un levier de sécurité 239
Activation d'un levier de sécurité 240
Débranchement d'un levier de sécurité 240
Expériences de surveillance 242
Surveiller en utilisant CloudWatch 243
Surveiller AWS les expériences FIS 244
AWS Métriques d'utilisation du FIS 244
Surveiller en utilisant EventBridge 245
Enregistrement des expériences 247
Autorisations 247
Schéma du journal 247
Enregistrer les destinations 249
Exemples d'enregistrements de journal 249
Activer la journalisation des expériences 254
Désactiver la journalisation des expériences 255
Enregistrez les appels d'API avec AWS CloudTrail 256
Utiliser CloudTrail 256
Comprendre les AWS entrées du fichier journal FIS 257
Résolution des problèmes 262
Codes d'erreur 262
Sécurité 265
Protection des données 265
Chiffrement au repos 267
Chiffrement en transit 267
Gestion des identités et des accès 267
Public ciblé 268
Authentification par des identités 268
Gestion des accès à l'aide de politiques 272
Comment fonctionne le service d'injection de AWS défauts avec IAM 275
Exemples de politiques 282
Utilisation de rôles liés à un service 292
AWS politiques gérées 295

Sécurité de l'infrastructure 300
AWS PrivateLink 300
Considérations 301
Création d'un point de terminaison d'un VPC d'interface 301
Créer une politique de point de terminaison de VPC 301
Balisage de vos ressources 304
Restrictions de balisage 304
Travailler avec des tags 304
Quotas et limites 306
Historique du document 327
..... cccxxxiii

Qu'est-ce que le service d'injection de AWS défauts ?

AWS Le service d'injection de défauts (AWS FIS) est un service géré qui vous permet de réaliser des expériences d'injection de défauts sur vos charges de AWS travail. L'injection de défauts est basée sur les principes de l'ingénierie du chaos. Ces expériences stressent une application en créant des événements perturbateurs afin que vous puissiez observer la réaction de votre application. Vous pouvez ensuite utiliser ces informations pour améliorer les performances et la résilience de vos applications afin qu'elles se comportent comme prévu.

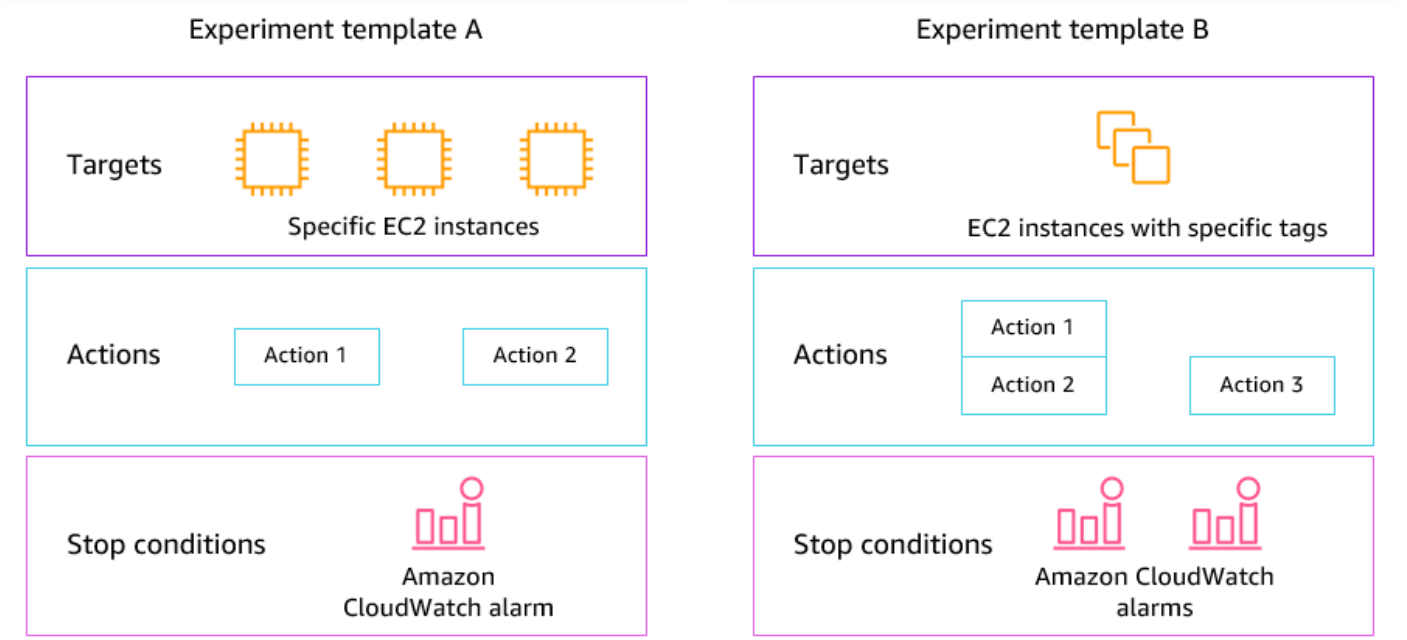
Pour utiliser AWS FIS, vous devez configurer et exécuter des expériences qui vous aident à créer les conditions réelles nécessaires pour détecter les problèmes d'application qui pourraient être difficiles à détecter autrement. AWS Le FIS fournit des modèles qui génèrent des perturbations, ainsi que les commandes et les garde-fous dont vous avez besoin pour exécuter des expériences en production, par exemple en annulant ou en arrêtant automatiquement l'expérience si des conditions spécifiques sont remplies.

> **Important**
>
> AWS FIS réalise des actions réelles sur les AWS ressources réelles de votre système. Par conséquent, avant d'utiliser AWS FIS pour exécuter des expériences en production, nous vous recommandons vivement de terminer une phase de planification et de réaliser les expériences dans un environnement de pré-production.

Pour plus d'informations sur la planification de votre expérience, voir Fiabilité des tests etPlanification de vos AWS expériences FIS. Pour plus d'informations sur le AWS FIS, consultez la section Service d'injection de AWS défauts.

AWS Concepts de la FIS

Pour utiliser le AWS FIS, vous réalisez des expériences sur vos AWS ressources afin de tester votre théorie sur le fonctionnement d'une application ou d'un système en cas de panne. Pour exécuter des expériences, vous devez d'abord créer un modèle d'expérience. Un modèle d'expérience est le plan de votre expérience. Il contient les actions, les cibles et les conditions d'arrêt de l'expérience. Après avoir créé un modèle de test, vous pouvez l'utiliser pour exécuter un test. Pendant que votre expérience est en cours, vous pouvez suivre sa progression et consulter son statut. Une expérience est terminée lorsque toutes les actions de l'expérience ont été exécutées.



Actions

Une action est une activité que le AWS FIS exécute sur une AWS ressource au cours d'une expérience. AWS FIS fournit un ensemble d'actions préconfigurées en fonction du type de AWS ressource. Chaque action est exécutée pendant une durée spécifiée pendant une expérience, ou jusqu'à ce que vous l'arrêtiez. Les actions peuvent être exécutées de manière séquentielle ou simultanée (en parallèle).

Cibles

Une cible est une ou plusieurs AWS ressources sur lesquelles le AWS FIS effectue une action au cours d'une expérience. Vous pouvez choisir des ressources spécifiques ou sélectionner un groupe de ressources en fonction de critères spécifiques, tels que des balises ou un état.

Conditions d'arrêt

AWS FIS fournit les commandes et les garde-fous dont vous avez besoin pour effectuer des expériences en toute sécurité sur vos charges de travail. AWS Une condition d'arrêt est un mécanisme permettant d'arrêter une expérience si celle-ci atteint un seuil que vous définissez comme une CloudWatch alarme Amazon. Si une condition d'arrêt est déclenchée pendant que l'expérience est en cours, le AWS FIS arrête l'expérience.

Soutenu Services AWS

AWS FIS fournit des actions préconfigurées pour des types spécifiques de cibles dans l'ensemble AWS des services. AWS La FIS soutient les actions visant à obtenir des ressources ciblées pour les domaines suivants : Services AWS

- Amazon CloudWatch
- Amazon DynamoDB
- Amazon EBS
- Amazon EC2
- Amazon ECS
- Amazon EKS
- Amazon ElastiCache
- Amazon RDS
- Amazon S3
- AWS Systems Manager
- Amazon VPC

Pour les expériences à compte unique, les ressources cibles doivent être identiques à Compte AWS celles de l'expérience. Vous pouvez exécuter des expériences AWS FIS qui ciblent les ressources d'un autre Compte AWS compte à l'aide d'expériences AWS FIS multi-comptes.

Pour de plus amples informations, veuillez consulter Actions pour la AWS FIS.

Accédez au AWS FIS

Vous pouvez travailler avec AWS FIS de l'une des manières suivantes :

- AWS Management Console— Fournit une interface Web que vous pouvez utiliser pour accéder au AWS FIS. Pour plus d'informations, consultez Utilisation de AWS Management Console.
- AWS Command Line Interface (AWS CLI) — Fournit des commandes pour un large éventail de AWS services, y compris AWS FIS, et est compatible avec Windows, macOS et Linux. Pour de plus amples informations, veuillez consulter AWS Command Line Interface. Pour plus d'informations sur les commandes pour AWS FIS, voir fis dans le manuel de référence des AWS CLI commandes.

- AWS CloudFormation— Créez des modèles qui décrivent vos AWS ressources. Vous utilisez les modèles pour provisionner et gérer ces ressources comme une seule unité. Pour plus d'informations, consultez la référence du type de ressource AWS Fault Injection Service.
- AWS SDKs— Fournit des informations spécifiques à la langue APIs et prend en charge de nombreux détails de connexion, tels que le calcul des signatures, la gestion des nouvelles tentatives de demande et la gestion des erreurs. Pour de plus amples informations, veuillez consulter AWS SDKs.
- API HTTPS — Fournit des actions d'API de bas niveau que vous pouvez appeler à l'aide de requêtes HTTPS. Pour plus d'informations, consultez la référence de l'API du service d'injection de AWS défauts.

Tarification du AWS FIS

Vous êtes facturé par minute pendant laquelle une action est exécutée, du début à la fin, en fonction du nombre de comptes cibles pour votre test. Pour plus d'informations, consultez la section Tarification AWS FIS.

Planification de vos AWS expériences FIS

L'injection de défauts est le processus qui consiste à stresser une application dans des environnements de test ou de production en créant des événements perturbateurs, tels que des pannes de serveur ou une limitation des API. En observant la façon dont le système réagit, vous pouvez ensuite mettre en œuvre des améliorations. Lorsque vous effectuez des tests sur votre système, cela peut vous aider à identifier les faiblesses systémiques de manière contrôlée, avant que ces faiblesses n'affectent les clients qui dépendent de votre système. Vous pouvez ensuite résoudre les problèmes de manière proactive afin d'éviter des résultats imprévisibles.

Avant de commencer à exécuter des expériences d'injection de défauts à l'aide du AWS FIS, nous vous recommandons de vous familiariser avec les principes et directives suivants.

> **Important**
>
> AWS FIS réalise des actions réelles sur les AWS ressources réelles de votre système. Par conséquent, avant de commencer à utiliser AWS FIS pour exécuter des expériences, nous vous recommandons vivement de terminer une phase de planification et un test dans un environnement de pré-production ou de test.

Table des matières

- Principes de base et directives
- Directives de planification des expériences

Principes de base et directives

Avant de commencer les expériences avec AWS FIS, suivez les étapes suivantes :

1. Identifier le déploiement cible pour l'expérience : commencez par identifier le déploiement cible. S'il s'agit de votre première expérience, nous vous recommandons de commencer dans un environnement de pré-production ou de test.
2. Passez en revue l'architecture de l'application : vous devez vous assurer d'avoir identifié tous les composants de l'application, les dépendances et les procédures de restauration pour chaque composant. Commencez par examiner l'architecture de l'application. En fonction de l'application, reportez-vous au AWS Well-Architected Framework.

3. Définissez le comportement permanent de votre système en termes de paramètres techniques et commerciaux importants, tels que la latence, la charge du processeur, les échecs de connexion par minute, le nombre de tentatives ou la vitesse de chargement des pages.
4. Formuler une hypothèse — Formez une hypothèse sur la façon dont vous vous attendez à ce que le comportement du système change au cours de l'expérience. La définition d'une hypothèse suit le format suivant :

   Si elle *`fault injection action`* est effectuée, elle ne *`business or technical metric impact`* doit pas dépasser*`value`*.

   Par exemple, l'hypothèse d'un service d'authentification peut se lire comme suit : « Si la latence du réseau augmente de 10 %, le nombre d'échecs de connexion augmente de moins de 1 % ». Une fois l'expérience terminée, vous évaluez si la résilience des applications correspond à vos attentes commerciales et techniques.

Nous vous recommandons également de suivre ces directives lorsque vous travaillez avec AWS FIS :

- Commencez toujours à expérimenter avec AWS FIS dans un environnement de test. Ne commencez jamais par un environnement de production. Au fur et à mesure que vous progressez dans vos expériences d'injection de défauts, vous pouvez expérimenter dans d'autres environnements contrôlés que l'environnement de test.
- Renforcez la confiance de votre équipe dans la résilience de vos applications en commençant par de petites expériences simples, telles que l'exécution de l'action aws:ec2:stop-instances sur une cible.
- L'injection de défauts peut entraîner de réels problèmes. Procédez avec prudence et assurez-vous que vos premières injections de défauts concernent les instances de test afin qu'aucun client ne soit affecté.
- Testez, testez et testez encore. L'injection de défauts est destinée à être mise en œuvre dans un environnement contrôlé avec des expériences bien planifiées. Cela vous permet de renforcer votre confiance dans les capacités de votre application et de vos outils à résister à des conditions turbulentes.
- Nous vous recommandons vivement de mettre en place un excellent programme de surveillance et d'alerte avant de commencer. Sans cela, vous ne serez pas en mesure de comprendre ou de mesurer l'impact de vos expériences, ce qui est essentiel pour des pratiques durables d'injection de défauts.

Directives de planification des expériences

Avec AWS FIS, vous réalisez des expériences sur vos AWS ressources afin de tester votre théorie sur le fonctionnement d'une application ou d'un système en cas de panne.

Les directives suivantes sont recommandées pour planifier vos expériences AWS FIS.

- Consultez l'historique des pannes : passez en revue les pannes et les événements précédents de votre système. Cela peut vous aider à vous faire une idée de l'état général et de la résilience de votre système. Avant de commencer à effectuer des tests sur votre système, vous devez résoudre les problèmes et faiblesses connus de votre système.
- Identifiez les services ayant le plus d'impact — Passez en revue vos services et identifiez ceux qui ont le plus d'impact sur vos utilisateurs finaux ou vos clients s'ils tombent en panne ou ne fonctionnent pas correctement.
- Identifier le système cible — Le système cible est le système sur lequel vous allez exécuter des expériences. Si vous utilisez AWS FIS pour la première fois ou si vous n'avez jamais effectué d'expériences d'injection de défauts auparavant, nous vous recommandons de commencer par exécuter des expériences sur un système de pré-production ou de test.
- Consultez votre équipe — Demandez-lui ce qui l'inquiète. Vous pouvez formuler une hypothèse pour prouver ou réfuter leurs inquiétudes. Vous pouvez également demander à votre équipe ce qui ne l'inquiète pas. Cette question peut révéler deux erreurs courantes : l'erreur des coûts irrécupérables et l'erreur du biais de confirmation. L'élaboration d'une hypothèse basée sur les réponses de votre équipe peut aider à fournir plus d'informations sur la réalité de l'état de votre système.
- Passez en revue l'architecture de votre application : passez en revue votre système ou votre application et assurez-vous d'avoir identifié tous les composants de l'application, les dépendances et les procédures de restauration pour chaque composant.

  Nous vous recommandons de consulter le AWS Well-Architected Framework. Le framework peut vous aider à créer une infrastructure sécurisée, performante, résiliente et efficace pour vos applications et vos charges de travail. Pour plus d'informations, veuillez consulter AWS Bien architecturé.
- Identifiez les métriques applicables — Vous pouvez surveiller l'impact d'un test sur vos AWS ressources à l'aide des CloudWatch métriques Amazon. Vous pouvez utiliser ces mesures pour déterminer le niveau de référence ou « état stable » lorsque votre application fonctionne de manière optimale. Vous pouvez ensuite surveiller ces mesures pendant ou après l'expérience

afin d'en déterminer l'impact. Pour de plus amples informations, veuillez consulter Surveillez les statistiques d'utilisation du AWS FIS à l'aide d'Amazon CloudWatch.

- Définissez un seuil de performance acceptable pour votre système : identifiez la métrique qui représente un état stable acceptable pour votre système. Vous allez utiliser cette métrique pour créer une ou plusieurs CloudWatch alarmes représentant une condition d'arrêt pour votre expérience. Si l'alarme est déclenchée, l'expérience est automatiquement arrêtée. Pour de plus amples informations, veuillez consulter Conditions d'arrêt pour AWS FIS.

AWS Composants du modèle d'expérience FIS

Vous utilisez les composants suivants pour créer des modèles d'expériences :

Actions

Les actions AWS FIS que vous souhaitez exécuter. Les actions peuvent être exécutées dans un ordre défini que vous spécifiez, ou elles peuvent être exécutées simultanément. Pour de plus amples informations, veuillez consulter Actions.

Cibles

Les AWS ressources sur lesquelles une action spécifique est réalisée. Pour de plus amples informations, veuillez consulter Cibles.

Conditions d'arrêt

Les CloudWatch alarmes qui définissent un seuil à partir duquel les performances de votre application ne sont pas acceptables. Si une condition d'arrêt est déclenchée alors qu'une expérience est en cours, le AWS FIS arrête l'expérience. Pour de plus amples informations, veuillez consulter Conditions d'arrêt.

Rôle d'expérience

Rôle IAM qui accorde à AWS FIS les autorisations nécessaires pour exécuter des expériences en votre nom. Pour de plus amples informations, veuillez consulter Rôle d'expérience.

Configuration du rapport d'expérimentation

Configuration permettant d'activer les rapports d'expérimentation. Pour de plus amples informations, veuillez consulter Configurations de rapports d'expérimentation pour AWS FIS.

Options d'expérimentation

Options pour le modèle d'expérience. Pour de plus amples informations, veuillez consulter Options d'expérimentation pour AWS FIS.

Votre compte possède des quotas liés au AWS FIS. Par exemple, il existe un quota sur le nombre d'actions par modèle d'expérience. Pour de plus amples informations, veuillez consulter Quotas et limites.

Syntaxe du modèle

Voici la syntaxe d'un modèle d'expérience.

```
{
            "description": "string",
            "targets": {},
            "actions": {},
            "stopConditions": [],
            "roleArn": "arn:aws:iam::123456789012:role/AllowFISActions",
            "experimentReportConfiguration":{},
            "experimentOptions":{},
            "tags": {}
        }
```

Pour obtenir des exemples, consultez Exemple de modèles .

Mise en route

Pour créer un modèle d'expérience à l'aide du AWS Management Console, voirCréation d'un modèle d'expérience.

Pour créer un modèle d'expérience à l'aide du AWS CLI, voirExemples de modèles d'expériences AWS FIS.

Actions pour la AWS FIS

Pour créer un modèle de test, vous devez définir une ou plusieurs actions. Pour obtenir la liste des actions prédéfinies fournies par le AWS FIS, voirRéférence des actions.

Vous ne pouvez exécuter une action qu'une seule fois au cours d'une expérience. Pour exécuter la même action AWS FIS plusieurs fois dans le même test, ajoutez-la plusieurs fois au modèle en utilisant des noms différents.

Table des matières

- Syntaxe des actions
- Identifiants d'action
- Paramètres d'action
- Objectifs d'action

- Durée de l'action
- Exemples d'actions

Syntaxe des actions

Voici la syntaxe d'une action.

```
{
    "actions": {
        "action_name": {
            "actionId": "aws:service:action-type",
            "description": "string",
            "parameters": {
                "name": "value"
             },
            "startAfter": ["action_name", ...],
            "targets": {
                "ResourceType": "target_name"
            }
        }
    }
}
```

Lorsque vous définissez une action, vous fournissez les informations suivantes :

action_name

Nom de l'action.

actionId

Identifiant de l'action.

description

Description facultative.

parameters

Tous les paramètres d'action.

startAfter

Toutes les actions qui doivent être terminées avant que cette action puisse commencer. Dans le cas contraire, l'action s'exécute au début de l'expérience.

targets

Toutes les cibles d'action.

Pour obtenir des exemples, consultez the section called "Exemples d'actions".

Identifiants d'action

Chaque action AWS FIS possède un identifiant au format suivant :

```
aws:service-name:action-type
```

Par exemple, l'action suivante arrête les EC2 instances Amazon cibles :

```
aws:ec2:stop-instances
```

Pour une liste complète des actions, consultez leAWS FIS Référence des actions.

Paramètres d'action

Certaines actions AWS FIS comportent des paramètres supplémentaires spécifiques à l'action. Ces paramètres sont utilisés pour transmettre des informations au AWS FIS lorsque l'action est exécutée.

AWS FIS prend en charge les types de pannes personnalisés à l'aide de l'`aws:ssm:send-commandaction`, qui utilise l'agent SSM et un document de commande SSM pour créer la condition de panne sur les instances ciblées. L'`aws:ssm:send-commandaction` inclut un `documentArn` paramètre qui prend le nom de ressource Amazon (ARN) d'un document SSM comme valeur. Vous spécifiez des valeurs pour les paramètres lorsque vous ajoutez l'action à votre modèle d'expérience.

Pour plus d'informations sur la définition des paramètres de l'`aws:ssm:send-commandaction`, consultezUtilisez l'aws:ssm:send-commandaction.

Dans la mesure du possible, vous pouvez saisir une configuration de restauration (également appelée action de post-action) dans les paramètres de l'action. Une post action restaure la cible dans l'état dans lequel elle se trouvait avant l'exécution de l'action. L'action de publication s'exécute après le délai spécifié dans la durée de l'action. Toutes les actions ne peuvent pas prendre en charge les actions de publication. Par exemple, si l'action met fin à une EC2 instance Amazon, vous ne pouvez pas récupérer l'instance une fois qu'elle a été résiliée.

Objectifs d'action

Une action s'exécute sur les ressources cibles que vous spécifiez. Après avoir défini une cible, vous pouvez indiquer son nom lorsque vous définissez une action.

```
"targets": {
    "ResourceType": "resource_name"
}
```

AWS Les actions de la FIS prennent en charge les types de ressources suivants pour les cibles d'action :

- AutoScalingGroups— Groupes Amazon EC2 Auto Scaling
- Compartiments — Compartiments Amazon S3
- Cluster — Clusters Amazon EKS
- Clusters : clusters Amazon ECS ou clusters de base de données Amazon Aurora
- DBInstances— Instances de base de données Amazon RDS
- Instances — EC2 Instances Amazon
- ManagedResources— Les clusters Amazon EKS, les équilibreurs de charge d' EC2 applications et de réseaux Amazon et les groupes Amazon EC2 Auto Scaling activés pour le changement de zone ARC.
- Groupes de nœuds : groupes de nœuds Amazon EKS
- Pods — Pods Kubernetes sur Amazon EKS
- ReplicationGroups— Groupes ElastiCache de réplication
- Rôles — Rôles IAM
- SpotInstances— Instances Amazon EC2 Spot
- Sous-réseaux : sous-réseaux VPC
- Tableaux — Tableaux globaux Amazon DynamoDB
- Tâches — Tâches Amazon ECS
- TransitGateways— Passerelles de transport en commun
- Volumes — Volumes Amazon EBS

Pour obtenir des exemples, consultez the section called “Exemples d'actions”.

Durée de l'action

Si une action inclut un paramètre que vous pouvez utiliser pour spécifier la durée de l'action, par défaut, l'action n'est considérée comme terminée qu'une fois la durée spécifiée écoulée. Si vous avez défini l'option d'`emptyTargetResolutionMode`expérimentation sur`skip`, l'action se terminera immédiatement avec le statut « ignoré » lorsqu'aucune cible n'a été résolue. Par exemple, si vous spécifiez une durée de 5 minutes, AWS FIS considère que l'action est terminée au bout de 5 minutes. Il lance ensuite l'action suivante, jusqu'à ce que toutes les actions soient terminées.

La durée peut être soit la durée pendant laquelle une condition d'action est maintenue, soit la durée pendant laquelle les métriques sont surveillées. Par exemple, la latence est injectée pendant la durée spécifiée. Pour les types d'action quasi instantanés, tels que la mise hors service d'une instance, les conditions d'arrêt sont surveillées pendant la durée spécifiée.

Si une action inclut une action de publication dans les paramètres de l'action, l'action de publication s'exécute une fois l'action terminée. Le temps nécessaire pour terminer l'action de publication peut entraîner un délai entre la durée d'action spécifiée et le début de l'action suivante (ou la fin de l'expérience, si toutes les autres actions sont terminées).

Exemples d'actions

Voici des exemples d'actions.

Exemples

- Arrêter EC2 les instances
- Interruptez les instances ponctuelles
- Perturber le trafic réseau
- Licencier les employés d'E
- Démarrer l'autoshift par zone ARC

Exemple : arrêter EC2 les instances

L'action suivante arrête les EC2 instances identifiées à l'aide de la cible nommée*targetInstances*. Au bout de deux minutes, il redémarre les instances cibles.

```
"actions": {
    "stopInstances": {
```

```
        "actionId": "aws:ec2:stop-instances",
        "parameters": {
            "startInstancesAfterDuration": "PT2M"
        },
        "targets": {
            "Instances": "targetInstances"
        }
    }
}
```

Exemple : Interrupt Spot Instances

L'action suivante arrête les instances Spot identifiées à l'aide de la cible nommée*targetSpotInstances*. Il attend deux minutes avant d'interrompre l'instance Spot.

```
"actions": {
    "interruptSpotInstances": {
        "actionId": "aws:ec2:send-spot-instance-interruptions",
        "parameters": {
            "durationBeforeInterruption": "PT2M"
        },
        "targets": {
            "SpotInstances": "targetSpotInstances"
        }
    }
}
```

Exemple : perturber le trafic réseau

L'action suivante refuse le trafic entre les sous-réseaux cibles et les sous-réseaux des autres zones de disponibilité.

```
"actions": {
    "disruptAZConnectivity": {
        "actionId": "aws:network:disrupt-connectivity",
        "parameters": {
            "scope": "availability-zone",
            "duration": "PT5M"
        },
        "targets": {
            "Subnets": "targetSubnets"
        }
```

```
    }
}
```

Exemple : licencier des employés d'EKS

L'action suivante met fin à 50 % des EC2 instances du cluster EKS identifiées à l'aide de la cible nommée*targetNodeGroups*.

```
"actions": {
    "terminateWorkers": {
        "actionId": "aws:eks:terminate-nodegroup-instances",
        "parameters": {
            "instanceTerminationPercentage": "50"
        },
        "targets": {
            "Nodegroups": "targetNodeGroups"
        }
    }
}
```

Exemple : démarrer l'autoshift par zone ARC

L'action suivante lance un changement automatique par zone ARC qui déplace les ressources gérées de *az-in-parameters* la zone pour. *duration-in-parameteres* Le type de ressource `ManagedResources` est utilisé comme clé pour le nom de la cible dans le modèle d'expérience AWS FIS.

```
{
    "description": "aaa",
    "targets": {
        "ManagedResources-Target-1": {
            "resourceType": "aws:arc:zonal-shift-managed-resource",
            "resourceArns": [
                "arn:aws:elasticloadbalancing:us-east-1:0124567890:loadbalancer/app/
application/11223312312516",
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "arc": {
```

```
            "actionId": "aws:arc:start-zonal-autoshift",
            "parameters": {
                "availabilityZoneIdentifier": "us-east-1a",
                "duration": "PT1M"
            },
            "targets": {
               "ManagedResources": "ManagedResources-Target-1"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none"
        }
    ],
    "roleArn": "arn:aws:iam::718579638765:role/fis",
    "tags": {},
    "experimentOptions": {
        "accountTargeting": "single-account",
        "emptyTargetResolutionMode": "fail"
    }
}
```

Objectifs pour le AWS FIS

Une cible est une ou plusieurs AWS ressources sur lesquelles une action est exécutée par le AWS Fault Injection Service (AWS FIS) au cours d'une expérience. Les cibles peuvent se trouver sur le même compte AWS que le test, ou sur un autre compte dans le cadre d'un test multi-comptes. Pour en savoir plus sur le ciblage des ressources dans un autre compte, consultezTravailler avec des expériences multi-comptes.

Vous définissez des cibles lorsque vous créez un modèle d'expérience. Vous pouvez utiliser la même cible pour plusieurs actions dans votre modèle de test.

AWS Le FIS identifie toutes les cibles au début de l'expérience, avant de lancer l'une des actions définies. AWS Le FIS utilise les ressources cibles qu'il sélectionne pour l'ensemble de l'expérience. Si aucune cible n'est trouvée, l'expérience échoue.

Table des matières

- Syntaxe cible
- Types de ressources

- Identifier les ressources cibles
  - Filtres de ressources
  - Paramètres des ressources
- Mode de sélection
- Exemples de cibles
- Exemples de filtres

Syntaxe cible

Voici la syntaxe d'une cible.

```
{
    "targets": {
        "target_name": {
            "resourceType": "resource-type",
            "resourceArns": [
                "resource-arn"
            ],
            "resourceTags": {
                "tag-key": "tag-value"
            },
            "parameters": {
                "parameter-name": "parameter-value"
            },
            "filters": [
                {
                    "path": "path-string",
                    "values": ["value-string"]
                }
            ],
            "selectionMode": "value"
        }
    }
}
```

Lorsque vous définissez une cible, vous fournissez les informations suivantes :

target_name

Nom de la cible.

resourceType

Type de ressource.

resourceArns

Les Amazon Resource Names (ARN) de ressources spécifiques.

resourceTags

Les balises appliquées à des ressources spécifiques.

parameters

Les paramètres qui identifient les cibles à l'aide d'attributs spécifiques.

filters

Les filtres de ressources couvrent les ressources cibles identifiées à l'aide d'attributs spécifiques.

selectionMode

Mode de sélection pour les ressources identifiées.

Pour obtenir des exemples, consultez the section called "Exemples de cibles".

Types de ressources

Chaque action AWS FIS est exécutée sur un type de AWS ressource spécifique. Lorsque vous définissez une cible, vous devez spécifier exactement un type de ressource. Lorsque vous spécifiez une cible pour une action, celle-ci doit être le type de ressource pris en charge par l'action.

Les types de ressources suivants sont pris en charge par le AWS FIS :

- aws:arc : zonal-shift-managed-resource — Une AWS ressource enregistrée avec ARC zonal shift
- aws:dynamodb:global-table — Une table globale Amazon DynamoDB
- aws:ec2:autoscaling-group — Un groupe Amazon Auto Scaling EC2
- aws:ec2:ebs-volume — Un volume Amazon EBS
- aws:ec2:instance — Une instance Amazon EC2
- aws:ec2:spot-instance — Une instance Amazon Spot EC2
- aws:ec2:subnet — Un sous-réseau Amazon VPC
- aws:ec2:transit-gateway — Une passerelle de transit

- aws:ecs:cluster — Un cluster Amazon ECS
- aws:ecs:task — Une tâche Amazon ECS
- aws:eks:cluster — Un cluster Amazon EKS
- aws:eks:nodegroup — Un groupe de nœuds Amazon EKS
- aws:eks:pod — Un pod Kubernetes
- aws:elasticache:replicationgroup — Un groupe de réplication ElastiCache
- aws:iam:role — Un rôle IAM
- aws:lambda:function — Une fonction AWS Lambda
- aws:rds:cluster — Un cluster de base de données Amazon Aurora
- aws:rds:db — Une instance de base de données Amazon RDS
- aws:s3:bucket — Un compartiment Amazon S3

Identifier les ressources cibles

Lorsque vous définissez une cible dans la console AWS FIS, vous pouvez choisir des AWS ressources spécifiques (d'un type de ressource spécifique) à cibler. Vous pouvez également laisser le AWS FIS identifier un groupe de ressources en fonction des critères que vous fournissez.

Pour identifier vos ressources cibles, vous pouvez spécifier les éléments suivants :

- Ressource IDs : ressource constituée IDs de AWS ressources spécifiques. Toutes les ressources IDs doivent représenter le même type de ressource.
- Balises de ressources : balises appliquées à des AWS ressources spécifiques.
- Filtres de ressources : chemin et valeurs représentant les ressources dotées d'attributs spécifiques. Pour de plus amples informations, veuillez consulter Filtres de ressources.
- Paramètres des ressources : paramètres qui représentent les ressources répondant à des critères spécifiques. Pour de plus amples informations, veuillez consulter Paramètres des ressources.

Considérations

- Vous ne pouvez pas spécifier à la fois un ID de ressource et une étiquette de ressource pour la même cible.
- Vous ne pouvez pas spécifier à la fois un ID de ressource et un filtre de ressources pour la même cible.

- Si vous spécifiez une balise de ressource avec une valeur de balise vide, elle n'est pas équivalente à un caractère générique. Il fait correspondre les ressources qui ont une balise avec la clé de balise spécifiée et une valeur de balise vide.
- Si vous spécifiez plusieurs balises, toutes les balises spécifiées doivent être présentes sur la ressource cible pour qu'elle soit sélectionnée (`AND`).

Filtres de ressources

Les filtres de ressources sont des requêtes qui identifient les ressources cibles en fonction d'attributs spécifiques. AWS FIS applique la requête à la sortie d'une action d'API contenant la description canonique de la AWS ressource, en fonction du type de ressource que vous spécifiez. Les ressources dont les attributs correspondent à la requête sont incluses dans la définition cible.

Chaque filtre est exprimé sous la forme d'un chemin d'attribut et de valeurs possibles. Un chemin est une séquence d'éléments, séparés par des points, qui décrivent le chemin permettant d'atteindre un attribut dans le résultat de l'action Décrire pour une ressource. Chaque période représente l'expansion d'un élément. Chaque élément doit être exprimé en cas de Pascal, même si le résultat de l'action Describe pour une ressource est en cas de chameau. Par exemple, vous devez utiliser`AvailabilityZone`, et non `availablityZone` comme élément d'attribut.

```
"filters": [
    {
        "path": "Component.Component.Component",
        "values": [
            "string"
        ]
    }
],
```

La logique suivante s'applique à tous les filtres de ressources :

- Si plusieurs filtres sont fournis, y compris des filtres ayant le même chemin, tous les filtres doivent correspondre pour qu'une ressource soit sélectionnée : `AND`
- Si plusieurs valeurs sont fournies pour un seul filtre, chaque valeur doit être mise en correspondance pour qu'une ressource soit sélectionnée : `OR`
- Si plusieurs valeurs sont trouvées à l'emplacement du chemin de l'appel d'API de description, une valeur doit être mise en correspondance pour qu'une ressource soit sélectionnée : `OR`

- Pour faire correspondre les paires clé/valeur d'une balise, vous devez plutôt sélectionner les ressources cibles par balises (voir ci-dessus).

Le tableau suivant inclut les actions et AWS CLI commandes d'API que vous pouvez utiliser pour obtenir les descriptions canoniques de chaque type de ressource. AWS FIS exécute ces actions en votre nom pour appliquer les filtres que vous spécifiez. La documentation correspondante décrit les ressources incluses par défaut dans les résultats. Par exemple, la documentation des DescribeInstances états dans lesquels des instances ont récemment été résiliées peut apparaître dans les résultats.

Type de ressource	Action d'API	AWS CLI commande
aws:arc:zonal-shift-managed-resource	ListManagedResources	list-managed-resources
aws:ec2:autoscaling-group	DescribeAutoScalingGroups	describe-auto-scaling-groups
aws:ec2:ebs-volume	DescribeVolumes	describe-volumes
aws:ec2:instance	DescribeInstances	décrire les instances
aws:ec2:subnet	DescribeSubnets	describe-subnets
aws:ec2:transit-gateway	DescribeTransitGateways	describe-transit-gateways
aws:ecs:cluster	DescribeClusters	describe-clusters
aws:ecs:task	DescribeTasks	décrire les tâches
aws:eks:cluster	DescribeClusters	describe-clusters
aws:eks:nodegroup	DescribeNodegroup	describe-nodegroup
aws:elasticache:replication group	DescribeReplicationGroups	describe-replication-groups
aws:iam:role	ListRoles	lister les rôles
aws:lambda:function	ListFunctions	fonctions de liste

Type de ressource	Action d'API	AWS CLI commande
aws:rds:cluster	Décrivez DBClusters	describe-db-clusters
aws:rds:db	Décrivez DBInstances	describe-db-instances
aws:s3:bucket	ListBuckets	list-buckets
aws:dynamodb:global-table	DescribeTable	décrivez-table

Pour obtenir des exemples, consultez the section called "Exemples de filtres".

Paramètres des ressources

Les paramètres des ressources identifient les ressources cibles en fonction de critères spécifiques.

Le type de ressource suivant prend en charge les paramètres.

aws:ec2:ebs-volume

- `availabilityZoneIdentifier`— Le code (par exemple, us-east-1a) de la zone de disponibilité qui contient les volumes cibles.

aws:ec2:subnet

- `availabilityZoneIdentifier`— Le code (par exemple, us-east-1a) ou l'ID AZ (par exemple, use1-az1) de la zone de disponibilité qui contient les sous-réseaux cibles.
- `vpc`— Le VPC qui contient les sous-réseaux cibles. Ne prend pas en charge plus d'un VPC par compte.

aws:ecs:task

- `cluster`— Le cluster qui contient les tâches cibles.
- `service`— Le service qui contient les tâches cibles.

aws:eks:pod

- `availabilityZoneIdentifier` – Facultatif. La zone de disponibilité qui contient les pods cibles. Par exemple, `us-east-1d`. Nous déterminons la zone de disponibilité d'un pod en comparant son adresse IP d'hôte et le CIDR du sous-réseau du cluster.
- `clusterIdentifier` : obligatoire. Le nom ou l'ARN du cluster EKS cible.
- `namespace` : obligatoire. L'espace de noms Kubernetes des pods cibles.

- `selectorType` : obligatoire. Type de sélecteur. Les valeurs possibles sont `labelSelector`, `deploymentName` et `podName`.
- `selectorValue` : obligatoire. La valeur du sélecteur. Cette valeur dépend de la valeur `deselectorType`.
- `targetContainerName` – Facultatif. Le nom du conteneur cible tel que défini dans les spécifications du pod. La valeur par défaut est le premier conteneur défini dans les spécifications de chaque pod cible.

aws:lambda:function

- `functionQualifier` – Facultatif. Version ou alias de la fonction à cibler. Si aucun qualificatif n'est spécifié, toutes les invocations seront prises en compte pour le ciblage. Si un alias avec plusieurs versions est spécifié, toutes les versions incluses dans l'alias seront prises en compte pour le ciblage, à condition qu'elles soient invoquées à l'aide d'un ARN contenant l'alias. Si l'alias spécial `$LATEST` est utilisé, les invocations à l'ARN de la fonction de base et les invocations incluses dans `$LATEST` l'ARN seront prises en compte pour l'injection d'erreurs. Pour plus d'informations sur les versions Lambda, voir Gérer les versions des fonctions Lambda dans le guide de l'utilisateur.AWS Lambda

aws:rds:cluster

- `writerAvailabilityZoneIdentifiers` – Facultatif. Les zones de disponibilité du rédacteur du cluster de base de données. Les valeurs possibles sont les suivantes : une liste d'identifiants de zone de disponibilité séparés par des virgules,. `all`

aws:rds:db

- `availabilityZoneIdentifiers` – Facultatif. Les zones de disponibilité de l'instance de base de données à affecter. Les valeurs possibles sont les suivantes : une liste d'identifiants de zone de disponibilité séparés par des virgules,. `all`

aws:elasticache:replicationgroup

- `availabilityZoneIdentifier` : obligatoire. Le code (par exemple, us-east-1a) ou l'ID AZ (par exemple, use1-az1) de la zone de disponibilité qui contient les nœuds cibles.

Mode de sélection

Vous définissez le périmètre des ressources identifiées en spécifiant un mode de sélection. AWS FIS prend en charge les modes de sélection suivants :

- `ALL`— Exécute l'action sur toutes les cibles.

- `COUNT(n)`— Exécute l'action sur le nombre de cibles spécifié, choisies au hasard parmi les cibles identifiées. Par exemple, COUNT (1) sélectionne l'une des cibles identifiées.
- `PERCENT(n)`— Exécute l'action sur le pourcentage de cibles spécifié, choisi au hasard parmi les cibles identifiées. Par exemple, PERCENT (25) sélectionne 25 % des cibles identifiées.

Si vous avez un nombre impair de ressources et que vous spécifiez 50 %, le AWS FIS arrondit à la valeur inférieure. Par exemple, si vous ajoutez cinq EC2 instances Amazon comme cibles et que le champ d'application est de 50 %, AWS FIS arrondit à deux instances. Vous ne pouvez pas spécifier un pourcentage inférieur à une ressource. Par exemple, si vous ajoutez quatre EC2 instances Amazon et que le champ d'application est de 5 %, AWS FIS ne peut pas sélectionner d'instance.

Si vous définissez plusieurs cibles en utilisant le même type de ressource cible, AWS FIS peut sélectionner la même ressource plusieurs fois.

Quel que soit le mode de sélection que vous utilisez, si l'étendue que vous spécifiez n'identifie aucune ressource, l'expérience échoue.

Exemples de cibles

Vous trouverez ci-dessous des exemples de cibles.

Exemples

- Instances dans le VPC spécifié avec les balises spécifiées
- Tâches avec les paramètres spécifiés

Exemple : instances du VPC spécifié avec les balises spécifiées

Les cibles possibles pour cet exemple sont les EC2 instances Amazon dans le VPC spécifié avec le tag env=prod. Le mode de sélection indique que le AWS FIS choisit l'une de ces cibles au hasard.

```
{
    "targets": {
        "randomInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "env": "prod"
            },
            "filters": [
```

```
                {
                    "path": "VpcId",
                    "values": [
                        "vpc-aabbcc11223344556"
                    ]
                }
            ],
            "selectionMode": "COUNT(1)"
        }
    }
}
```

Exemple : tâches avec les paramètres spécifiés

Les cibles possibles pour cet exemple sont les tâches Amazon ECS avec le cluster et le service spécifiés. Le mode de sélection indique que le AWS FIS choisit l'une de ces cibles au hasard.

```
{
    "targets": {
        "randomTask": {
            "resourceType": "aws:ecs:task",
            "parameters": {
                "cluster": "myCluster",
                "service": "myService"
            },
            "selectionMode": "COUNT(1)"
        }
    }
}
```

Exemples de filtres

Vous trouverez ci-dessous des exemples de filtres.

Exemples

- EC2 instances
- clusters de bases de données

Exemple : EC2 instances

Lorsque vous spécifiez un filtre pour une action qui prend en charge le type de ressource aws:ec2:instance, AWS FIS utilise la commande EC2 describe-instances Amazon et applique le filtre pour identifier les cibles.

La describe-instances commande renvoie une sortie JSON dans laquelle chaque instance est une structure sous laquelle se trouve une structure`Instances`. Ce qui suit est une sortie partielle qui inclut les champs marqués d'un*`italics`*. Nous fournirons des exemples qui utilisent ces champs pour spécifier un chemin d'attribut à partir de la structure de la sortie JSON.

```
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "ImageId": "ami-00111111111111111",
                    "InstanceId": "i-00aaaaaaaaaaaaaa",
                    "InstanceType": "t2.micro",
                    "KeyName": "virginia-kp",
                    "LaunchTime": "2020-09-30T11:38:17.000Z",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-10-0-1-240.ec2.internal",
                    "PrivateIpAddress": "10.0.1.240",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-203-0-113-17.compute-1.amazonaws.com",
                    "PublicIpAddress": "203.0.113.17",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "StateTransitionReason": "",
                    "SubnetId": "subnet-aabbcc11223344556",
                    "VpcId": "vpc-00bbbbbbbbbbbbbbb",
                    ...
                    "NetworkInterfaces": [
                    {
```

```
                        ...
                        "Groups": [
                            {
                                "GroupName": "sec-group-1",
                                "GroupId": "sg-a0011223344556677"
                            },
                            {
                                "GroupName": "sec-group-1",
                                "GroupId": "sg-b9988776655443322"
                            }
                        ],
                        ...
                    },
                    ...
                },
                ...
                {
                    ...
                }
            ],
            "OwnerId": "123456789012",
            "ReservationId": "r-aaaaaabbbbb111111"
        },
        ...
    ]
}
```

Pour sélectionner des instances dans une zone de disponibilité spécifique à l'aide d'un filtre de ressources, spécifiez le chemin d'attribut `AvailabilityZone` et le code de la zone de disponibilité comme valeur. Par exemple :

```
"filters": [
    {
        "path": "Placement.AvailabilityZone",
        "values": [ "us-east-1a" ]
    }
],
```

Pour sélectionner des instances dans un sous-réseau spécifique à l'aide d'un filtre de ressources, spécifiez le chemin d'attribut `SubnetId` et l'ID du sous-réseau comme valeur. Par exemple :

```
"filters": [
```

```
    {
        "path": "SubnetId",
        "values": [ "subnet-aabbcc11223344556" ]
    }
],
```

Pour sélectionner des instances qui se trouvent dans un état d'instance spécifique, spécifiez le chemin d'attribut `Name` et l'un des noms d'état suivants comme valeur : `pending` | `running` | `shutting-down` | `terminated` | `stopping` |`stopped`. Par exemple :

```
"filters": [
    {
        "path": "State.Name",
        "values": [ "running" ]
    }
],
```

Pour sélectionner des instances auxquelles un certain nombre de groupes de sécurité sont attachés, spécifiez un filtre unique avec le chemin d'attribut pour *`GroupId`* et plusieurs groupes de sécurité IDs. Par exemple :

```
"filters": [
    {
        "path": "NetworkInterfaces.Groups.GroupId",
        "values": [
                "sg-a0011223344556677",
                "sg-f1100110011001100"
            ]
    }
],
```

Pour sélectionner des instances auxquelles un certain nombre de groupes de sécurité sont attachés, spécifiez plusieurs filtres avec le chemin d'attribut pour *`GroupId`* et un identifiant de groupe de sécurité unique pour chaque filtre. Par exemple :

```
"filters": [
    {
        "path": "NetworkInterfaces.Groups.GroupId",
        "values": [
            "sg-a0011223344556677"
        ]
```

```
    },
    {
        "path": "NetworkInterfaces.Groups.GroupId",
        "values": [
            "sg-b9988776655443322"
        ]
    }
],
```

Exemple : cluster Amazon RDS (cluster de base de données)

Lorsque vous spécifiez un filtre pour une action qui prend en charge le type de ressource aws:rds:cluster, FIS AWS exécute la describe-db-clusters commande Amazon RDS et applique le filtre pour identifier les cibles.

La describe-db-clusters commande renvoie une sortie JSON similaire à la suivante pour chaque cluster de base de données. Ce qui suit est une sortie partielle qui inclut les champs marqués d'un*italics*. Nous fournirons des exemples qui utilisent ces champs pour spécifier un chemin d'attribut à partir de la structure de la sortie JSON.

```
[
    {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-east-2a",
            "us-east-2b",
            "us-east-2c"
        ],
        "BackupRetentionPeriod": 7,
        "DatabaseName": "",
        "DBClusterIdentifier": "database-1",
        "DBClusterParameterGroup": "default.aurora-postgresql11",
        "DBSubnetGroup": "default-vpc-01234567abc123456",
        "Status": "available",
        "EarliestRestorableTime": "2020-11-13T15:08:32.211Z",
        "Endpoint": "database-1.cluster-example.us-east-2.rds.amazonaws.com",
        "ReaderEndpoint": "database-1.cluster-ro-example.us-east-2.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora-postgresql",
        "EngineVersion": "11.7",
        ...
    }
```

```
]
```

Pour appliquer un filtre de ressources qui renvoie uniquement les clusters de base de données qui utilisent un moteur de base de données spécifique, spécifiez le chemin d'attribut `Engine` et la valeur `aurora-postgresql` comme indiqué dans l'exemple suivant.

```
"filters": [
    {
        "path": "Engine",
        "values": [ "aurora-postgresql" ]
    }
],
```

Pour appliquer un filtre de ressources qui renvoie uniquement les clusters de base de données d'une zone de disponibilité spécifique, spécifiez le chemin et la valeur de l'attribut, comme indiqué dans l'exemple suivant.

```
"filters": [
    {
        "path": "AvailabilityZones",
        "values": [ "us-east-2a" ]
    }
],
```

Conditions d'arrêt pour AWS FIS

AWS Le service d'injection de défauts (AWS FIS) fournit des commandes et des garde-corps vous permettant de réaliser des expériences en toute sécurité sur des charges de travail. AWS Une condition d'arrêt est un mécanisme permettant d'arrêter une expérience si celle-ci atteint un seuil que vous définissez comme une CloudWatch alarme Amazon. Si une condition d'arrêt est déclenchée pendant une expérience, le AWS FIS arrête l'expérience. Vous ne pouvez pas reprendre une expérience interrompue.

Pour créer une condition d'arrêt, définissez d'abord l'état stable de votre application ou de votre service. L'état d'équilibre correspond à une performance optimale de votre application, définie en termes de paramètres commerciaux ou techniques. Par exemple, la latence, la charge du processeur ou le nombre de tentatives. Vous pouvez utiliser l'état permanent pour créer une CloudWatch alarme qui vous permettra d'arrêter une expérience si votre application ou votre service atteint un état dans

lequel ses performances ne sont pas acceptables. Pour plus d'informations, consultez la section Utilisation des CloudWatch alarmes Amazon dans le guide de CloudWatch l'utilisateur Amazon.

Votre compte dispose d'un quota quant au nombre de conditions d'arrêt que vous pouvez spécifier dans un modèle d'expérience. Pour de plus amples informations, veuillez consulter Quotas et limites pour le service d'injection de AWS défauts.

Syntaxe de la condition d'arrêt

Lorsque vous créez un modèle d'expérience, vous spécifiez une ou plusieurs conditions d'arrêt en spécifiant les CloudWatch alarmes que vous avez créées.

```
{
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:region:123456789012:alarm:alarm-name"
        }
    ]
}
```

L'exemple suivant indique que le modèle d'expérience ne spécifie aucune condition d'arrêt.

```
{
    "stopConditions": [
        {
            "source": "none"
        }
    ]
}
```

En savoir plus

Pour un didacticiel expliquant comment créer une CloudWatch alarme et ajouter une condition d'arrêt à un modèle d'expérience, voirExécuter le stress du processeur sur une instance.

Pour plus d'informations sur les CloudWatch métriques disponibles pour les types de ressources pris en charge par le AWS FIS, consultez les rubriques suivantes :

- Surveillez vos instances à l'aide de CloudWatch

- CloudWatch Métriques Amazon ECS
- Surveillance des métriques Amazon RDS à l'aide de CloudWatch
- Surveillance des métriques Run Command à l'aide de CloudWatch

Rôles IAM pour les expériences AWS FIS

AWS Identity and Access Management (IAM) est un AWS service qui aide un administrateur à contrôler en toute sécurité l'accès aux AWS ressources. Pour utiliser le AWS FIS, vous devez créer un rôle IAM qui accorde au AWS FIS les autorisations nécessaires pour que le AWS FIS puisse exécuter des expériences en votre nom. Vous spécifiez ce rôle d'expérience lorsque vous créez un modèle d'expérience. Pour une expérience à compte unique, la politique IAM relative au rôle d'expérience doit autoriser la modification des ressources que vous spécifiez comme cibles dans votre modèle d'expérience. Dans le cas d'un test multi-comptes, le rôle d'expérimentation doit autoriser le rôle d'orchestrateur à assumer le rôle IAM pour chaque compte cible. Pour de plus amples informations, veuillez consulter Autorisations pour les expériences multi-comptes.

Nous vous recommandons de suivre la pratique de sécurité standard consistant à accorder le moindre privilège. Vous pouvez le faire en spécifiant des ressources ARNs ou des balises spécifiques dans vos politiques.

Pour vous aider à démarrer rapidement avec AWS FIS, nous proposons des politiques AWS gérées que vous pouvez spécifier lorsque vous créez un rôle d'essai. Vous pouvez également utiliser ces politiques comme modèle lorsque vous créez vos propres documents de politique en ligne.

Table des matières

- Prérequis
- Option 1 : créer un rôle d'essai et associer une politique AWS gérée
- Option 2 : créer un rôle d'essai et ajouter un document de politique intégré

Prérequis

Avant de commencer, installez AWS CLI et créez la politique de confiance requise.

Installez le AWS CLI

> Avant de commencer, installez et configurez la AWS CLI. Lorsque vous configurez le AWS CLI, vous êtes invité à entrer des AWS informations d'identification. Les exemples de cette procédure

supposent que vous avez également configuré une région par défaut. Sinon, ajoutez l'option `--region` à chaque commande. Pour plus d'informations, consultez Installation ou mise à jour de la AWS CLI et Configuration de la AWS CLI.

Créez une politique de relation de confiance

Un rôle d'expérimentation doit avoir une relation de confiance qui permet au service AWS FIS d'assumer ce rôle. Créez un fichier texte nommé `fis-role-trust-policy.json` et ajoutez-y la politique de relation de confiance suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                  "fis.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Nous vous recommandons d'utiliser les clés de condition `aws:SourceAccount` et `aws:SourceArn` pour vous protéger contre le problème du député confus. Le compte source est le propriétaire de l'expérience et l'ARN source est l'ARN de l'expérience. Par exemple, vous devez ajouter le bloc de conditions suivant à votre politique de confiance.

```
"Condition": {
    "StringEquals": {
        "aws:SourceAccount": "account_id"
    },
    "ArnLike": {
        "aws:SourceArn": "arn:aws:fis:region:account_id:experiment/*"
    }
}
```

Ajouter des autorisations pour assumer les rôles de compte cible (tests multi-comptes uniquement)

Pour les expériences multi-comptes, vous avez besoin d'autorisations permettant au compte d'orchestrateur d'assumer les rôles de compte cible. Vous pouvez modifier l'exemple suivant et l'ajouter en tant que document de politique intégré pour assumer les rôles de compte cible :

```
{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource":[
        "arn:aws:iam::target_account_id:role/role_name"
    ]
}
```

Option 1 : créer un rôle d'essai et associer une politique AWS gérée

Utilisez l'une des politiques AWS gérées par AWS FIS pour démarrer rapidement.

Pour créer un rôle d'essai et y associer une politique AWS gérée

1. Vérifiez qu'il existe une politique gérée pour les actions AWS FIS de votre expérience. Dans le cas contraire, vous devrez plutôt créer votre propre document de politique en ligne. Pour de plus amples informations, veuillez consulter the section called “AWS politiques gérées”.
2. Utilisez la commande create-role suivante pour créer un rôle et ajouter la politique de confiance que vous avez créée dans les conditions préalables.

   ```
   aws iam create-role --role-name my-fis-role --assume-role-policy-document
    file://fis-role-trust-policy.json
   ```

3. Utilisez la attach-role-policycommande suivante pour joindre la politique AWS gérée.

   ```
   aws iam attach-role-policy --role-name my-fis-role --policy-arn fis-policy-arn
   ```

   Où se *fis-policy-arn* trouve l'un des suivants :

   - arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorEC2Access
   - arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorECSAccess
   - arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorEKSAccess

- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorNetworkAccess
- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorRDSAccess
- arn:aws:iam::aws:policy/service-role/AWSFaultInjectionSimulatorSSMAccess

Option 2 : créer un rôle d'essai et ajouter un document de politique intégré

Utilisez cette option pour les actions qui n'ont pas de politique gérée, ou pour inclure uniquement les autorisations requises pour votre expérience spécifique.

Pour créer un test et ajouter un document de politique intégré

1. Utilisez la commande create-role suivante pour créer un rôle et ajouter la politique de confiance que vous avez créée dans les conditions préalables.

```
aws iam create-role --role-name my-fis-role --assume-role-policy-document
 file://fis-role-trust-policy.json
```

2. Créez un fichier texte nommé `fis-role-permissions-policy.json` et ajoutez-y une politique d'autorisation. Pour un exemple que vous pouvez utiliser comme point de départ, consultez ce qui suit.

   - Actions d'injection de défauts : commencez par la politique suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowFISExperimentRoleFaultInjectionActions",
            "Effect": "Allow",
            "Action": [
                "fis:InjectApiInternalError",
                "fis:InjectApiThrottleError",
                "fis:InjectApiUnavailableError"
            ],
            "Resource": "arn:*:fis:*:*:experiment/*"
        }
    ]
}
```

   - Actions Amazon EBS : commencez par la politique suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:PauseVolumeIO"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*"
        }
    ]
}
```

- EC2 Actions Amazon — Commencez par la politique AWSFaultInjectionSimulatorEC2d'accès.
- Actions Amazon ECS : commencez par la AWSFaultInjectionSimulatorECSAccesspolitique.
- Actions Amazon EKS : commencez par la AWSFaultInjectionSimulatorEKSAccesspolitique.
- Actions réseau : commencez par la AWSFaultInjectionSimulatorNetworkAccesspolitique.
- Actions Amazon RDS : commencez par la AWSFaultInjectionSimulatorRDSAccesspolitique.
- Actions de Systems Manager : commencez par la AWSFaultInjectionSimulatorSSMAccesspolitique.

3. Utilisez la put-role-policycommande suivante pour ajouter la politique d'autorisation que vous avez créée à l'étape précédente.

```
aws iam put-role-policy --role-name my-fis-role --policy-name my-fis-policy --
policy-document file://fis-role-permissions-policy.json
```

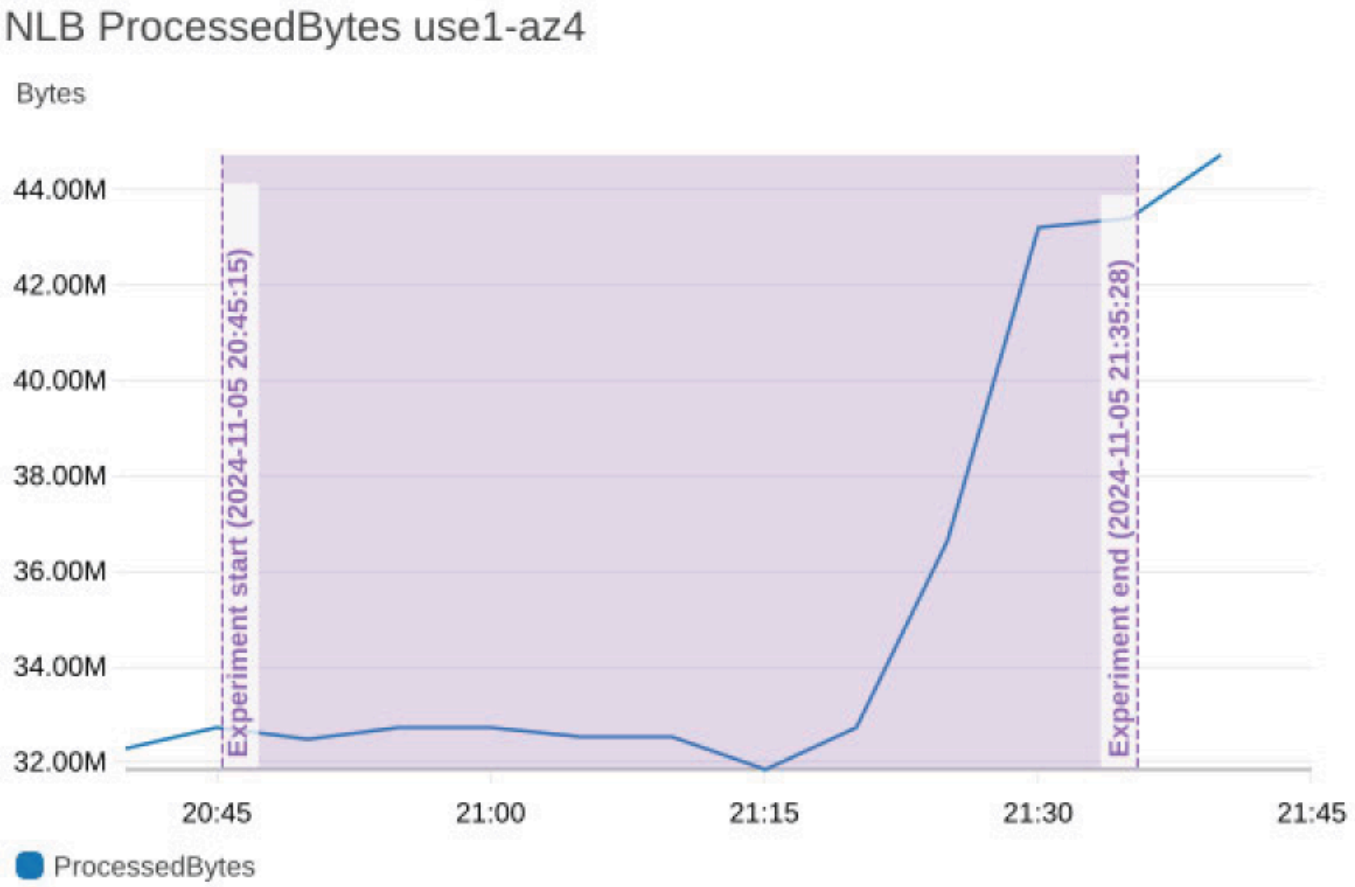
Configurations de rapports d'expérimentation pour AWS FIS

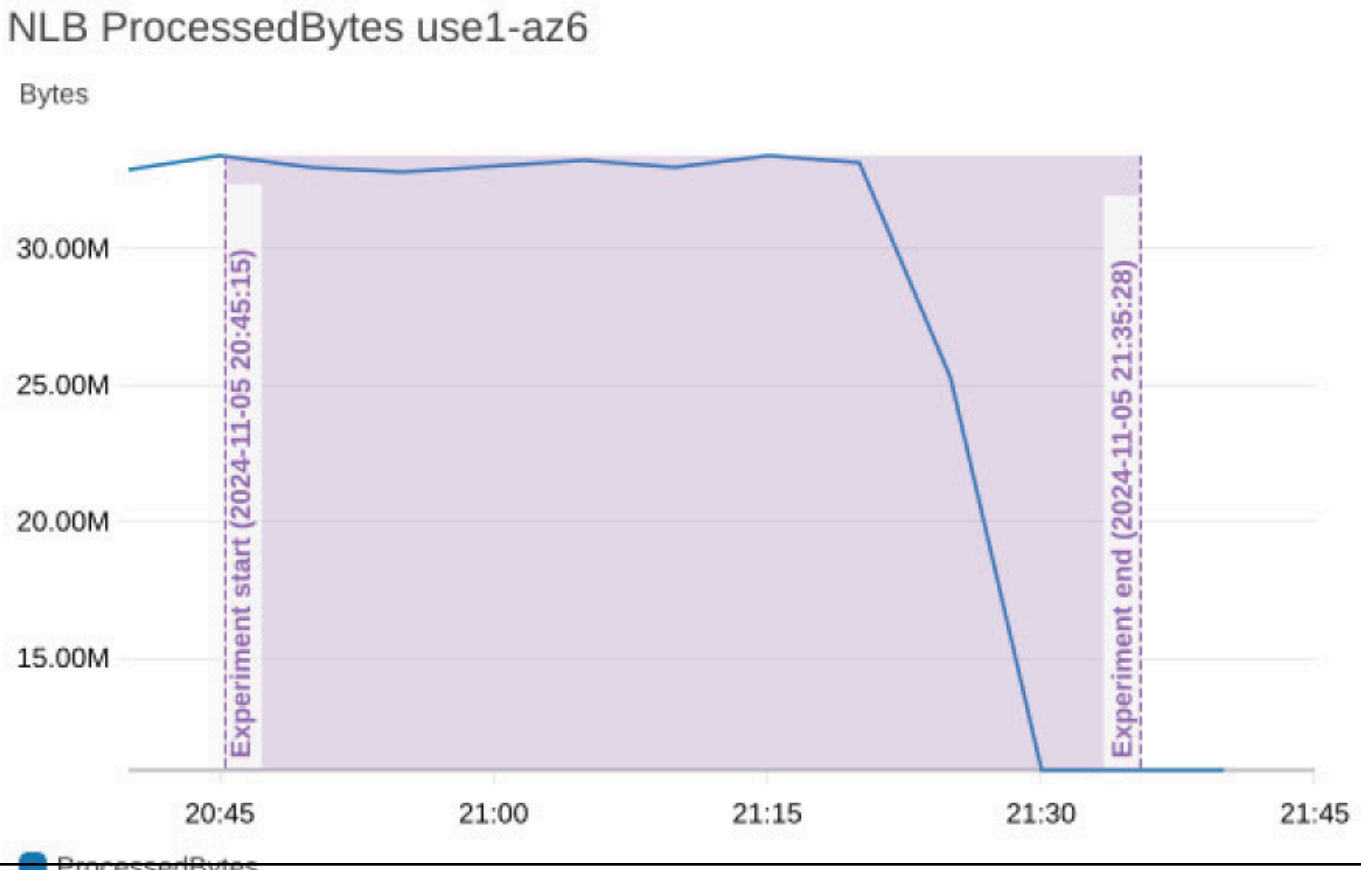
Vous pouvez activer le service d'injection de AWS défauts (FIS) pour générer des rapports pour les expériences, ce qui facilite la production de preuves des tests de résilience. Le rapport de test est un

document PDF qui résume les actions du test et capture éventuellement la réponse de l'application à partir d'un CloudWatch tableau de bord que vous spécifiez. Pour voir un exemple de rapport d'expérience, téléchargez le fichier zip ici.

Pour activer et configurer le contenu du rapport généré pour l'expérience, vous définissez la configuration du rapport d'expérience pour le modèle d'expérience. Lorsque vous spécifiez un CloudWatch tableau de bord, AWS FIS inclut un graphique instantané de tous les widgets du tableau de bord donné, annoté avec les heures de début et de fin de l'expérience sur une durée que vous spécifiez, comme indiqué dans l'exemple ci-dessous.

Cet exemple illustre l'impact d'une expérience de perte de paquets dans une zone de disponibilité (AZ). Lorsque la perte de paquets est introduite dans la zone AZ use1-az6, le trafic passe de use1-az6 à use1-az4, de sorte que le nombre d'octets traités par l'équilibreur de charge dans cette zone de zone diminue.


NLB ProcessedBytes use1-az4
Bytes
44.00M
42.00M
40.00M
38.00M
36.00M
34.00M
32.00M
Experiment start (2024-11-05 20:45:15)
Experiment end (2024-11-05 21:35:28)
20:45
21:00
21:15
21:30
21:45
ProcessedBytes


NLB ProcessedBytes use1-az6
Bytes
30.00M
25.00M
20.00M
15.00M
Experiment start (2024-11-05 20:45:15)
Experiment end (2024-11-05 21:35:28)
20:45
21:00
21:15
21:30
21:45
ProcessedBytes

À la fin de l'expérience, le rapport peut être téléchargé depuis la console AWS FIS et est également stocké dans un compartiment Amazon S3. Si vous incluez un CloudWatch tableau de bord dans la configuration de votre rapport, des images de chaque widget sont également fournies. Aucun rapport n'est généré pour les tests exécutés `cancelled` ou exécutés dans le cadre de l'aperçu de la cible (avec ActionsMode défini sur). `skip-all` Une fois que l'expérience dépasse la limite de conservation des données de l'expérience, le rapport ne sera disponible que depuis le compartiment Amazon S3. AWS Des frais de la FIS s'appliquent pour chaque rapport fourni, à l'exception de ceux qui échouent en raison d'erreurs internes. Pour plus d'informations, consultez les sections Tarification du service d'injection de AWS défauts etQuotas et limites pour le service d'injection de AWS défauts. Des frais d'ingestion et de stockage pour Amazon S3 ainsi que des frais CloudWatch d'API pour GetMetricWidgetImageles GetDashboarddemandes peuvent s'appliquer. Pour plus d'informations, consultez la section Tarification et CloudWatch tarification d'Amazon S3.

Table des matières

- Syntaxe de configuration du rapport d'expérimentation
- Autorisations relatives aux rapports d'expérimentation
- Meilleures pratiques en matière de rapports d'expérimentation

Syntaxe de configuration du rapport d'expérimentation

Voici la syntaxe de la configuration du rapport d'expérience, une section facultative du modèle d'expérience.

```
{
    "experimentReportConfiguration": {
        "outputs": {
            "s3Configuration": {
                "bucketName": "my-bucket-name",
                "prefix": "report-storage-prefix"
            }
        },
        "dataSources": {
            "cloudWatchDashboards": [
                {
                    "dashboardIdentifier": "arn:aws:cloudwatch::123456789012:dashboard/
MyDashboard"
                }
            ]
        },
```

```
        "preExperimentDuration": "PT20M",
        "postExperimentDuration": "PT20M"
    }
}
```

À l'aide du`experimentReportConfiguration`, vous pouvez personnaliser la destination de sortie, les données d'entrée et les fenêtres temporelles pour les données à inclure dans le rapport d'expérience, ce qui peut vous aider à mieux comprendre l'impact et les résultats de vos expériences AWS FIS. Lorsque vous définissez la configuration du rapport d'expérimentation, vous fournissez les informations suivantes :

outputs

Section du `experimentReportConfiguration` qui indique où le rapport d'expérience sera livré. Dans`outputs`, vous spécifiez le `s3Configuration` en fournissant les informations suivantes :

- `bucketName`- Le nom du compartiment Amazon S3 dans lequel le rapport sera stocké. Le seau doit se trouver dans la même région que l'expérience.
- `prefix`(Facultatif) - Un préfixe dans le compartiment Amazon S3 dans lequel le rapport sera stocké. Ce champ est vivement recommandé afin de limiter l'accès au préfixe uniquement.

Sources de données

Section facultative `experimentReportConfiguration` qui spécifie les sources de données supplémentaires qui seront incluses dans le rapport d'expérience.

- `cloudWatchDashboards`- Un tableau de CloudWatch bord qui sera inclus dans le rapport. Limité à un CloudWatch tableau de bord.
- `dashboardIdentifier`- L'ARN du CloudWatch tableau de bord. Des graphiques instantanés de chaque widget du type `metric` indiqué dans ce tableau de bord seront inclus dans le rapport, à l'exception des statistiques interrégionales.

preExperimentDuration

Section facultative `experimentReportConfiguration` qui définit la durée préalable à l'expérience pour les mesures du CloudWatch tableau de bord à inclure dans le rapport, jusqu'à 30 minutes. Il doit s'agir d'une période qui représente l'état d'équilibre de votre application. Par exemple, une durée préalable à l'expérience de 5 minutes signifie que les graphiques instantanés incluront des mesures 5 minutes avant le début de l'expérience. Le format de la durée est ISO 8601 et la valeur par défaut est de 20 minutes.

postExperimentDuration

Section facultative `experimentReportConfiguration` qui définit la durée post-expérience pour les mesures du CloudWatch tableau de bord à inclure dans le rapport, jusqu'à 2 heures. Il doit s'agir d'une durée qui représente l'état d'équilibre ou la période de rétablissement de votre application. Par exemple, si vous spécifiez une durée post-expérience de 5 minutes, les graphiques instantanés incluront des mesures jusqu'à 5 minutes après la fin de l'expérience. Le format de la durée est ISO 8601 et la valeur par défaut est de 20 minutes.

Autorisations relatives aux rapports d'expérimentation

Pour permettre à AWS FIS de générer et de stocker le rapport d'expérience, vous devez autoriser les opérations suivantes à partir de votre rôle IAM d'expérience AWS FIS :

- `cloudwatch:GetDashboard`
- `cloudwatch:GetMetricWidgetImage`
- `s3:GetObject`
- `s3:PutObject`

Nous vous recommandons de suivre les meilleures pratiques en matière de AWS sécurité et de limiter le rôle de test au compartiment et au préfixe. Voici un exemple de déclaration de politique qui restreint l'accès au rôle d'essai.

```
{
    "Version": "2012-10-17",
    "Statement":
        [
            {
                "Action": [
                    "s3:PutObject",
                    "s3:GetObject"
                    ],
                "Resource": "arn:aws:s3:::my-experiment-report-bucket/my-prefix/*",
                "Effect": "Allow"
            },
            {
                "Action": [
                    "cloudwatch:GetDashboard"
                    ],
```

```
                "Resource": "arn:aws:cloudwatch::012345678912:dashboard/my-experiment-
report-dashboard",
                "Effect": "Allow"
            },
            {
                "Action": [
                    "cloudwatch:GetMetricWidgetImage"
                    ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
 }
```

Autorisations supplémentaires pour les rapports transmis aux compartiments Amazon S3 chiffrés à l'aide de clés gérées par le client (CMK)

Si le compartiment Amazon S3 que vous spécifiez `S3Configuration` est chiffré avec CMK, vous devez accorder les autorisations supplémentaires suivantes au rôle d'expérience FIS dans votre politique de clés KMS :

- `kms:GenerateDataKey`
- `kms:Decrypt`

Voici un exemple de déclaration de politique clé KMS qui autorise le rôle d'expérience FIS à écrire des rapports dans des compartiments chiffrés :

```
{
    "Sid": "Allow FIS experiment report",
    "Effect": "Allow",
    "Principal":
    {
        "AWS": [
            "arn:aws:iam::012345678912:role/FISExperimentRole",
        ]
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
        ],
    "Resource": "*"
```

```
    }
```

Meilleures pratiques en matière de rapports d'expérimentation

Les meilleures pratiques relatives à l'utilisation de la configuration du rapport d'expérimentation AWS FIS sont les suivantes :

- Avant de commencer une expérience, générez un aperçu de la cible pour vérifier que votre modèle d'expérience est configuré comme prévu. L'aperçu des cibles vous donnera des informations sur les cibles attendues de votre expérience. Pour en savoir plus, consultez Génération d'un aperçu de la cible à partir d'un modèle d'expérience.
- Le rapport ne doit pas être utilisé pour résoudre les problèmes liés à l'échec des expériences. Utilisez plutôt les journaux d'expériences pour résoudre les erreurs d'expérimentation. Nous vous recommandons de vous fier au rapport uniquement pour les expériences que vous avez déjà effectuées et que vous avez terminées avec succès.
- Limitez l'attribution du rôle IAM à l'expérience et accédez aux objets au compartiment et au préfixe de destination S3. Nous vous recommandons de dédier le préfixe bucket/uniquement aux rapports d'expérimentation AWS FIS et de ne pas autoriser d'autres AWS services à accéder à ce bucket et à ce préfixe.
- Utilisez Amazon S3 Object Lock pour empêcher la suppression ou le remplacement du rapport pendant une durée déterminée ou indéfiniment. Pour en savoir plus, consultez la section Verrouillage d'objets avec Object Lock.
- Si votre CloudWatch tableau de bord se trouve dans un compte distinct au sein de la même région, vous pouvez utiliser CloudWatch l'observabilité entre comptes pour activer votre compte AWS FIS orchestrator en tant que compte de surveillance et le compte distinct en tant que compte source depuis la CloudWatch console ou les commandes Observability Access Manager dans l'API and. AWS CLI Pour en savoir plus, consultez la CloudWatch section Observabilité entre comptes.

Options d'expérimentation pour AWS FIS

Les options d'expérience sont des paramètres facultatifs pour une expérience. Vous pouvez définir certaines options d'expérience sur le modèle d'expérience. Des options d'expérience supplémentaires sont définies lorsque vous commencez l'expérience.

La syntaxe des options de test que vous définissez dans le modèle d'expérience est la suivante.

```
{
```

```
        "experimentOptions": {
        "accountTargeting": "single-account | multi-account",
            "emptyTargetResolutionMode": "fail | skip"
    }
}
```

Si vous ne spécifiez aucune option d'expérience lorsque vous créez le modèle d'expérience, la valeur par défaut de chaque option est utilisée.

La syntaxe des options de test que vous définissez au début de l'expérience est la suivante.

```
{
        "experimentOptions": {
            "actionsMode": "run-all | skip-all"
     }
}
```

Si vous ne spécifiez aucune option d'expérience lorsque vous commencez l'expérience, la valeur par défaut `run-all` est utilisée.

Table des matières

- Ciblage des comptes
- Mode de résolution cible vide
- Mode actions

Ciblage des comptes

Si vous avez plusieurs AWS comptes avec des ressources que vous souhaitez cibler dans le cadre d'un test, vous pouvez définir un test multi-comptes à l'aide de l'option de test de ciblage de comptes. Vous exécutez des tests multi-comptes à partir d'un compte orchestrateur qui ont un impact sur les ressources de plusieurs comptes cibles. Le compte de l'orchestrateur est propriétaire du modèle AWS FIS d'expérience et de l'expérience. Un compte cible est un compte AWS individuel dont les ressources peuvent être affectées par une AWS FIS expérience. Pour de plus amples informations, veuillez consulter Travailler avec des expériences multi-comptes pour AWS FIS.

Vous utilisez le ciblage par compte pour indiquer l'emplacement de vos ressources cibles. Vous pouvez fournir deux valeurs pour le ciblage des comptes :

- compte unique — Par défaut. L'expérience ciblera uniquement les ressources du AWS compte sur lequel l' AWS FIS expérience est exécutée.
- multi-comptes — L'expérience peut cibler les ressources de plusieurs comptes AWS.

Configurations du compte cible

Pour exécuter un test multi-comptes, vous devez définir une ou plusieurs configurations de compte cible. Une configuration de compte cible spécifie l'AccountID, le ROLearn et la description de chaque compte dont les ressources sont ciblées dans l'expérience. Le compte IDs des configurations du compte cible pour un modèle d'expérience doit être unique.

Lorsque vous créez un modèle de test multi-comptes, le modèle d'expérience renvoie un champ en lecture seule`targetAccountConfigurationsCount`, c'est-à-dire le décompte de toutes les configurations de compte cible pour le modèle de test.

La syntaxe d'une configuration de compte cible est la suivante.

```
{
    accountId: "123456789012",
    roleArn: "arn:aws:iam::123456789012:role/AllowFISActions",
    description: "fis-ec2-test"
}
```

Lorsque vous créez une configuration de compte cible, vous fournissez les informations suivantes :

accountId

ID de compte AWS à 12 chiffres du compte cible.

roleArn

Un rôle IAM octroyant AWS FIS des autorisations pour effectuer des actions sur le compte cible.

description

Description facultative.

Pour en savoir plus sur l'utilisation des configurations de comptes cibles, consultezTravailler avec des expériences multi-comptes pour AWS FIS.

Mode de résolution cible vide

Ce mode vous donne la possibilité d'autoriser les expériences à se terminer même lorsqu'une ressource cible n'est pas résolue.

- fail — Par défaut. Si aucune ressource n'est résolue pour la cible, l'expérience est immédiatement terminée avec un statut de`failed`.
- skip — Si aucune ressource n'est résolue pour la cible, l'expérience se poursuit et toutes les actions sans cibles résolues sont ignorées. Les actions dont les cibles sont définies à l'aide d'identifiants uniques, tels que ARNs, ne peuvent pas être ignorées. Si aucune cible définie à l'aide d'un identifiant unique n'est trouvée, l'expérience est immédiatement terminée avec un statut de `failed`

Mode actions

Le mode Actions est un paramètre facultatif que vous pouvez spécifier lorsque vous démarrez une expérience. Vous pouvez configurer le mode actions `skip-all` pour générer un aperçu de la cible avant d'injecter des défauts dans vos ressources cibles. L'aperçu de la cible vous permet de vérifier les points suivants :

- Que vous avez configuré votre modèle de test pour cibler les ressources que vous attendez. Les ressources réellement ciblées lorsque vous démarrez cette expérience peuvent être différentes de celles de l'aperçu, car les ressources peuvent être supprimées, mises à jour ou échantillonnées de manière aléatoire.
- Que vos configurations de journalisation sont correctement configurées.
- Que pour les expériences multi-comptes, vous avez correctement configuré un rôle IAM pour chacune des configurations de votre compte cible.

> **Note**
>
> Le `skip-all` mode ne vous permet pas de vérifier que vous disposez des autorisations nécessaires pour exécuter le AWS FIS test et effectuer des actions sur vos ressources.

Le paramètre du mode actions accepte les valeurs suivantes :

- `run-all`- (Par défaut) L'expérience prendra des mesures sur les ressources cibles.

- `skip-all`- L'expérience ignorera toutes les actions sur les ressources cibles.

Pour en savoir plus sur la façon de définir le paramètre du mode actions lorsque vous démarrez une expérience, consultezGénération d'un aperçu de la cible à partir d'un modèle d'expérience.

AWS FIS Référence des actions

Une action est l'activité d'injection de défauts que vous exécutez sur une cible à l'aide de AWS Fault Injection Service (AWS FIS). AWS FIS fournit des actions préconfigurées pour des types spécifiques de cibles dans l'ensemble des AWS services. Vous ajoutez des actions à un modèle de test, que vous utilisez ensuite pour exécuter des tests.

Cette référence décrit les actions courantes dans AWS FIS, notamment des informations sur les paramètres de l'action et les autorisations IAM requises. Vous pouvez également répertorier les AWS FIS actions prises en charge à l'aide de la AWS FIS console ou de la commande list-actions depuis AWS Command Line Interface ()AWS CLI. Une fois que vous avez le nom d'une action spécifique, vous pouvez afficher des informations détaillées sur cette action à l'aide de la commande get-action. Pour plus d'informations sur l'utilisation des AWS FIS commandes avec le AWS CLI, consultez le guide de AWS Command Line Interface l'utilisateur et consultez le manuel de référence des AWS CLI commandes.

Pour plus d'informations sur le fonctionnement AWS FIS des actions, reportez-vous Actions pour la AWS FIS aux sections etComment fonctionne le service d'injection de AWS défauts avec IAM.

Actions

- Actions d'injection de défauts
- Action de rétablissement
- Attendre une action
- CloudWatch Actions Amazon
- Actions Amazon DynamoDB
- Actions Amazon EBS
- EC2 Actions Amazon
- Actions d'Amazon ECS
- Actions d'Amazon EKS
- ElastiCache Actions Amazon
- AWS Lambda actions
- Actions du réseau
- Actions Amazon RDS
- Actions Amazon S3

- Actions de Systems Manager
- Utiliser les documents SSM de Systems Manager avec FIS AWS
- Utilisez les actions AWS FIS aws:ecs:task
- Utilisez les actions AWS FIS aws:eks:pod
- Utilisez les actions AWS FIS aws:lambda:function

Actions d'injection de défauts

AWS FIS prend en charge les actions d'injection de défauts suivantes.

Actions

- aws:fis:inject-api-internal-error
- aws:fis:inject-api-throttle-error
- aws:fis:inject-api-unavailable-error

aws:fis:inject-api-internal-error

Injecte des erreurs internes dans les demandes effectuées par le rôle IAM cible. La réponse spécifique dépend de chaque service et de chaque API. Pour plus d'informations, consultez la documentation du SDK et de l'API de votre service.

Type de ressource

- aws:iam:role

Paramètres

- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- service— L'espace de noms de AWS l'API cible. La valeur prise en charge est `ec2`.
- percentage— Le pourcentage (1 à 100) d'appels dans lesquels le défaut a été injecté.
- operations— Les opérations dans lesquelles injecter le défaut sont séparées par des virgules. Pour obtenir la liste des actions d'API pour l'espace de `ec2` noms, consultez Actions dans le manuel Amazon EC2 API Reference.

Autorisations

- `fis:InjectApiInternalError`

aws:fis:inject-api-throttle-error

Injecte des erreurs de régulation dans les demandes effectuées par le rôle IAM cible. La réponse spécifique dépend de chaque service et de chaque API. Pour plus d'informations, consultez la documentation du SDK et de l'API de votre service.

Type de ressource

- aws:iam:role

Paramètres

- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- service— L'espace de noms de AWS l'API cible. La valeur prise en charge est `ec2`.
- percentage— Le pourcentage (1 à 100) d'appels dans lesquels le défaut a été injecté.
- operations— Les opérations dans lesquelles injecter le défaut sont séparées par des virgules. Pour obtenir la liste des actions d'API pour l'espace de `ec2` noms, consultez Actions dans le manuel Amazon EC2 API Reference.

Autorisations

- `fis:InjectApiThrottleError`

aws:fis:inject-api-unavailable-error

Injecte des erreurs non disponibles dans les demandes effectuées par le rôle IAM cible. La réponse spécifique dépend de chaque service et de chaque API. Pour plus d'informations, consultez la documentation du SDK et de l'API de votre service.

Type de ressource

- aws:iam:role

Paramètres

- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- service— L'espace de noms de AWS l'API cible. La valeur prise en charge est `ec2`.
- percentage— Le pourcentage (1 à 100) d'appels dans lesquels le défaut a été injecté.
- operations— Les opérations dans lesquelles injecter le défaut sont séparées par des virgules. Pour obtenir la liste des actions d'API pour l'espace de `ec2` noms, consultez Actions dans le manuel Amazon EC2 API Reference.

Autorisations

- `fis:InjectApiUnavailableError`

Action de rétablissement

Les actions de restauration sont effectuées pour atténuer les risques ou protéger les applications en cas de panne.

AWS FIS prend en charge les actions de restauration suivantes.

aws:arc:start-zonal-autoshift

Déplace automatiquement le trafic des ressources prises en charge hors d'une zone de disponibilité (AZ) potentiellement altérée et le redirige vers une zone saine AZs dans la même région AWS. Cela permet de faire l'expérience de l'autoshift zonal via FIS. L'autoshift zonal est une fonctionnalité d'Amazon Application Recovery Controller (ARC) qui permet AWS de transférer le trafic d'une ressource hors d'une zone de zone de disponibilité, en votre nom, lorsqu'il est AWS déterminé qu'une défaillance est susceptible d'affecter les clients de la zone de zone de disponibilité.

Lorsque vous exécutez l'`aws:arc:start-zonal-autoshift`action, AWS FIS gère le décalage de zone à l'aide du StartZonalShift UpdateZonalShift, et CancelZonalShift APIs en définissant le

`expiresIn` champ pour ces demandes sur 1 minute comme mécanisme de sécurité. Cela permet d' AWS FIS annuler rapidement le décalage de zone en cas d'événements imprévus tels que des pannes de réseau ou des problèmes du système. Dans la console ARC, le champ du délai d'expiration affiche AWS FIS-managed, et l'expiration prévue réelle est déterminée par la durée spécifiée dans l'action de changement de zone.

Type de ressource

- aws:arc:zonal-shift-managed-resource

  Les ressources gérées par transfert zonal sont des types de ressources tels que les clusters Amazon EKS, les Amazon EC2 Application and Network Load Balancers et les groupes Amazon EC2 Auto Scaling qui peuvent être activés pour l'autoshift zonal ARC. Pour plus d'informations, consultez les ressources prises en charge et l'activation des ressources de décalage automatique zonales dans le Guide du développeur ARC.

Paramètres

- duration— La durée pendant laquelle le trafic sera transféré. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- availabilityZoneIdentifier— Le trafic s'éloigne de cette AZ. Il peut s'agir d'un nom AZ (us-east-1a) ou d'un ID AZ (use1-az1).
- managedResourceTypes— Les types de ressources à partir desquels le trafic sera transféré, séparés par des virgules. Les options possibles sont `ASG` (Auto Scaling Group), `ALB` (Application Load Balancer), `NLB` (Network Load Balancer) et `EKS` (Amazon EKS).
- zonalAutoshiftStatus— L'`zonalAutoshiftStatus`état des ressources que vous souhaitez cibler. Les options possibles sont `ENABLEDDISABLED`, et`ANY`. L'argument par défaut est `ENABLED`.

Autorisations

- arc-zonal-shift:StartZonalShift
- arc-zonal-shift:GetManagedResource
- arc-zonal-shift:UpdateZonalShift
- arc-zonal-shift:CancelZonalShift
- arc-zonal-shift:ListManagedResources

- mise à l'échelle automatique : DescribeTags
- étiquette : GetResources

Attendre une action

AWS FIS prend en charge l'action d'attente suivante.

aws:fis:wait

Exécute l'action d' AWS FIS attente.

Paramètres

- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- Aucun

CloudWatch Actions Amazon

AWS FIS prend en charge l' CloudWatch action Amazon suivante.

aws:cloudwatch:assert-alarm-state

Vérifie que les alarmes spécifiées sont dans l'un des états d'alarme spécifiés.

Type de ressource

- Aucun

Paramètres

- alarmArns— Les ARNs alarmes, séparées par des virgules. Vous pouvez définir jusqu'à cinq alarmes.

- alarmStates— Les états d'alarme, séparés par des virgules. Les états d'alarme possibles sont `OK`ALARM, et`INSUFFICIENT_DATA`.

Autorisations

- `cloudwatch:DescribeAlarms`

Actions Amazon DynamoDB

AWS FIS prend en charge l'action Amazon DynamoDB suivante.

aws:dynamodb:global-table-pause-replication

Suspend la réplication de table globale Amazon DynamoDB vers n'importe quelle table de réplication. Les tables peuvent continuer à être répliquées jusqu'à 5 minutes après le début de l'action.

L'instruction suivante sera ajoutée dynamiquement à la politique pour la table globale DynamoDB cible :

```
{
   "Statement":[
      {
         "Sid": "DoNotModifyFisDynamoDbPauseReplicationEXPxxxxxxxxxxxxxx",
         "Effect":"Deny",
         "Principal":{
            "AWS":"arn:aws:iam::123456789012:role/aws-service-role/
replication.dynamodb.amazonaws.com/AWSServiceRoleForDynamoDBReplication"
         },
         "Action":[
            "dynamodb:GetItem",
            "dynamodb:PutItem",
            "dynamodb:UpdateItem",
            "dynamodb:DeleteItem",
            "dynamodb:DescribeTable",
            "dynamodb:UpdateTable",
            "dynamodb:Scan",
            "dynamodb:DescribeTimeToLive",
            "dynamodb:UpdateTimeToLive"
         ],
         "Resource":"arn:aws:dynamodb:us-east-1:123456789012:table/ExampleGlobalTable",
```

```
         "Condition": {
            "DateLessThan": {
            "aws:CurrentTime": "2024-04-10T09:51:41.511Z"
         }
       }
      }
   ]
}
```

L'instruction suivante sera ajoutée dynamiquement à la politique de flux pour la table globale DynamoDB cible :

```
{
   "Statement":[
      {
         "Sid": "DoNotModifyFisDynamoDbPauseReplicationEXPxxxxxxxxxxxxxx",
         "Effect":"Deny",
         "Principal":{
            "AWS":"arn:aws:iam::123456789012:role/aws-service-role/
replication.dynamodb.amazonaws.com/AWSServiceRoleForDynamoDBReplication"
         },
         "Action":[
            "dynamodb:GetRecords",
            "dynamodb:DescribeStream",
            "dynamodb:GetShardIterator"
         ],
         "Resource":"arn:aws:dynamodb:us-east-1:123456789012:table/ExampleGlobalTable/
stream/2023-08-31T09:50:24.025",
         "Condition": {
            "DateLessThan": {
            "aws:CurrentTime": "2024-04-10T09:51:41.511Z"
         }
      }
   ]
}
```

Si aucune politique de ressources n'est attachée à une table ou à un flux cible, une politique de ressources est créée pour la durée de l'expérience et automatiquement supprimée à la fin de l'expérience. Dans le cas contraire, l'instruction d'erreur est insérée dans une politique existante, sans aucune modification supplémentaire des déclarations de stratégie existantes. La déclaration d'erreur est ensuite supprimée de la politique à la fin de l'expérience.

Type de ressource

- aws:dynamodb:global-table

Paramètres

- duration— Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- `dynamodb:PutResourcePolicy`
- `dynamodb:DeleteResourcePolicy`
- `dynamodb:GetResourcePolicy`
- `dynamodb:DescribeTable`
- `tag:GetResources`

Actions Amazon EBS

AWS FIS prend en charge l'action Amazon EBS suivante.

aws:ebs:pause-volume-io

Suspend I/O les opérations sur les volumes EBS cibles. Les volumes cibles doivent se trouver dans la même zone de disponibilité et doivent être attachés à des instances basées sur le système Nitro. Les volumes ne peuvent pas être attachés à des instances d'un Outpost.

Pour lancer l'expérience à l'aide de la EC2 console Amazon, consultez la section Fault testing on Amazon EBS dans le guide de l' EC2 utilisateur Amazon.

Type de ressource

- aws:ec2:ebs-volume

Paramètres

- duration— La durée, d'une seconde à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute, PT5 S représente cinq secondes et PT6 H représente six heures. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures. Si la durée est courte, par exemple PT5 S, elle I/O est suspendue pendant la durée spécifiée, mais la fin de l'expérience peut prendre plus de temps en raison du temps nécessaire à son initialisation.

Autorisations

- `ec2:DescribeVolumes`
- `ec2:PauseVolumeIO`
- `tag:GetResources`

EC2 Actions Amazon

AWS FIS prend en charge les EC2 actions Amazon suivantes.

Actions

- aws:ec2:api-insufficient-instance-capacity-error
- aws:ec2:asg-insufficient-instance-capacity-error
- aws:ec2:reboot-instances
- aws:ec2:send-spot-instance-interruptions
- aws:ec2:stop-instances
- aws:ec2:terminate-instances

AWS FIS prend également en charge les actions d'injection de défauts via l'agent AWS Systems Manager SSM. Systems Manager utilise un document SSM qui définit les actions à effectuer sur les EC2 instances. Vous pouvez utiliser votre propre document pour injecter des erreurs personnalisées, ou vous pouvez utiliser des documents SSM préconfigurés. Pour de plus amples informations, veuillez consulter the section called “Actions relatives aux documents SSM”.

aws:ec2:api-insufficient-instance-capacity-error

Injecte des réponses `InsufficientInstanceCapacity` d'erreur aux demandes effectuées par les rôles IAM cibles. Les opérations prises en charge sont les CreateFleet appels RunInstances CreateCapacityReservation StartInstances,,. Les demandes qui incluent des demandes de capacité dans plusieurs zones de disponibilité ne sont pas prises en charge. Cette action ne permet pas de définir des cibles à l'aide de balises de ressources, de filtres ou de paramètres.

Type de ressource

- aws:iam:role

Paramètres

- duration— Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- availabilityZoneIdentifiers— Liste des zones de disponibilité séparées par des virgules. Supporte les noms de zone IDs (par exemple`"use1-az1, use1-az2"`) et de zone (par exemple`"us-east-1a"`).
- percentage— Le pourcentage (1 à 100) d'appels dans lesquels le défaut a été injecté.

Autorisations

- `ec2:InjectApiError`avec une `ec2:FisActionId` valeur de clé de condition définie sur `aws:ec2:api-insufficient-instance-capacity-error` et `ec2:FisTargetArns` une clé de condition définie pour cibler les rôles IAM.

Pour un exemple de politique, consultez Exemple : utilisez des clés de condition pour `ec2:InjectApiError`.

aws:ec2:asg-insufficient-instance-capacity-error

Injecte des réponses `InsufficientInstanceCapacity` d'erreur aux demandes effectuées par les groupes Auto Scaling cibles. Cette action prend uniquement en charge les groupes Auto Scaling utilisant des modèles de lancement. Pour en savoir plus sur les erreurs liées à une capacité d'instance insuffisante, consultez le guide de EC2 l'utilisateur Amazon.

Type de ressource

- aws:ec2:autoscaling-group

Paramètres

- duration— Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- availabilityZoneIdentifiers— Liste des zones de disponibilité séparées par des virgules. Supporte les noms de zone IDs (par exemple`"use1-az1, use1-az2"`) et de zone (par exemple`"us-east-1a"`).
- percentage : facultatif. Pourcentage (1 à 100) de demandes de lancement du groupe Auto Scaling cible pour injecter le défaut. La valeur par défaut est 100.

Autorisations

- `ec2:InjectApiError`avec la clé de condition ec2 : FisActionId valeur définie sur `aws:ec2:asg-insufficient-instance-capacity-error` et clé de `ec2:FisTargetArns` condition définie pour cibler les groupes Auto Scaling.
- `autoscaling:DescribeAutoScalingGroups`

Pour un exemple de politique, consultez Exemple : utilisez des clés de condition pour `ec2:InjectApiError`.

aws:ec2:reboot-instances

Exécute l'action d' EC2 API Amazon RebootInstancessur les EC2 instances cibles.

Type de ressource

- aws:ec2:instance

Paramètres

- Aucun

Autorisations

- `ec2:RebootInstances`
- `ec2:DescribeInstances`

AWS politique gérée

- AWSFaultInjectionSimulatorEC2Accès

aws:ec2:send-spot-instance-interruptions

Interrompt les instances Spot cibles. Envoie un avis d'interruption des instances Spot aux instances Spot cibles deux minutes avant de les interrompre. Le temps d'interruption est déterminé par le durationBeforeInterruptionparamètre spécifié. Deux minutes après l'heure d'interruption, les instances Spot sont résiliées ou arrêtées, en fonction de leur comportement d'interruption. Une instance Spot qui est interrompue par AWS FIS reste à l'arrêt tant que vous ne la redémarrez pas.

Immédiatement après le lancement de l'action, l'instance cible reçoit une recommandation de rééquilibrage d'EC2 instance. Si vous l'avez spécifié durationBeforeInterruption, il peut y avoir un délai entre la recommandation de rééquilibrage et l'avis d'interruption.

Pour de plus amples informations, veuillez consulter the section called "Interruptions des instances de test Spot". Sinon, pour lancer l'expérience à l'aide de la EC2 console Amazon, consultez la section Initiate a Spot Instance interruption dans le guide de EC2 l'utilisateur Amazon.

Type de ressource

- aws:ec2:spot-instance

Paramètres

- durationBeforeInterruption— Le temps d'attente avant d'interrompre l'instance, de 2 à 15 minutes. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT2 M représente deux minutes. Dans la AWS FIS console, vous entrez le nombre de minutes.

Autorisations

- `ec2:SendSpotInstanceInterruptions`

- `ec2:DescribeInstances`

AWS politique gérée

- AWSFaultInjectionSimulatorEC2Accès

aws:ec2:stop-instances

Exécute l'action d' EC2 API Amazon StopInstancessur les EC2 instances cibles.

Type de ressource

- aws:ec2:instance

Paramètres

- startInstancesAfterDuration : facultatif. Le temps d'attente avant de démarrer l'instance, compris entre une minute et 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures. Si l'instance possède un volume EBS chiffré, vous devez AWS FIS autoriser la clé KMS utilisée pour chiffrer le volume, ou ajouter le rôle d'expérience à la politique de clé KMS.
- completeIfInstancesTerminated : facultatif. Si c'est vrai, et si `startInstancesAfterDuration` c'est également vrai, cette action n'échouera pas lorsque les EC2 instances ciblées ont été résiliées par une demande distincte en dehors de FIS et ne peuvent pas être redémarrées. Par exemple, les groupes Auto Scaling peuvent mettre fin aux EC2 instances arrêtées sous leur contrôle avant que cette action ne soit terminée. La valeur par défaut est false.

Autorisations

- `ec2:StopInstances`
- `ec2:StartInstances`
- `ec2:DescribeInstances` : facultatif. Obligatoire avec completeIfInstancesTerminated pour valider l'état de l'instance à la fin de l'action.
- `kms:CreateGrant` : facultatif. Requis avec startInstancesAfterDuration pour redémarrer les instances avec des volumes chiffrés.

AWS politique gérée

- AWSFaultInjectionSimulatorEC2Accès

aws:ec2:terminate-instances

Exécute l'action d' EC2 API Amazon TerminateInstancessur les EC2 instances cibles.

Type de ressource

- aws:ec2:instance

Paramètres

- Aucun

Autorisations

- `ec2:TerminateInstances`
- `ec2:DescribeInstances`

AWS politique gérée

- AWSFaultInjectionSimulatorEC2Accès

Actions d'Amazon ECS

AWS FIS prend en charge les actions Amazon ECS suivantes.

Actions

- aws:ecs:drain-container-instances
- aws:ecs:stop-task
- aws:ecs:task-cpu-stress
- aws:ecs:task-io-stress
- aws:ecs:task-kill-process
- aws:ecs:task-network-blackhole-port

- aws:ecs:task-network-latency
- aws:ecs:task-network-packet-loss

aws:ecs:drain-container-instances

Exécute l'action d'API Amazon ECS UpdateContainerInstancesStatepour drainer le pourcentage spécifié d' EC2 instances Amazon sous-jacentes sur les clusters cibles.

Type de ressource

- aws:ecs:cluster

Paramètres

- drainagePercentage— Le pourcentage (1-100).
- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- `ecs:DescribeClusters`
- `ecs:UpdateContainerInstancesState`
- `ecs:ListContainerInstances`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorECSAccess

aws:ecs:stop-task

Exécute l'action d'API Amazon ECS StopTaskpour arrêter la tâche cible.

Type de ressource

- aws:ecs:task

Paramètres

- Aucun

Autorisations

- `ecs:DescribeTasks`
- `ecs:ListTasks`
- `ecs:StopTask`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorECSAccess

aws:ecs:task-cpu-stress

Réduit le stress du processeur sur les tâches cibles. Utilise le document AWSFIS-RunSSM - CPU-Stress. Les tâches doivent être gérées par AWS Systems Manager. Pour de plus amples informations, veuillez consulter Actions de tâches ECS.

Type de ressource

- aws:ecs:task

Paramètres

- duration— La durée du test de stress, au format ISO 8601.
- percent : facultatif. Pourcentage de charge cible, compris entre 0 (aucune charge) et 100 (pleine charge). La valeur par défaut est 100.
- workers : facultatif. Le nombre de facteurs de stress à utiliser. La valeur par défaut est 0, qui utilise tous les facteurs de stress.
- installDependencies : facultatif. Si cette valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur le conteneur annexe pour l'agent SSM, si elles ne sont pas déjà installées. L'argument par défaut est `True`. La dépendance eststress-ng.

Autorisations

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-io-stress

Fait I/O peser du stress sur les tâches cibles. Utilise le document AWSFIS-Run-IO-Stress SSM. Les tâches doivent être gérées par AWS Systems Manager. Pour de plus amples informations, veuillez consulter Actions de tâches ECS.

Type de ressource

- aws:ecs:task

Paramètres

- duration— La durée du test de stress, au format ISO 8601.
- percent : facultatif. Pourcentage d'espace libre sur le système de fichiers à utiliser pendant le test de stress. La valeur par défaut est de 80 %.
- workers : facultatif. Le nombre de workers. Les travailleurs effectuent une combinaison d' read/write opérations séquentielles, aléatoires et mappées en mémoire, de synchronisation forcée et de suppression du cache. Plusieurs processus enfants exécutent différentes I/O opérations sur le même fichier. La valeur par défaut est 1.
- installDependencies : facultatif. Si cette valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur le conteneur annexe pour l'agent SSM, si elles ne sont pas déjà installées. L'argument par défaut est `True`. La dépendance eststress-ng.

Autorisations

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-kill-process

Arrête le processus spécifié dans les tâches à l'aide de la killall commande. Utilise le AWSFIS-Rundocument SSM -Kill-Process. La définition de la tâche doit être `pidMode` définie sur`task`. Les tâches doivent être gérées par AWS Systems Manager. Pour de plus amples informations, veuillez consulter Actions de tâches ECS.

Type de ressource

- aws:ecs:task

Paramètres

- processName— Nom du processus à arrêter.
- signal : facultatif. Le signal à envoyer avec la commande. Les valeurs possibles sont `SIGTERM` (que le récepteur peut choisir d'ignorer) et `SIGKILL` (qui ne peuvent pas être ignorées). La valeur par défaut est `SIGTERM`.
- installDependencies – Facultatif. Si cette valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur le conteneur annexe pour l'agent SSM, si elles ne sont pas déjà installées. L'argument par défaut est `True`. La dépendance estkillall.

Autorisations

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-network-blackhole-port

Supprime le trafic entrant ou sortant pour le protocole et le port spécifiés, en utilisant les points de terminaison Amazon ECS Fault Injection. Utilise le document AWSFIS-RunSSM -Network-Blackhole-Port-ECS. La définition de la tâche doit être `pidMode` définie sur`task`. Les tâches doivent être gérées par AWS Systems Manager. Vous ne pouvez pas `networkMode` définir ce paramètre `bridge` dans la définition de la tâche. Pour de plus amples informations, veuillez consulter Actions de tâches ECS.

Lorsqu'il `useEcsFaultInjectionEndpoints` est défini sur`false`, le défaut utilise l'`iptables`outil et utilise le document SSM AWSFIS-Run-Network-Blackhole-Port.

Type de ressource

- aws:ecs:task

Paramètres

- duration— La durée du test, au format ISO 8601.
- port— Le numéro de port.
- trafficType— Le type de trafic. Les valeurs possibles sont `ingress` et `egress`.
- protocol : facultatif. Protocole. Les valeurs possibles sont `tcp` et udp. La valeur par défaut est `tcp`.
- installDependencies – Facultatif. Si cette valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur le conteneur annexe pour l'agent SSM, si elles ne sont pas déjà installées. L'argument par défaut est `True`. Les dépendances sont atdcurl-minimal, dig etjq.
- useEcsFaultInjectionEndpoints : facultatif. Si ce paramètre est défini sur true, l'injection de défauts Amazon ECS APIs sera utilisée. La valeur par défaut est false.

Autorisations

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-network-latency

Ajoute de la latence et de l'instabilité à l'interface réseau pour le trafic sortant vers des sources spécifiques, en utilisant les points de terminaison Amazon ECS Fault Injection. Utilise le document AWSFIS-RunSSM -Network-Latency-ECS. La définition de la tâche doit être `pidMode` définie sur`task`. Les tâches doivent être gérées par AWS Systems Manager. Vous ne pouvez pas `networkMode` définir ce paramètre `bridge` dans la définition de la tâche. Pour de plus amples informations, veuillez consulter Actions de tâches ECS.

Lorsque `useEcsFaultInjectionEndpoints` ce paramètre est défini sur`false`, le défaut utilise l'`tc`outil et utilise le document AWSFIS-RunSSM -Network-Latency-Sources.

Type de ressource

- aws:ecs:task

Paramètres

- duration— La durée du test, au format ISO 8601.
- delayMilliseconds : facultatif. Le délai, en millisecondes. La valeur par défaut est 200.
- jitterMilliseconds : facultatif. L'instabilité, en millisecondes. La valeur par défaut est 10.
- sources : facultatif. Les sources, séparées par des virgules, sans espaces. Les valeurs possibles sont les suivantes : une IPv4 adresse, un bloc IPv4 CIDR, un nom de domaine et`S3`. `DYNAMODB` Si vous spécifiez `DYNAMODB` ou`S3`, cela ne s'applique qu'au point de terminaison régional de la région actuelle. La valeur par défaut est 0.0.0.0/0, qui correspond à l'ensemble du trafic. IPv4
- installDependencies : facultatif. Si cette valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur le conteneur annexe pour l'agent SSM, si elles ne sont pas déjà installées. L'argument par défaut est `True`. Les dépendances sont atdcurl-minimal,dig, jq etlsof.
- useEcsFaultInjectionEndpoints : facultatif. Si ce paramètre est défini sur true, l'injection de défauts Amazon ECS APIs sera utilisée. La valeur par défaut est false.

Autorisations

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

aws:ecs:task-network-packet-loss

Ajoute la perte de paquets à l'interface réseau pour le trafic sortant vers des sources spécifiques, en utilisant les points de terminaison Amazon ECS Fault Injection. Utilise le document AWSFIS-RunSSM -Network-Packet-Loss-ECS. La définition de la tâche doit être `pidMode` définie sur`task`. Les tâches doivent être gérées par AWS Systems Manager. Vous ne pouvez pas `networkMode` définir ce paramètre `bridge` dans la définition de la tâche. Pour de plus amples informations, veuillez consulter Actions de tâches ECS.

Lorsque `useEcsFaultInjectionEndpoints` ce paramètre est défini sur`false`, le défaut utilise l'`tc`outil et utilise le document SSM AWSFIS-Run-Network-Packet-Loss-Sources.

Type de ressource

- aws:ecs:task

Paramètres

- duration— La durée du test, au format ISO 8601.
- lossPercent : facultatif. Pourcentage de perte de paquets. La valeur par défaut est de 7 %.
- sources : facultatif. Les sources, séparées par des virgules, sans espaces. Les valeurs possibles sont les suivantes : une IPv4 adresse, un bloc IPv4 CIDR, un nom de domaine et`S3`. `DYNAMODB` Si vous spécifiez `DYNAMODB` ou`S3`, cela ne s'applique qu'au point de terminaison régional de la région actuelle. La valeur par défaut est 0.0.0.0/0, qui correspond à l'ensemble du trafic. IPv4
- installDependencies : facultatif. Si cette valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur le conteneur annexe pour l'agent SSM, si elles ne sont pas déjà installées. L'argument par défaut est `True`. Les dépendances sont atdcurl-minimal,dig, jq etlsof.
- useEcsFaultInjectionEndpoints : facultatif. Si ce paramètre est défini sur true, l'injection de défauts Amazon ECS APIs sera utilisée. La valeur par défaut est false.

Autorisations

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

Actions d'Amazon EKS

AWS FIS prend en charge les actions Amazon EKS suivantes.

Actions

- aws:eks:inject-kubernetes-custom-resource
- aws:eks:pod-cpu-stress

- aws:eks:pod-delete
- aws:eks:pod-io-stress
- aws:eks:pod-memory-stress
- aws:eks:pod-network-blackhole-port
- aws:eks:pod-network-latency
- aws:eks:pod-network-packet-loss
- aws:eks:terminate-nodegroup-instances

aws:eks:inject-kubernetes-custom-resource

Exécute une expérience ChaosMesh ou Litmus sur un seul cluster cible. Vous devez installer ChaosMesh Litmus sur le cluster cible.

Lorsque vous créez un modèle d'expérience et définissez un type de cible`aws:eks:cluster`, vous devez cibler cette action sur un seul Amazon Resource Name (ARN). Cette action ne permet pas de définir des cibles à l'aide de balises de ressources, de filtres ou de paramètres.

Lors de l'installation ChaosMesh, vous devez spécifier le runtime du conteneur approprié. À partir de la version 1.23 d'Amazon EKS, le runtime par défaut est passé de Docker à. containerd À partir de la version 1.24, Docker a été supprimé.

Type de ressource

- aws:eks:cluster

Paramètres

- kubernetesApiVersion— La version API de la ressource personnalisée Kubernetes. Les valeurs possibles sont `chaos-mesh.org/v1alpha1` |`litmuschaos.io/v1alpha1`.
- kubernetesKind— Le type de ressource personnalisée Kubernetes. La valeur dépend de la version de l'API.
  - `chaos-mesh.org/v1alpha1`— Les valeurs possibles sont `AWSChaos` `DNSChaos` `GCPChaos` `HTTPChaos` | `IOChaos` | `JVMChaos` | `KernelChaos` `NetworkChaos` | `PhysicalMachineChaos` | `PodChaos` `PodHttpChaos` | `PodIOChaos` | `PodNetworkChaos` | `Schedule` `StressChaos` | `TimeChaos` |

  - `litmuschaos.io/v1alpha1`— La valeur possible est`ChaosEngine`.
- kubernetesNamespace— L'espace de noms Kubernetes.
- kubernetesSpec— `spec` Section de la ressource personnalisée Kubernetes, au format JSON.
- maxDuration— La durée maximale autorisée pour l'exécution de l'automatisation, comprise entre une minute et 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

Aucune autorisation AWS Identity and Access Management (IAM) n'est requise pour cette action. Les autorisations requises pour utiliser cette action sont contrôlées par Kubernetes à l'aide de l'autorisation RBAC. Pour plus d'informations, consultez la section Utilisation de l'autorisation RBAC dans la documentation officielle de Kubernetes. Pour plus d'informations sur Chaos Mesh, consultez la documentation officielle de Chaos Mesh. Pour plus d'informations sur Litmus, consultez la documentation officielle de Litmus.

aws:eks:pod-cpu-stress

Exerce un stress du processeur sur les pods cibles. Pour de plus amples informations, veuillez consulter Actions du EKS Pod.

Type de ressource

- aws:eks:pod

Paramètres

- duration— La durée du test de stress, au format ISO 8601.
- percent : facultatif. Pourcentage de charge cible, compris entre 0 (aucune charge) et 100 (pleine charge). La valeur par défaut est 100.
- workers : facultatif. Le nombre de facteurs de stress à utiliser. La valeur par défaut est 0, qui utilise tous les facteurs de stress.
- kubernetesServiceAccount— Le compte de service Kubernetes. Pour plus d'informations sur les autorisations requises, consultez la rubrique the section called “Configuration du compte de service Kubernetes”.

- fisPodContainerImage : facultatif. L'image du conteneur utilisée pour créer le module d'injection défectueux. Par défaut, les images fournies par AWS FIS. Pour de plus amples informations, veuillez consulter the section called "Images du conteneur Pod".
- maxErrorsPercent – Facultatif. Pourcentage de cibles susceptibles de tomber en panne avant l'échec de l'injection de défauts. La valeur par défaut est 0.
- fisPodLabels : facultatif. Les étiquettes Kubernetes attachées au pod d'orchestration des pannes créé par FIS.
- fisPodAnnotations : facultatif. Les annotations Kubernetes associées au pod d'orchestration d'erreurs créé par FIS.
- fisPodSecurityPolicy : facultatif. La politique des normes de sécurité Kubernetes à utiliser pour le module d'orchestration des pannes créé par FIS et les conteneurs éphémères. Les valeurs possibles sont`privileged`, `baseline` et`restricted`. Cette action est compatible avec tous les niveaux de politique.

Autorisations

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-delete

Supprime les pods cibles. Pour de plus amples informations, veuillez consulter Actions du EKS Pod.

Type de ressource

- aws:eks:pod

Paramètres

- gracePeriodSeconds : facultatif. Durée, en secondes, pendant laquelle le module doit se terminer correctement. Si la valeur est 0, nous exécutons l'action immédiatement. Si la valeur est nulle, nous utilisons le délai de grâce par défaut pour le pod.
- kubernetesServiceAccount— Le compte de service Kubernetes. Pour plus d'informations sur les autorisations requises, consultez la rubrique the section called "Configuration du compte de service Kubernetes".
- fisPodContainerImage : facultatif. L'image du conteneur utilisée pour créer le module d'injection défectueux. Par défaut, les images fournies par AWS FIS. Pour de plus amples informations, veuillez consulter the section called "Images du conteneur Pod".
- maxErrorsPercent – Facultatif. Pourcentage de cibles susceptibles de tomber en panne avant l'échec de l'injection de défauts. La valeur par défaut est 0.
- fisPodLabels : facultatif. Les étiquettes Kubernetes attachées au pod d'orchestration des pannes créé par FIS.
- fisPodAnnotations : facultatif. Les annotations Kubernetes associées au pod d'orchestration d'erreurs créé par FIS.
- fisPodSecurityPolicy : facultatif. La politique des normes de sécurité Kubernetes à utiliser pour le module d'orchestration des pannes créé par FIS et les conteneurs éphémères. Les valeurs possibles sont`privileged`, `baseline` et`restricted`. Cette action est compatible avec tous les niveaux de politique.

Autorisations

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-io-stress

Exerce du I/O stress sur les capsules cibles. Pour de plus amples informations, veuillez consulter Actions du EKS Pod.

Type de ressource

- aws:eks:pod

Paramètres

- duration— La durée du test de stress, au format ISO 8601.
- workers : facultatif. Le nombre de workers. Les travailleurs effectuent une combinaison d' read/write opérations séquentielles, aléatoires et mappées en mémoire, de synchronisation forcée et de suppression du cache. Plusieurs processus enfants exécutent différentes I/O opérations sur le même fichier. La valeur par défaut est 1.
- percent : facultatif. Pourcentage d'espace libre sur le système de fichiers à utiliser pendant le test de stress. La valeur par défaut est de 80 %.
- kubernetesServiceAccount— Le compte de service Kubernetes. Pour plus d'informations sur les autorisations requises, consultez la rubrique the section called “Configuration du compte de service Kubernetes”.
- fisPodContainerImage : facultatif. L'image du conteneur utilisée pour créer le module d'injection défectueux. Par défaut, les images fournies par AWS FIS. Pour de plus amples informations, veuillez consulter the section called “Images du conteneur Pod”.
- maxErrorsPercent – Facultatif. Pourcentage de cibles susceptibles de tomber en panne avant l'échec de l'injection de défauts. La valeur par défaut est 0.
- fisPodLabels : facultatif. Les étiquettes Kubernetes attachées au pod d'orchestration des pannes créé par FIS.
- fisPodAnnotations : facultatif. Les annotations Kubernetes associées au pod d'orchestration d'erreurs créé par FIS.
- fisPodSecurityPolicy : facultatif. La politique des normes de sécurité Kubernetes à utiliser pour le module d'orchestration des pannes créé par FIS et les conteneurs éphémères. Les valeurs possibles sont`privileged`, `baseline` et`restricted`. Cette action est compatible avec tous les niveaux de politique.

Autorisations

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-memory-stress

Exerce un stress de mémoire sur les pods cibles. Pour de plus amples informations, veuillez consulter Actions du EKS Pod.

Type de ressource

- aws:eks:pod

Paramètres

- duration— La durée du test de stress, au format ISO 8601.
- workers : facultatif. Le nombre de facteurs de stress à utiliser. La valeur par défaut est 1.
- percent : facultatif. Pourcentage de mémoire virtuelle à utiliser pendant le test de stress. La valeur par défaut est de 80 %.
- kubernetesServiceAccount— Le compte de service Kubernetes. Pour plus d'informations sur les autorisations requises, consultez la rubrique the section called “Configuration du compte de service Kubernetes”.
- fisPodContainerImage : facultatif. L'image du conteneur utilisée pour créer le module d'injection défectueux. Par défaut, les images fournies par AWS FIS. Pour de plus amples informations, veuillez consulter the section called “Images du conteneur Pod”.
- maxErrorsPercent – Facultatif. Pourcentage de cibles susceptibles de tomber en panne avant l'échec de l'injection de défauts. La valeur par défaut est 0.
- fisPodLabels : facultatif. Les étiquettes Kubernetes attachées au pod d'orchestration des pannes créé par FIS.

- fisPodAnnotations : facultatif. Les annotations Kubernetes associées au pod d'orchestration d'erreurs créé par FIS.
- fisPodSecurityPolicy : facultatif. La politique des normes de sécurité Kubernetes à utiliser pour le module d'orchestration des pannes créé par FIS et les conteneurs éphémères. Les valeurs possibles sont`privileged`, `baseline` et`restricted`. Cette action est compatible avec tous les niveaux de politique.

Autorisations

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-network-blackhole-port

Supprime le trafic entrant ou sortant pour le protocole et le port spécifiés. Compatible uniquement avec la politique relative aux normes de sécurité de Kubernetes. `privileged` Pour de plus amples informations, veuillez consulter Actions du EKS Pod.

Type de ressource

- aws:eks:pod

Paramètres

- duration— La durée du test, au format ISO 8601.
- protocol— Le protocole. Les valeurs possibles sont `tcp` et udp.
- trafficType— Le type de trafic. Les valeurs possibles sont `ingress` et `egress`.
- port— Le numéro de port.
- kubernetesServiceAccount— Le compte de service Kubernetes. Pour plus d'informations sur les autorisations requises, consultez la rubrique the section called “Configuration du compte de service Kubernetes”.

- fisPodContainerImage : facultatif. L'image du conteneur utilisée pour créer le module d'injection défectueux. Par défaut, les images fournies par AWS FIS. Pour de plus amples informations, veuillez consulter the section called "Images du conteneur Pod".
- maxErrorsPercent – Facultatif. Pourcentage de cibles susceptibles de tomber en panne avant l'échec de l'injection de défauts. La valeur par défaut est 0.
- fisPodLabels : facultatif. Les étiquettes Kubernetes attachées au pod d'orchestration des pannes créé par FIS.
- fisPodAnnotations : facultatif. Les annotations Kubernetes associées au pod d'orchestration d'erreurs créé par FIS.

Autorisations

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-network-latency

Ajoute de la latence et de l'instabilité à l'interface réseau à l'aide de l'tcoutil pour le trafic à destination ou en provenance de sources spécifiques. Compatible uniquement avec la politique relative aux normes de sécurité de Kubernetes. `privileged` Pour de plus amples informations, veuillez consulter Actions du EKS Pod.

Type de ressource

- aws:eks:pod

Paramètres

- duration— La durée du test, au format ISO 8601.
- interface : facultatif. L'interface réseau. La valeur par défaut est `eth0`.

- delayMilliseconds – Facultatif. Le délai, en millisecondes. La valeur par défaut est 200.
- jitterMilliseconds : facultatif. L'instabilité, en millisecondes. La valeur par défaut est 10.
- sources : facultatif. Les sources, séparées par des virgules, sans espaces. Les valeurs possibles sont les suivantes : une IPv4 adresse, un bloc IPv4 CIDR, un nom de domaine et`S3`. `DYNAMODB` Si vous spécifiez `DYNAMODB` ou`S3`, cela ne s'applique qu'au point de terminaison régional de la région actuelle. La valeur par défaut est 0.0.0.0/0, qui correspond à l'ensemble du trafic. IPv4
- kubernetesServiceAccount— Le compte de service Kubernetes. Pour plus d'informations sur les autorisations requises, consultez la rubrique the section called “Configuration du compte de service Kubernetes”.
- fisPodContainerImage : facultatif. L'image du conteneur utilisée pour créer le module d'injection défectueux. Par défaut, les images fournies par AWS FIS. Pour de plus amples informations, veuillez consulter the section called “Images du conteneur Pod”.
- maxErrorsPercent – Facultatif. Pourcentage de cibles susceptibles de tomber en panne avant l'échec de l'injection de défauts. La valeur par défaut est 0.
- fisPodLabels : facultatif. Les étiquettes Kubernetes attachées au pod d'orchestration des pannes créé par FIS.
- fisPodAnnotations : facultatif. Les annotations Kubernetes associées au pod d'orchestration d'erreurs créé par FIS.

Autorisations

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:pod-network-packet-loss

Ajoute la perte de paquets à l'interface réseau à l'aide de l'tcoutil. Compatible uniquement avec la politique relative aux normes de sécurité de Kubernetes. `privileged` Pour de plus amples informations, veuillez consulter Actions du EKS Pod.

Type de ressource

- aws:eks:pod

Paramètres

- duration— La durée du test, au format ISO 8601.
- interface : facultatif. L'interface réseau. La valeur par défaut est `eth0`.
- lossPercent – Facultatif. Pourcentage de perte de paquets. La valeur par défaut est de 7 %.
- sources : facultatif. Les sources, séparées par des virgules, sans espaces. Les valeurs possibles sont les suivantes : une IPv4 adresse, un bloc IPv4 CIDR, un nom de domaine et`S3`. `DYNAMODB` Si vous spécifiez `DYNAMODB` ou`S3`, cela ne s'applique qu'au point de terminaison régional de la région actuelle. La valeur par défaut est 0.0.0.0/0, qui correspond à l'ensemble du trafic. IPv4
- kubernetesServiceAccount— Le compte de service Kubernetes. Pour plus d'informations sur les autorisations requises, consultez la rubrique the section called “Configuration du compte de service Kubernetes”.
- fisPodContainerImage : facultatif. L'image du conteneur utilisée pour créer le module d'injection défectueux. Par défaut, les images fournies par AWS FIS. Pour de plus amples informations, veuillez consulter the section called “Images du conteneur Pod”.
- maxErrorsPercent – Facultatif. Pourcentage de cibles susceptibles de tomber en panne avant l'échec de l'injection de défauts. La valeur par défaut est 0.
- fisPodLabels : facultatif. Les étiquettes Kubernetes attachées au pod d'orchestration des pannes créé par FIS.
- fisPodAnnotations : facultatif. Les annotations Kubernetes associées au pod d'orchestration d'erreurs créé par FIS.

Autorisations

- `eks:DescribeCluster`
- `ec2:DescribeSubnets`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

aws:eks:terminate-nodegroup-instances

Exécute l'action d' EC2 API Amazon TerminateInstancessur le groupe de nœuds cible.

Type de ressource

- aws:eks:nodegroup

Paramètres

- instanceTerminationPercentage— Le pourcentage (1 à 100) d'instances à terminer.

Autorisations

- `ec2:DescribeInstances`
- `ec2:TerminateInstances`
- `eks:DescribeNodegroup`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorEKSAccess

ElastiCache Actions Amazon

AWS FIS prend en charge l' ElastiCache action suivante.

aws:elasticache:replicationgroup-interrupt-az-power

Interrompt l'alimentation des nœuds de la zone de disponibilité spécifiée pour les groupes de ElastiCache réplication cibles lorsque le mode multi-AZ est activé. Une seule zone de disponibilité par groupe de réplication peut être affectée à la fois. Lorsqu'un nœud principal est ciblé, la réplique de lecture correspondante présentant le moins de retard de réplication est promue au rang principal. Les remplacements de répliques en lecture dans la zone de disponibilité spécifiée sont bloqués pendant la durée de cette action, ce qui signifie que les groupes de réplication cibles fonctionnent avec une capacité réduite. La cible de cette action prend en charge les moteurs Redis et Valkey. L'action ne prend pas en charge l'option de déploiement « sans serveur ».

Type de ressource

- aws:elasticache:replicationgroup

Paramètres

- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- `elasticache:InterruptClusterAzPower`
- `elasticache:DescribeReplicationGroups`
- `tag:GetResources`

> **Note**
>
> L'action d'alimentation ElastiCache Interrupt AZ prend désormais en charge tous les types de groupes de réplication, y compris Valkey et Redis. Pour mieux représenter cette fonctionnalité, l'action a été renommée. Si vous l'utilisez actuellement`aws:elasticache:interrupt-cluster-az-power`, nous vous recommandons de migrer vers la nouvelle action `aws:elasticache:replicationgroup-interrupt-az-power` afin de tirer parti des dernières fonctionnalités.

AWS Lambda actions

AWS Lambda prend en charge les actions Lambda suivantes

Actions

- aws:lambda:invocation-add-delay
- aws:lambda:invocation-error
- aws:lambda:invocation-http-integration-response

aws:lambda:invocation-add-delay

Retarde le démarrage d'une fonction pendant le nombre de millisecondes que vous spécifiez. L'effet de cette action est similaire à celui des démarrages à froid Lambda, mais le temps supplémentaire est utilisé dans le cadre de la durée facturée et est appliqué à tous les environnements d'exécution au lieu d'affecter uniquement les nouveaux environnements d'exécution. Cela signifie que vous pouvez rencontrer à la fois un démarrage à froid Lambda et ce délai. En définissant une valeur de latence supérieure au délai d'expiration configuré sur la fonction Lambda, cette action donnera également accès à un événement de temporisation haute fidélité.

Type de ressource

- aws:lambda:function

Paramètres

- durée : durée de l'action. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- InvocationPercentage — Facultatif. Le pourcentage (1 à 100) d'invocations de fonctions dans lesquelles injecter le défaut. La valeur par défaut est 100.
- startupDelayMilliseconds : facultatif. Temps d'attente en millisecondes (0 à 900 000) entre l'invocation et l'exécution du code de fonction. La valeur par défaut est 1000.

Autorisations

- `s3:PutObject`
- `s3:DeleteObject`
- `lambda:GetFunction`
- `tag:GetResources`

aws:lambda:invocation-error

Marque les appels de fonctions Lambda comme ayant échoué. Cette action est utile pour tester les mécanismes de gestion des erreurs, tels que les alarmes et les configurations de nouvelle tentative.

Lorsque vous utilisez cette action, vous devez choisir d'exécuter ou non le code de fonction avant de renvoyer une erreur.

Type de ressource

- aws:lambda:function

Paramètres

- durée : durée de l'action. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- InvocationPercentage — Facultatif. Le pourcentage (1 à 100) d'invocations de fonctions dans lesquelles injecter le défaut. La valeur par défaut est 100.
- PreventExecution — Si la valeur est vraie, l'action renvoie l'erreur sans exécuter la fonction.

Autorisations

- `s3:PutObject`
- `s3:DeleteObject`
- `lambda:GetFunction`
- `tag:GetResources`

aws:lambda:invocation-http-integration-response

Modifie le comportement de la fonction. Vous sélectionnez un type de contenu et un code de réponse HTTP pour prendre en charge les intégrations avec ALB, API-GW et VPC Lattice. Pour activer les intégrations en amont ou en aval ayant un impact sélectif, vous pouvez choisir de renvoyer directement la réponse modifiée ou d'exécuter la fonction et de la remplacer une fois l'exécution de la fonction terminée.

Type de ressource

- aws:lambda:function

Paramètres

- contentTypeHeader— Valeur de chaîne de l'en-tête du type de contenu HTTP à renvoyer par la fonction Lambda.
- durée : durée de l'action. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- InvocationPercentage — Facultatif. Le pourcentage (1 à 100) d'invocations de fonctions dans lesquelles injecter le défaut. La valeur par défaut est 100.
- PreventExecution — Si la valeur est vraie, l'action renvoie la réponse sans exécuter la fonction.
- StatusCode — Valeur du code d'état HTTP (000-999) à renvoyer par la fonction Lambda.

Autorisations

- `s3:PutObject`
- `s3:DeleteObject`
- `lambda:GetFunction`
- `tag:GetResources`

Actions du réseau

AWS FIS prend en charge les actions réseau suivantes.

Actions

- aws:network:disrupt-connectivity
- aws:network:route-table-disrupt-cross-region-connectivity
- aws:network:transit-gateway-disrupt-cross-region-connectivity

aws:network:disrupt-connectivity

Refuse le trafic spécifié vers les sous-réseaux cibles en clonant temporairement la liste de contrôle d'accès réseau (ACL réseau) d'origine associée au sous-réseau cible. FIS ajoute des règles de refus à l'ACL du réseau cloné, qui possède une balise ManagedByFIS=true, et l'associe au sous-réseau pendant la durée de l'action. À la fin de l'action, FIS supprime l'ACL réseau cloné et restaure l'association ACL réseau d'origine.

Type de ressource

- aws:ec2:subnet

Paramètres

- scope— Le type de trafic à refuser. Lorsque le champ d'application ne l'est pas`all`, le nombre maximum d'entrées dans le réseau ACLs est de 20. Les valeurs possibles sont :
  - `all`— Refuse tout le trafic entrant et sortant du sous-réseau. Notez que cette option autorise le trafic intra-sous-réseau, y compris le trafic à destination et en provenance des interfaces réseau du sous-réseau.
  - `availability-zone`— Refuse le trafic intra-VPC à destination et en provenance de sous-réseaux dans d'autres zones de disponibilité. Le nombre maximum de sous-réseaux pouvant être ciblés dans un VPC est de 30.
  - `dynamodb`— Refuse le trafic à destination et en provenance du point de terminaison régional pour DynamoDB dans la région actuelle.
  - `prefix-list`— Refuse le trafic à destination et en provenance de la liste de préfixes spécifiée.
  - `s3`— Refuse le trafic à destination et en provenance du point de terminaison régional pour Amazon S3 dans la région actuelle.
  - `vpc`— Refuse le trafic entrant et sortant du VPC.
- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- prefixListIdentifier— Si le champ d'application est le cas`prefix-list`, il s'agit de l'identifiant de la liste de préfixes gérée par le client. Vous pouvez spécifier un nom, un ID ou un ARN. La liste de préfixes peut comporter au maximum 10 entrées.

Autorisations

- `ec2:CreateNetworkAcl`— Crée l'ACL réseau avec le tag ManagedByFis=true.
- `ec2:CreateNetworkAclEntry`— L'ACL réseau doit avoir le tag ManagedByFis=true.
- `ec2:CreateTags`
- `ec2:DeleteNetworkAcl`— L'ACL réseau doit avoir le tag ManagedByFis=true.
- `ec2:DescribeManagedPrefixLists`

- `ec2:DescribeNetworkAcls`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:GetManagedPrefixListEntries`
- `ec2:ReplaceNetworkAclAssociation`

AWS politique gérée

- AWSFaultInjectionSimulatorNetworkAccess

aws:network:route-table-disrupt-cross-region-connectivity

Bloque le trafic provenant des sous-réseaux cibles et destiné à la région spécifiée. Crée des tables de routage qui incluent tous les itinéraires que la région doit isoler. Pour permettre à FIS de créer ces tables de routage, augmentez le quota `routes per route table` Amazon VPC à 250, plus le nombre de routes dans vos tables de routage existantes.

Type de ressource

- aws:ec2:subnet

Paramètres

- `region`— Le code de la région à isoler (par exemple, eu-west-1).
- `duration`— La durée de l'action. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- `ec2:AssociateRouteTable`
- `ec2:CreateManagedPrefixList` †
- `ec2:CreateNetworkInterface` †
- `ec2:CreateRoute` †
- `ec2:CreateRouteTable` †

- ec2:CreateTags †
- ec2:DeleteManagedPrefixList †
- ec2:DeleteNetworkInterface †
- ec2:DeleteRouteTable †
- ec2:DescribeManagedPrefixLists
- ec2:DescribeNetworkInterfaces
- ec2:DescribeRouteTables
- ec2:DescribeSubnets
- ec2:DescribeVpcPeeringConnections
- ec2:DescribeVpcs
- ec2:DisassociateRouteTable
- ec2:GetManagedPrefixListEntries
- ec2:ModifyManagedPrefixList †
- ec2:ModifyVpcEndpoint
- ec2:ReplaceRouteTableAssociation

† Délimité à l'aide de la balisemanagedByFIS=true. Il n'est pas nécessaire de gérer cette balise. AWS FIS ajoute et supprime cette balise au cours de l'expérience.

AWS politique gérée

- AWSFaultInjectionSimulatorNetworkAccess

aws:network:transit-gateway-disrupt-cross-region-connectivity

Bloque le trafic provenant de la passerelle de transit cible en appairant les pièces jointes destinées à la région spécifiée.

Type de ressource

- aws:ec2:transit-gateway

Paramètres

- region— Le code de la région à isoler (par exemple, eu-west-1).

- `duration`— La durée de l'action. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- `ec2:AssociateTransitGatewayRouteTable`
- `ec2:DescribeTransitGatewayAttachments`
- `ec2:DescribeTransitGatewayPeeringAttachments`
- `ec2:DescribeTransitGateways`
- `ec2:DisassociateTransitGatewayRouteTable`

AWS politique gérée

- AWSFaultInjectionSimulatorNetworkAccess

Actions Amazon RDS

AWS FIS prend en charge les actions Amazon RDS suivantes.

Actions

- aws:rds:failover-db-cluster
- aws:rds:reboot-db-instances

aws:rds:failover-db-cluster

Exécute l'action Failover de l'API Amazon RDS DBCluster sur le cluster de base de données Aurora cible.

Type de ressource

- aws:rds:cluster

Paramètres

- Aucun

Autorisations

- `rds:FailoverDBCluster`
- `rds:DescribeDBClusters`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorRDSAccess

aws:rds:reboot-db-instances

Exécute l'action Reboot de l'API Amazon RDS DBInstance sur l'instance de base de données cible.

Type de ressource

- aws:rds:db

Paramètres

- forceFailover : facultatif. Si la valeur est vraie, et si les instances sont multi-AZ, force le basculement d'une zone de disponibilité à l'autre. La valeur par défaut est false.

Autorisations

- `rds:RebootDBInstance`
- `rds:DescribeDBInstances`
- `tag:GetResources`

AWS politique gérée

- AWSFaultInjectionSimulatorRDSAccess

Actions Amazon S3

AWS FIS prend en charge l'action Amazon S3 suivante.

Actions

- aws:s3:bucket-pause-replication

aws:s3:bucket-pause-replication

Suspend la réplication des compartiments source cible vers les compartiments de destination. Les compartiments de destination peuvent se trouver dans différentes régions AWS ou dans la même région que le compartiment source. Les objets existants peuvent continuer à être répliqués jusqu'à une heure après le début de l'action. Cette action prend uniquement en charge le ciblage par balises. Pour en savoir plus sur Amazon S3 Replication, consultez le guide de l'utilisateur d'Amazon S3.

Type de ressource

- aws:s3:bucket

Paramètres

- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.
- region— La région AWS dans laquelle se trouvent les buckets de destination.
- destinationBuckets : facultatif. Liste séparée par des virgules des compartiments S3 de destination.
- prefixes : facultatif. Liste séparée par des virgules des préfixes de clé d'objet S3 provenant des filtres de règles de réplication. Les règles de réplication des compartiments cibles avec un filtre basé sur le ou les préfixes seront suspendues.

Autorisations

- `S3:PutReplicationConfiguration`avec clé de condition `S3:IsReplicationPauseRequest` réglée sur `True`
- `S3:GetReplicationConfiguration`avec clé de condition `S3:IsReplicationPauseRequest` réglée sur `True`
- `S3:PauseReplication`
- `S3:ListAllMyBuckets`
- `tag:GetResources`

Pour un exemple de politique, consultez Exemple : utilisez des clés de condition pour `aws:s3:bucket-pause-replication`.

Actions de Systems Manager

AWS FIS prend en charge les actions Systems Manager suivantes.

Actions

- aws:ssm:send-command
- aws:ssm:start-automation-execution

aws:ssm:send-command

Exécute l'action API Systems Manager SendCommandsur les EC2 instances cibles. Le document Systems Manager (document SSM) définit les actions que Systems Manager effectue sur vos instances. Pour de plus amples informations, veuillez consulter Utilisez l'aws:ssm:send-commandaction.

Type de ressource

- aws:ec2:instance

Paramètres

- documentArn— Le nom de ressource Amazon (ARN) du document. Dans la console, ce paramètre est complété pour vous si vous choisissez une valeur dans Type d'action qui correspond à l'un des documents AWS FIS SSM préconfigurés.
- documentVersion : facultatif. Version du document. S'il est vide, la version par défaut s'exécute.
- documentParameters— Conditionnel. Les paramètres obligatoires et facultatifs acceptés par le document. Le format est un objet JSON dont les clés sont des chaînes et les valeurs sont des chaînes ou des tableaux de chaînes.
- duration— La durée, d'une minute à 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- `ssm:SendCommand`
- `ssm:ListCommands`
- `ssm:CancelCommand`

AWS politique gérée

- AWSFaultInjectionSimulatorEC2Accès

aws:ssm:start-automation-execution

Exécute l'action API Systems Manager StartAutomationExecution.

Type de ressource

- Aucun

Paramètres

- documentArn— Le nom de ressource Amazon (ARN) du document d'automatisation.
- documentVersion : facultatif. Version du document. S'il est vide, la version par défaut s'exécute.
- documentParameters— Conditionnel. Les paramètres obligatoires et facultatifs acceptés par le document. Le format est un objet JSON dont les clés sont des chaînes et les valeurs sont des chaînes ou des tableaux de chaînes.
- maxDuration— La durée maximale autorisée pour l'exécution de l'automatisation, comprise entre une minute et 12 heures. Dans l' AWS FIS API, la valeur est une chaîne au format ISO 8601. Par exemple, PT1 M représente une minute. Dans la AWS FIS console, vous entrez le nombre de secondes, de minutes ou d'heures.

Autorisations

- `ssm:GetAutomationExecution`
- `ssm:StartAutomationExecution`
- `ssm:StopAutomationExecution`
- `iam:PassRole` : facultatif. Obligatoire si le document d'automatisation joue un rôle.

AWS politique gérée

- AWSFaultInjectionSimulatorSSMAccess

Utiliser les documents SSM de Systems Manager avec FIS AWS

AWS FIS prend en charge les types de pannes personnalisés via l'agent AWS Systems Manager SSM et l'action AWS FIS. aws:ssm:send-command Les documents SSM préconfigurés de Systems Manager (documents SSM) qui peuvent être utilisés pour créer des actions d'injection de défauts courantes sont disponibles sous forme de AWS documents publics commençant par le AWSFIS préfixe -.

L'agent SSM est un logiciel Amazon qui peut être installé et configuré sur des EC2 instances Amazon, des serveurs sur site ou des machines virtuelles ()VMs. Cela permet à Systems Manager de gérer ces ressources. L'agent traite les demandes provenant de Systems Manager, puis les exécute comme indiqué dans la demande. Vous pouvez inclure votre propre document SSM pour injecter des erreurs personnalisées, ou faire référence à l'un des documents publics appartenant à Amazon.

Prérequis

Pour les actions qui nécessitent que l'agent SSM exécute l'action sur la cible, vous devez vous assurer que les points suivants sont respectés :

- L'agent est installé sur la cible. L'agent SSM est installé par défaut sur certaines Amazon Machine Images (AMIs). Sinon, vous pouvez installer l'agent SSM sur vos instances. Pour plus d'informations, consultez la section Installation manuelle de l'agent SSM pour les EC2 instances dans le Guide de l'AWS Systems Manager utilisateur.
- Systems Manager est autorisé à effectuer des actions sur vos instances. Vous accordez l'accès à l'aide d'un profil d'instance IAM. Pour plus d'informations, consultez les sections Créer un profil d'instance IAM pour Systems Manager et Attacher un profil d'instance IAM à une EC2 instance dans le Guide de l'AWS Systems Manager utilisateur.

Utilisez l'aws:ssm:send-commandaction

Un document SSM définit les actions exécutées par Systems Manager sur vos instances gérées. Systems Manager inclut un certain nombre de documents préconfigurés, mais vous pouvez également créer les vôtres. Pour plus d'informations sur la création de votre propre document SSM,

consultez la section Creating Systems Manager dans le guide de l'AWS Systems Manager utilisateur. Pour plus d'informations sur les documents SSM en général, consultez les AWS Systems Manager documents du Guide de l'AWS Systems Manager utilisateur.

AWS FIS fournit des documents SSM préconfigurés. Vous pouvez consulter les documents SSM préconfigurés sous Documents dans la AWS Systems Manager console : https://console.aws.amazon.com/systems-manager/ documents. Vous pouvez également choisir parmi une sélection de documents préconfigurés dans la console AWS FIS. Pour de plus amples informations, veuillez consulter Documents AWS FIS SSM préconfigurés.

Pour utiliser un document SSM dans vos expériences AWS FIS, vous pouvez utiliser l'aws:ssm:send-commandaction. Cette action récupère et exécute le document SSM spécifié sur vos instances cibles.

Lorsque vous utilisez l'`aws:ssm:send-command`action dans votre modèle de test, vous devez spécifier des paramètres supplémentaires pour l'action, notamment les suivants :

- documentArn : obligatoire. Le nom de ressource Amazon (ARN) du document SSM.
- documentParameters— Conditionnel. Les paramètres obligatoires et facultatifs acceptés par le document SSM. Le format est un objet JSON dont les clés sont des chaînes et les valeurs sont des chaînes ou des tableaux de chaînes.
- documentVersion : facultatif. Version du document SSM à exécuter.

Vous pouvez consulter les informations d'un document SSM (y compris les paramètres du document) à l'aide de la console Systems Manager ou de la ligne de commande.

Pour afficher les informations relatives à un document SSM à l'aide de la console

1. Ouvrez la AWS Systems Manager console à l'adresse https://console.aws.amazon.com/systems-manager/.
2. Dans le panneau de navigation, cliquez sur Documents.
3. Sélectionnez le document, puis cliquez sur l'onglet Détails.

Pour afficher les informations relatives à un document SSM à l'aide de la ligne de commande

Utilisez la commande SSM describe-document.

Documents AWS FIS SSM préconfigurés

Vous pouvez utiliser des documents AWS FIS SSM préconfigurés avec l'`aws:ssm:send-command`action dans vos modèles d'expérience.

Prérequis

- Les documents SSM préconfigurés fournis par AWS FIS ne sont pris en charge que sur les systèmes d'exploitation suivants :
  - Amazon Linux 2023, Amazon Linux 2
  - Ubuntu
  - RHEL 8, 9
  - CentOS 9
- Les documents SSM préconfigurés fournis par AWS FIS ne sont pris en charge que sur les instances. EC2 Ils ne sont pas pris en charge sur les autres types de nœuds gérés, tels que les serveurs locaux.

Pour utiliser ces documents SSM dans des expériences sur des tâches ECS, utilisez le document correspondantthe section called “Actions d'Amazon ECS”. Par exemple, l'aws:ecs:task-cpu-stressaction utilise le AWSFIS-Run-CPU-Stress document.

Documents

- AWSFIS-Run-CPU-Stress
- AWSFIS-Run-Disk-Fill
- AWSFIS-Run-IO-Stress
- AWSFIS-Run-Kill-Process
- AWSFIS-Run-Memory-Stress
- AWSFIS-Run-Network-Blackhole-Port
- AWSFIS-Run-Network-Latency
- AWSFIS-Run-Network-Latency-Sources
- AWSFIS-Run-Network-Packet-Loss
- AWSFIS-Run-Network-Packet-Loss-Sources

Différence entre la durée de l'action et DurationSeconds dans les AWS documents FIS SSM

Certains documents SSM limitent leur propre temps d'exécution, par exemple le DurationSeconds paramètre est utilisé par certains documents AWS FIS SSM préconfigurés. Par conséquent, vous devez spécifier deux durées indépendantes dans la définition de l'action AWS FIS :

- Action duration: Pour les expériences comportant une seule action, la durée de l'action est équivalente à la durée de l'expérience. Dans le cas de plusieurs actions, la durée de l'expérience dépend de la durée des actions individuelles et de l'ordre dans lequel elles sont exécutées. AWS Le FIS surveille chaque action jusqu'à ce que sa durée soit écoulée.
- Paramètre du document DurationSeconds : durée, spécifiée en secondes, pendant laquelle le document SSM sera exécuté.

Vous pouvez choisir différentes valeurs pour les deux types de durée :

- Action duration exceeds DurationSeconds: L'exécution du document SSM se termine avant que l'action ne soit terminée. AWS Le FIS attend que la durée de l'action soit écoulée avant que les actions suivantes ne soient lancées.
- Action duration is shorter than DurationSeconds: Le document SSM continue l'exécution une fois l'action terminée. Si l'exécution du document SSM est toujours en cours et que la durée de l'action est expirée, le statut de l'action est défini sur Terminé. AWS Le FIS surveille uniquement l'exécution jusqu'à ce que la durée de l'action soit écoulée.

Notez que certains documents SSM ont des durées variables. Par exemple, les documents AWS FIS SSM ont la possibilité d'installer des prérequis, ce qui peut prolonger la durée d'exécution globale au-delà du paramètre spécifié. DurationSeconds Ainsi, si vous définissez la durée de l'action sur la même valeur, il est possible que le script SSM s'exécute plus longtemps que la durée de l'action. DurationSeconds

AWSFIS-Run-CPU-Stress

Exécute le stress du processeur sur une instance à l'aide de l'stress-ngoutil. Utilise le document AWSFIS-RunSSM -CPU-Stress.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-CPU-Stress

ARN

arn:aws:ssm:region::document/AWSFIS-Run-CPU-Stress

Paramètres du document

- DurationSeconds : obligatoire. Durée du test de stress du processeur, en secondes.
- CPU : facultatif. Le nombre de facteurs de stress du processeur à utiliser. La valeur par défaut est 0, qui utilise tous les facteurs de stress du processeur.
- LoadPercent : facultatif. Pourcentage de charge du processeur cible, compris entre 0 (aucune charge) et 100 (pleine charge). La valeur par défaut est 100.
- InstallDependencies : facultatif. Si la valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. La dépendance eststress-ng.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Disk-Fill

Alloue de l'espace disque sur le volume racine d'une instance afin de simuler une panne complète du disque. Utilise le document AWSFIS-RunSSM -Disk-Fill.

Si l'expérience à l'origine de cette erreur est arrêtée, soit manuellement, soit par le biais d'une condition d'arrêt, AWS FIS tente de revenir en arrière en annulant le document SSM en cours d'exécution. Toutefois, si le disque est plein à 100 %, soit en raison d'une panne, soit en raison d'une panne liée à l'activité de l'application, Systems Manager risque de ne pas être en mesure de terminer l'opération d'annulation. Par conséquent, si vous devez arrêter l'expérience, assurez-vous que le disque ne sera pas plein à 100 %.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Disk-Fill

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Disk-Fill

Paramètres du document

- DurationSeconds : obligatoire. Durée du test de remplissage du disque, en secondes.

- Percent : facultatif. Pourcentage du disque à allouer lors du test de remplissage du disque. La valeur par défaut est 95 %.
- InstallDependencies : facultatif. Si la valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. Les dépendances sontatd, kmod etfallocate.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-IO-Stress

Exécute le stress d'E/S sur une instance à l'aide de l'stress-ngoutil. Utilise le document AWSFIS-Run-IO-Stress SSM.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-IO-Stress

ARN

arn:aws:ssm:region::document/AWSFIS-Run-IO-Stress

Paramètres du document

- DurationSeconds : obligatoire. Durée du test de stress IO, en secondes.
- Workers : facultatif. Nombre de travailleurs qui effectuent une combinaison d' read/write opérations séquentielles, aléatoires et mappées en mémoire, de synchronisation forcée et de suppression du cache. Plusieurs processus enfants exécutent différentes I/O opérations sur le même fichier. La valeur par défaut est 1.
- Percent : facultatif. Pourcentage d'espace libre sur le système de fichiers à utiliser pendant le test de stress IO. La valeur par défaut est de 80 %.
- InstallDependencies : facultatif. Si la valeur est`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. La dépendance eststress-ng.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"Workers":"1", "Percent":"80", "DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Kill-Process

Arrête le processus spécifié dans l'instance à l'aide de la killall commande. Utilise le AWSFIS-Rundocument SSM -Kill-Process.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Kill-Process

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Kill-Process

Paramètres du document

- ProcessName : obligatoire. Nom du processus à arrêter.
- Signal : facultatif. Le signal à envoyer avec la commande. Les valeurs possibles sont `SIGTERM` (que le récepteur peut choisir d'ignorer) et `SIGKILL` (qui ne peuvent pas être ignorées). La valeur par défaut est `SIGTERM`.
- InstallDependencies – Facultatif. Si la valeur est`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. La dépendance estkillall.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"ProcessName":"myapplication", "Signal":"SIGTERM"}
```

AWSFIS-Run-Memory-Stress

Exécute un stress mémoire sur une instance à l'aide de l'stress-ngoutil. Utilise le document AWSFIS-Run-Memory-Stress SSM.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Memory-Stress

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Memory-Stress

Paramètres du document

- DurationSeconds : obligatoire. Durée du test de stress mnésique, en secondes.
- Workers : facultatif. Nombre de facteurs de stress liés à la mémoire virtuelle. La valeur par défaut est 1.
- Percent : obligatoire. Pourcentage de mémoire virtuelle à utiliser pendant le test de stress lié à la mémoire.
- InstallDependencies : facultatif. Si la valeur est`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. La dépendance eststress-ng.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"Percent":"80", "DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Blackhole-Port

Supprime le trafic entrant ou sortant pour le protocole et le port à l'aide de l'iptablesoutil. Utilise le document AWSFIS-RunSSM -Network-Blackhole-Port.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Network-Blackhole-Port

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Blackhole-Port

Paramètres du document

- Protocol : obligatoire. Protocole. Les valeurs possibles sont `tcp` et udp.
- Port : obligatoire. Numéro de port.
- TrafficType : facultatif. Type de trafic. Les valeurs possibles sont `ingress` et `egress`. La valeur par défaut est `ingress`.
- DurationSeconds : obligatoire. Durée du test du trou noir du réseau, en secondes.
- InstallDependencies : facultatif. Si la valeur est`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. Les dépendances sont atddig,lsof, etiptables.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"Protocol":"tcp", "Port":"8080", "TrafficType":"egress", "DurationSeconds":"60",
 "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Latency

Ajoute de la latence à l'interface réseau à l'aide de l'tcoutil. Utilise le document AWSFIS-RunSSM - Network-Latency.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Network-Latency

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Latency

Paramètres du document

- Interface : facultatif. L'interface réseau. La valeur par défaut est `eth0`.
- DelayMilliseconds – Facultatif. Le délai, en millisecondes. La valeur par défaut est 200.
- DurationSeconds : obligatoire. Durée du test de latence du réseau, en secondes.
- InstallDependencies : facultatif. Si la valeur est`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. Les dépendances sont atddig, ettc.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"DelayMilliseconds":"200", "Interface":"eth0", "DurationSeconds":"60",
 "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Latency-Sources

Ajoute de la latence et de l'instabilité à l'interface réseau à l'aide de l'tcoutil pour le trafic à destination ou en provenance de sources spécifiques. Utilise le document AWSFIS-RunSSM -Network-Latency-Sources.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Network-Latency-Sources

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Latency-Sources

Paramètres du document

- Interface : facultatif. L'interface réseau. La valeur par défaut est `eth0`.
- DelayMilliseconds – Facultatif. Le délai, en millisecondes. La valeur par défaut est 200.
- JitterMilliseconds : facultatif. L'instabilité, en millisecondes. La valeur par défaut est 10.
- Sources : obligatoire. Les sources, séparées par des virgules, sans espaces. Les valeurs possibles sont les suivantes : une IPv4 adresse, un bloc IPv4 CIDR, un nom de domaine et`S3`. `DYNAMODB` Si vous spécifiez `DYNAMODB` ou`S3`, cela ne s'applique qu'au point de terminaison régional de la région actuelle.
- TrafficType : facultatif. Type de trafic. Les valeurs possibles sont `ingress` et `egress`. La valeur par défaut est `ingress`.
- DurationSeconds : obligatoire. Durée du test de latence du réseau, en secondes.
- InstallDependencies : facultatif. Si la valeur est`True`, Systems Manager installe les dépendances requises sur les instances cibles si elles ne sont pas déjà installées. L'argument par défaut est `True`. Les dépendances sont atddig,jq,lsof, ettc.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"DelayMilliseconds":"200", "JitterMilliseconds":"15",
 "Sources":"S3,www.example.com,72.21.198.67", "Interface":"eth0",
 "TrafficType":"egress", "DurationSeconds":"60", "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Packet-Loss

Ajoute la perte de paquets à l'interface réseau à l'aide de l'tcoutil. Utilise le document AWSFIS-RunSSM -Network-Packet-Loss.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Network-Packet-Loss

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Packet-Loss

Paramètres du document

- Interface : facultatif. L'interface réseau. La valeur par défaut est `eth0`.
- LossPercent – Facultatif. Pourcentage de perte de paquets. La valeur par défaut est de 7 %.
- DurationSeconds : obligatoire. Durée du test de perte de paquets réseau, en secondes.
- InstallDependencies : facultatif. Si la valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur les instances cibles. L'argument par défaut est `True`. Les dépendances sont atdlsof,dig, ettc.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"LossPercent":"15", "Interface":"eth0", "DurationSeconds":"60",
 "InstallDependencies":"True"}
```

AWSFIS-Run-Network-Packet-Loss-Sources

Ajoute la perte de paquets à l'interface réseau à l'aide de l'tcoutil pour le trafic à destination ou en provenance de sources spécifiques. Utilise le document SSM AWSFIS-Run-Network-Packet-Loss-Sources.

Type d'action (console uniquement)

aws:ssm:send-command/AWSFIS-Run-Network-Packet-Loss-Sources

ARN

arn:aws:ssm:region::document/AWSFIS-Run-Network-Packet-Loss-Sources

Paramètres du document

- Interface : facultatif. L'interface réseau. La valeur par défaut est `eth0`.
- LossPercent – Facultatif. Pourcentage de perte de paquets. La valeur par défaut est de 7 %.
- Sources : obligatoire. Les sources, séparées par des virgules, sans espaces. Les valeurs possibles sont les suivantes : une IPv4 adresse, un bloc IPv4 CIDR, un nom de domaine et`S3`. `DYNAMODB` Si vous spécifiez `DYNAMODB` ou`S3`, cela ne s'applique qu'au point de terminaison régional de la région actuelle.
- TrafficType : facultatif. Type de trafic. Les valeurs possibles sont `ingress` et `egress`. La valeur par défaut est `ingress`.

- DurationSeconds : obligatoire. Durée du test de perte de paquets réseau, en secondes.
- InstallDependencies : facultatif. Si la valeur est définie sur cette valeur`True`, Systems Manager installe les dépendances requises sur les instances cibles. L'argument par défaut est `True`. Les dépendances sont atddig,jq,lsof, ettc.

Voici un exemple de chaîne que vous pouvez saisir dans la console.

```
{"LossPercent":"15", "Sources":"S3,www.example.com,72.21.198.67", "Interface":"eth0",
 "TrafficType":"egress", "DurationSeconds":"60", "InstallDependencies":"True"}
```

Exemples

Pour un exemple de modèle d'expérience, voirthe section called "Exécuter un document AWS FIS SSM préconfiguré".

Pour voir un exemple de didacticiel, consultez la section Exécuter le stress du processeur sur une instance.

Résolution des problèmes

Suivez la procédure ci-dessous pour résoudre les problèmes.

Pour résoudre les problèmes liés aux documents SSM

1. Ouvrez la AWS Systems Manager console à l'adresse https://console.aws.amazon.com/systems-manager/.
2. Dans le volet de navigation, choisissez Node Management, Run Command.
3. Dans l'onglet Historique des commandes, utilisez les filtres pour localiser l'exécution du document.
4. Choisissez l'ID de la commande pour ouvrir sa page de détails.
5. Choisissez l'ID de l'instance. Passez en revue le résultat et les erreurs pour chaque étape.

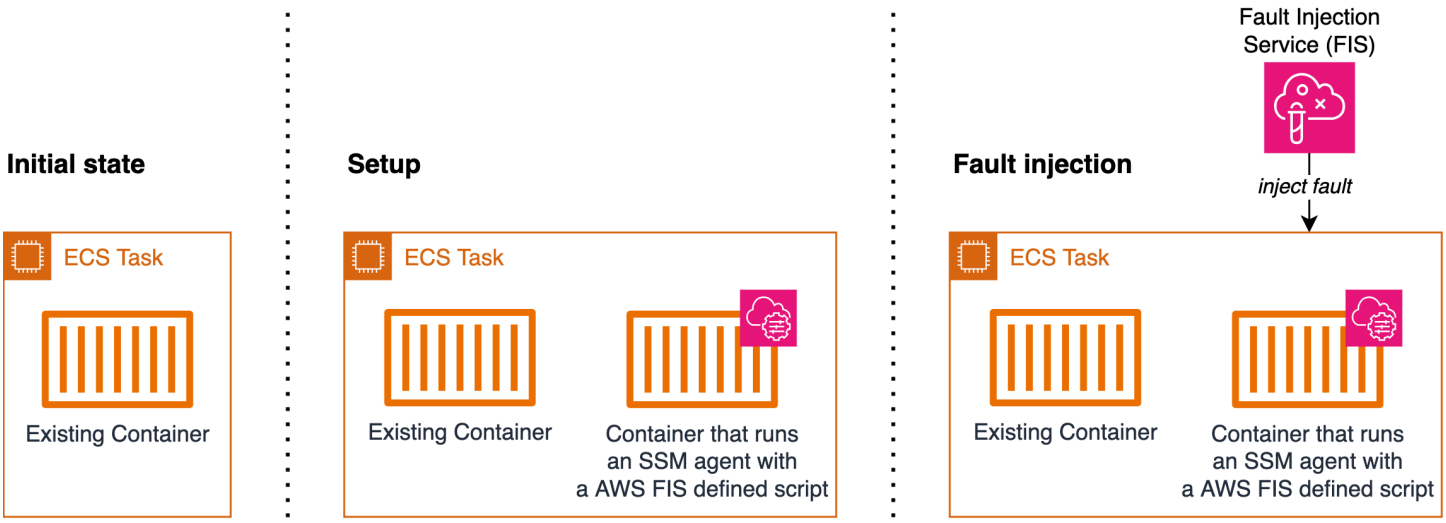
Utilisez les actions AWS FIS aws:ecs:task

Vous pouvez utiliser les actions aws:ecs:task pour injecter des erreurs dans vos tâches Amazon ECS. Les types de capacité Amazon EC2 et Fargate sont pris en charge.

Ces actions utilisent les documents AWS Systems Manager (SSM) pour injecter des erreurs. Pour utiliser `aws:ecs:task` les actions, vous devez ajouter un conteneur avec un agent SSM à votre définition de tâche Amazon Elastic Container Service (Amazon ECS). Le conteneur exécute un script défini par AWS FIS qui enregistre la tâche Amazon ECS en tant qu'instance gérée dans le service SSM. En outre, le script récupère les métadonnées des tâches pour ajouter des balises à l'instance gérée. La configuration permettra à AWS FIS de résoudre la tâche cible. Ce paragraphe fait référence à la configuration dans le schéma ci-dessous.

Lorsque vous exécutez un ciblage d'expériences AWS FIS`aws:ecs:task`, AWS FIS mappe les tâches Amazon ECS cibles que vous spécifiez dans un modèle d'expérience AWS FIS à un ensemble d'instances gérées par SSM à l'aide d'une balise de ressource,. `ECS_TASK_ARN` La valeur de la balise est l'ARN de la tâche Amazon ECS associée dans laquelle les documents SSM doivent être exécutés. Ce paragraphe fait référence à l'injection de défauts dans le schéma ci-dessous.

Le schéma suivant illustre la configuration et l'injection de défauts sur une tâche avec un conteneur existant.



Actions

- the section called “aws:ecs:task-cpu-stress”
- the section called “aws:ecs:task-io-stress”
- the section called “aws:ecs:task-kill-process”
- the section called “aws:ecs:task-network-blackhole-port”
- the section called “aws:ecs:task-network-latency”
- the section called “aws:ecs:task-network-packet-loss”

Limites

- Les actions suivantes ne peuvent pas être exécutées en parallèle :
  - aws:ecs:task-network-blackhole-port
  - aws:ecs:task-network-latency
  - aws:ecs:task-network-packet-loss
- Si vous avez activé Amazon ECS Exec, vous devez le désactiver avant de pouvoir utiliser ces actions.
- L'exécution du document SSM peut avoir le statut Annulé même si l'expérience a l'état Terminé. Lors de l'exécution d'actions Amazon ECS, la durée fournie par le client est utilisée à la fois pour la durée de l'action dans l'expérience et pour la durée du document Amazon EC2 Systems Manager (SSM). Une fois l'action initiée, le document SSM met un certain temps à s'exécuter. Par conséquent, lorsque la durée d'action spécifiée est atteinte, il se peut qu'il reste encore quelques secondes au document SSM pour terminer son exécution. Lorsque la durée de l'action de l'expérience est atteinte, l'action est arrêtée et l'exécution du document SSM est annulée. L'injection du défaut a réussi.

Prérequis

- Ajoutez les autorisations suivantes au rôle d'expérience AWS FIS :
  - `ssm:SendCommand`
  - `ssm:ListCommands`
  - `ssm:CancelCommand`
- Ajoutez les autorisations suivantes au rôle IAM de la tâche Amazon ECS :
  - `ssm:CreateActivation`
  - `ssm:AddTagsToResource`
  - `iam:PassRole`

  Notez que vous pouvez spécifier l'ARN du rôle d'instance géré comme ressource pour `iam:PassRole`.
- Créez un rôle IAM d'exécution de tâches Amazon ECS et ajoutez la politique ECSTask ExecutionRolePolicy gérée par Amazon.

- Dans la définition de la tâche, définissez la variable `MANAGED_INSTANCE_ROLE_NAME` d'environnement sur le nom du rôle d'instance géré. Il s'agit du rôle qui sera attaché aux tâches enregistrées en tant qu'instances gérées dans SSM.
- Ajoutez les autorisations suivantes au rôle d'instance géré :
  - `ssm:DeleteActivation`
  - `ssm:DeregisterManagedInstance`
- Ajoutez la politique SSMManaged InstanceCore gérée par Amazon au rôle d'instance gérée.
- Ajoutez un conteneur d'agent SSM à la définition de tâche Amazon ECS. Le script de commande enregistre les tâches Amazon ECS en tant qu'instances gérées.

```
{
    "name": "amazon-ssm-agent",
    "image": "public.ecr.aws/amazon-ssm-agent/amazon-ssm-agent:latest",
    "cpu": 0,
    "links": [],
    "portMappings": [],
    "essential": false,
    "entryPoint": [],
    "command": [
        "/bin/bash",
        "-c",
        "set -e; dnf upgrade -y; dnf install jq procps awscli -y; term_handler()
 { echo \"Deleting SSM activation $ACTIVATION_ID\"; if ! aws ssm delete-
activation --activation-id $ACTIVATION_ID --region $ECS_TASK_REGION; then
 echo \"SSM activation $ACTIVATION_ID failed to be deleted\" 1>&2; fi;
 MANAGED_INSTANCE_ID=$(jq -e -r .ManagedInstanceID /var/lib/amazon/ssm/registration);
 echo \"Deregistering SSM Managed Instance $MANAGED_INSTANCE_ID\"; if ! aws
 ssm deregister-managed-instance --instance-id $MANAGED_INSTANCE_ID --region
 $ECS_TASK_REGION; then echo \"SSM Managed Instance $MANAGED_INSTANCE_ID
 failed to be deregistered\" 1>&2; fi; kill -SIGTERM $SSM_AGENT_PID; }; trap
 term_handler SIGTERM SIGINT; if [[ -z $MANAGED_INSTANCE_ROLE_NAME ]]; then
 echo \"Environment variable MANAGED_INSTANCE_ROLE_NAME not set, exiting\"
 1>&2; exit 1; fi; if ! ps ax | grep amazon-ssm-agent | grep -v grep > /dev/
null; then if [[ -n $ECS_CONTAINER_METADATA_URI_V4 ]] ; then echo \"Found ECS
 Container Metadata, running activation with metadata\"; TASK_METADATA=$(curl
 \"${ECS_CONTAINER_METADATA_URI_V4}/task\"); ECS_TASK_AVAILABILITY_ZONE=$(echo
 $TASK_METADATA | jq -e -r '.AvailabilityZone'); ECS_TASK_ARN=$(echo $TASK_METADATA
 | jq -e -r '.TaskARN'); ECS_TASK_REGION=$(echo $ECS_TASK_AVAILABILITY_ZONE | sed
 's/.$//'); ECS_TASK_AVAILABILITY_ZONE_REGEX='^(af|ap|ca|cn|eu|me|sa|us|us-gov)-
(central|north|(north(east|west))|south|south(east|west)|east|west)-[0-9]{1}[a-z]
{1}$'; if ! [[ $ECS_TASK_AVAILABILITY_ZONE =~ $ECS_TASK_AVAILABILITY_ZONE_REGEX ]];
```

```
 then echo \"Error extracting Availability Zone from ECS Container Metadata,
 exiting\" 1>&2; exit 1; fi; ECS_TASK_ARN_REGEX='^arn:(aws|aws-cn|aws-us-gov):ecs:
[a-z0-9-]+:[0-9]{12}:task/[a-zA-Z0-9_-]+/[a-zA-Z0-9]+$'; if ! [[ $ECS_TASK_ARN
 =~ $ECS_TASK_ARN_REGEX ]]; then echo \"Error extracting Task ARN from ECS
 Container Metadata, exiting\" 1>&2; exit 1; fi; CREATE_ACTIVATION_OUTPUT=
$(aws ssm create-activation --iam-role $MANAGED_INSTANCE_ROLE_NAME --
tags Key=ECS_TASK_AVAILABILITY_ZONE,Value=$ECS_TASK_AVAILABILITY_ZONE
 Key=ECS_TASK_ARN,Value=$ECS_TASK_ARN Key=FAULT_INJECTION_SIDECAR,Value=true --
region $ECS_TASK_REGION); ACTIVATION_CODE=$(echo $CREATE_ACTIVATION_OUTPUT | jq
 -e -r .ActivationCode); ACTIVATION_ID=$(echo $CREATE_ACTIVATION_OUTPUT | jq -e
 -r .ActivationId); if ! amazon-ssm-agent -register -code $ACTIVATION_CODE -id
 $ACTIVATION_ID -region $ECS_TASK_REGION; then echo \"Failed to register with AWS
 Systems Manager (SSM), exiting\" 1>&2; exit 1; fi; amazon-ssm-agent & SSM_AGENT_PID=
$!; wait $SSM_AGENT_PID; else echo \"ECS Container Metadata not found, exiting\"
 1>&2; exit 1; fi; else echo \"SSM agent is already running, exiting\" 1>&2; exit 1;
 fi"
    ],
    "environment": [
        {
            "name": "MANAGED_INSTANCE_ROLE_NAME",
            "value": "SSMManagedInstanceRole"
        }
    ],
    "environmentFiles": [],
    "mountPoints": [],
    "volumesFrom": [],
    "secrets": [],
    "dnsServers": [],
    "dnsSearchDomains": [],
    "extraHosts": [],
    "dockerSecurityOptions": [],
    "dockerLabels": {},
    "ulimits": [],
    "logConfiguration": {},
    "systemControls": []
}
```

Pour une version plus lisible du script, voirthe section called “Version de référence du script”.

- Activez l'injection APIs de défauts Amazon ECS en définissant le `enableFaultInjection` champ dans la définition de tâche Amazon ECS :

```
"enableFaultInjection": true,
```

- Lorsque vous utilisez les `aws:ecs:task-network-packet-loss` actions `aws:ecs:task-network-blackhole-portaws:ecs:task-network-latency`, ou sur les tâches Fargate, le paramètre de l'action doit être défini `useEcsFaultInjectionEndpoints` sur. `true`
- Lorsque vous utilisez les `aws:ecs:task-network-packet-loss` actions `aws:ecs:task-kill-processaws:ecs:task-network-blackhole-port`,`aws:ecs:task-network-latency`, et,,,,, la définition de tâche Amazon ECS doit être `pidMode` définie sur`task`.
- Lorsque vous utilisez les `aws:ecs:task-network-packet-loss` actions `aws:ecs:task-network-blackhole-portaws:ecs:task-network-latency`,, et, la définition de tâche Amazon ECS doit avoir été `networkMode` définie sur une valeur autre que`bridge`.

Version de référence du script

Vous trouverez ci-dessous une version plus lisible du script dans la section Exigences, à titre de référence.

```
#!/usr/bin/env bash

# This is the activation script used to register ECS tasks as Managed Instances in SSM
# The script retrieves information form the ECS task metadata endpoint to add three
 tags to the Managed Instance
#  - ECS_TASK_AVAILABILITY_ZONE: To allow customers to target Managed Instances / Tasks
 in a specific Availability Zone
#  - ECS_TASK_ARN: To allow customers to target Managed Instances / Tasks by using the
 Task ARN
#  - FAULT_INJECTION_SIDECAR: To make it clear that the tasks were registered as
 managed instance for fault injection purposes. Value is always 'true'.
# The script will leave the SSM Agent running in the background
# When the container running this script receives a SIGTERM or SIGINT signal, it will
 do the following cleanup:
#  - Delete SSM activation
#  - Deregister SSM managed instance

set -e # stop execution instantly as a query exits while having a non-zero

dnf upgrade -y
dnf install jq procps awscli -y

term_handler() {
  echo "Deleting SSM activation $ACTIVATION_ID"
```

```
  if ! aws ssm delete-activation --activation-id $ACTIVATION_ID --region
 $ECS_TASK_REGION; then
    echo "SSM activation $ACTIVATION_ID failed to be deleted" 1>&2
  fi

  MANAGED_INSTANCE_ID=$(jq -e -r .ManagedInstanceID /var/lib/amazon/ssm/registration)
  echo "Deregistering SSM Managed Instance $MANAGED_INSTANCE_ID"
  if ! aws ssm deregister-managed-instance --instance-id $MANAGED_INSTANCE_ID --region
 $ECS_TASK_REGION; then
    echo "SSM Managed Instance $MANAGED_INSTANCE_ID failed to be deregistered" 1>&2
  fi

  kill -SIGTERM $SSM_AGENT_PID
}
trap term_handler SIGTERM SIGINT

# check if the required IAM role is provided
if [[ -z $MANAGED_INSTANCE_ROLE_NAME ]] ; then
  echo "Environment variable MANAGED_INSTANCE_ROLE_NAME not set, exiting" 1>&2
  exit 1
fi

# check if the agent is already running (it will be if ECS Exec is enabled)
if ! ps ax | grep amazon-ssm-agent | grep -v grep > /dev/null; then

  # check if ECS Container Metadata is available
  if [[ -n $ECS_CONTAINER_METADATA_URI_V4 ]] ; then

    # Retrieve info from ECS task metadata endpoint
    echo "Found ECS Container Metadata, running activation with metadata"
    TASK_METADATA=$(curl "${ECS_CONTAINER_METADATA_URI_V4}/task")
    ECS_TASK_AVAILABILITY_ZONE=$(echo $TASK_METADATA | jq -e -r '.AvailabilityZone')
    ECS_TASK_ARN=$(echo $TASK_METADATA | jq -e -r '.TaskARN')
    ECS_TASK_REGION=$(echo $ECS_TASK_AVAILABILITY_ZONE | sed 's/.$//')

    # validate ECS_TASK_AVAILABILITY_ZONE
    ECS_TASK_AVAILABILITY_ZONE_REGEX='^(af|ap|ca|cn|eu|me|sa|us|us-gov)-(central|north|
(north(east|west))|south|south(east|west)|east|west)-[0-9]{1}[a-z]{1}$'
    if ! [[ $ECS_TASK_AVAILABILITY_ZONE =~ $ECS_TASK_AVAILABILITY_ZONE_REGEX ]] ; then
      echo "Error extracting Availability Zone from ECS Container Metadata, exiting"
 1>&2
      exit 1
    fi
```

```
    # validate ECS_TASK_ARN
    ECS_TASK_ARN_REGEX='^arn:(aws|aws-cn|aws-us-gov):ecs:[a-z0-9-]+:[0-9]{12}:task/[a-
zA-Z0-9_-]+/[a-zA-Z0-9]+$'
    if ! [[ $ECS_TASK_ARN =~ $ECS_TASK_ARN_REGEX ]] ; then
      echo "Error extracting Task ARN from ECS Container Metadata, exiting" 1>&2
      exit 1
    fi

    # Create activation tagging with Availability Zone and Task ARN
    CREATE_ACTIVATION_OUTPUT=$(aws ssm create-activation \
      --iam-role $MANAGED_INSTANCE_ROLE_NAME \
      --tags Key=ECS_TASK_AVAILABILITY_ZONE,Value=$ECS_TASK_AVAILABILITY_ZONE
 Key=ECS_TASK_ARN,Value=$ECS_TASK_ARN Key=FAULT_INJECTION_SIDECAR,Value=true \
      --region $ECS_TASK_REGION)

    ACTIVATION_CODE=$(echo $CREATE_ACTIVATION_OUTPUT | jq -e -r .ActivationCode)
    ACTIVATION_ID=$(echo $CREATE_ACTIVATION_OUTPUT | jq -e -r .ActivationId)

    # Register with AWS Systems Manager (SSM)
    if ! amazon-ssm-agent -register -code $ACTIVATION_CODE -id $ACTIVATION_ID -region
 $ECS_TASK_REGION; then
      echo "Failed to register with AWS Systems Manager (SSM), exiting" 1>&2
      exit 1
    fi

    # the agent needs to run in the background, otherwise the trapped signal
    # won't execute the attached function until this process finishes
    amazon-ssm-agent &
    SSM_AGENT_PID=$!

    # need to keep the script alive, otherwise the container will terminate
    wait $SSM_AGENT_PID

  else
    echo "ECS Container Metadata not found, exiting" 1>&2
    exit 1
  fi

else
  echo "SSM agent is already running, exiting" 1>&2
  exit 1
fi
```

Exemple de modèle d'expérience

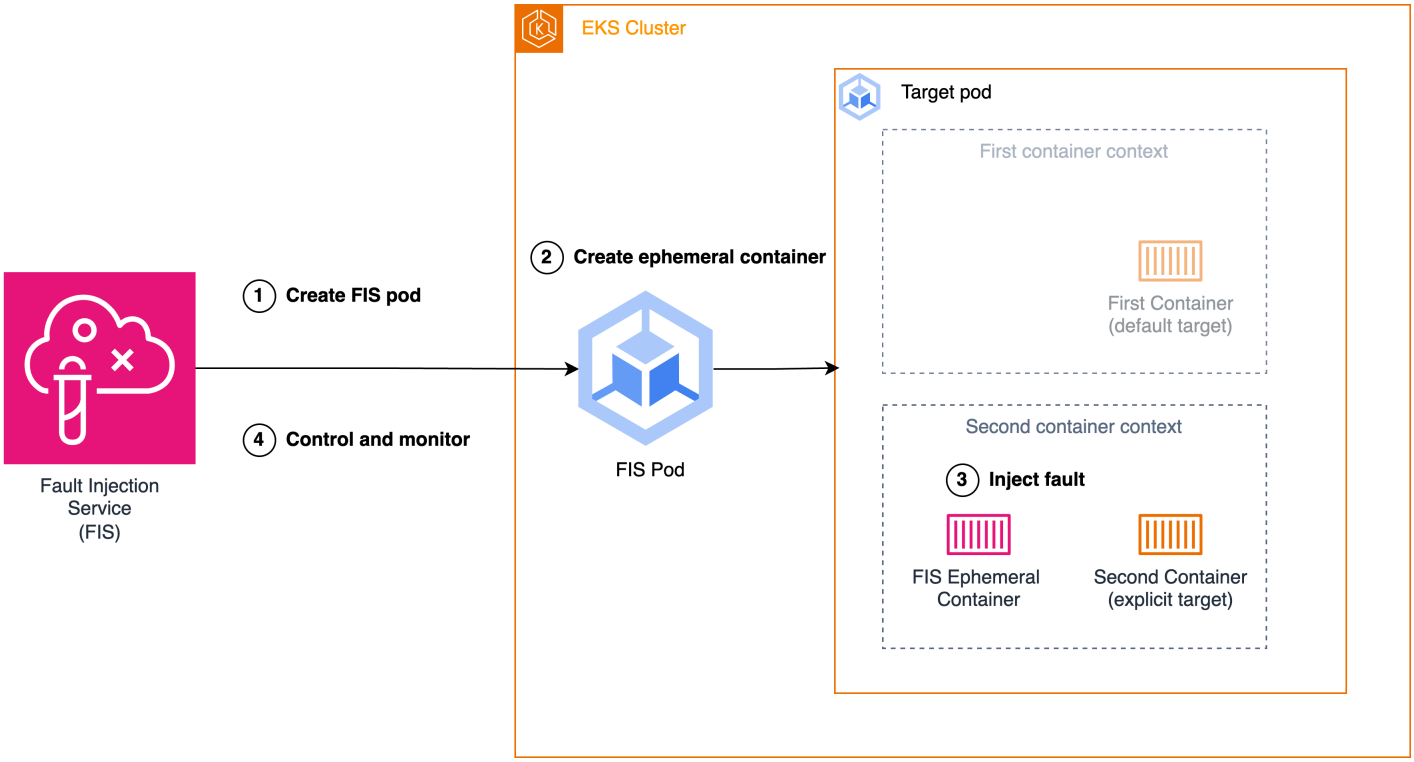
Voici un exemple de modèle d'expérience pour l'the section called "aws:ecs:task-cpu-stress"action.

```
{
    "description": "Run CPU stress on the target ECS tasks",
    "targets": {
        "myTasks": {
            "resourceType": "aws:ecs:task",
            "resourceArns": [
                "arn:aws:ecs:us-east-1:111122223333:task/my-
cluster/09821742c0e24250b187dfed8EXAMPLE"
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "EcsTask-cpu-stress": {
            "actionId": "aws:ecs:task-cpu-stress",
            "parameters": {
                "duration": "PT1M"
            },
            "targets": {
                "Tasks": "myTasks"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none",
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/fis-experiment-role",
    "tags": {}
}
```

Utilisez les actions AWS FIS aws:eks:pod

Vous pouvez utiliser les actions aws:eks:pod pour injecter des erreurs dans les pods Kubernetes exécutés dans vos clusters EKS.

Lorsqu'une action est lancée, le FIS récupère l'image du conteneur FIS Pod. Cette image est ensuite utilisée pour créer un Pod dans le cluster EKS ciblé. Le Pod nouvellement créé est chargé d'injecter, de contrôler et de surveiller le défaut. Pour toutes les actions FIS EKS, à l'exception de aws:eks:pod-delete, l'injection d'erreurs est réalisée grâce à l'utilisation de conteneurs éphémères, une fonctionnalité de Kubernetes qui permet de créer des conteneurs temporaires dans un pod existant. Le conteneur éphémère est démarré dans le même espace de noms que le conteneur cible et exécute les tâches d'injection de défauts souhaitées. Si aucun conteneur cible n'est spécifié, le premier conteneur de la spécification Pod est sélectionné comme cible.



1. FIS crée le module FIS dans le cluster cible spécifié dans le modèle d'expérience.
2. Le module FIS crée un conteneur éphémère dans le pod cible dans le même espace de noms que le conteneur cible.
3. Le conteneur éphémère injecte des défauts dans l'espace de noms du conteneur cible.
4. Le FIS Pod contrôle et surveille l'injection défectueuse du contenant éphémère et le FIS contrôle et surveille le FIS Pod.

À la fin de l'expérience ou en cas d'erreur, le contenant éphémère et le module FIS sont retirés.

Actions

- the section called "aws:eks:pod-cpu-stress"
- the section called "aws:eks:pod-delete"
- the section called "aws:eks:pod-io-stress"
- the section called "aws:eks:pod-memory-stress"
- the section called "aws:eks:pod-network-blackhole-port"
- the section called "aws:eks:pod-network-latency"
- the section called "aws:eks:pod-network-packet-loss"

Limites

- Les actions suivantes ne fonctionnent pas avec AWS Fargate :
  - aws:eks:pod-network-blackhole-port
  - aws:eks:pod-network-latency
  - aws:eks:pod-network-packet-loss
- Les actions suivantes ne sont pas compatibles avec le mode `bridge` réseau :
  - aws:eks:pod-network-blackhole-port
  - aws:eks:pod-network-latency
  - aws:eks:pod-network-packet-loss
- Les actions suivantes nécessitent des autorisations root dans le conteneur éphémère.
  - aws:eks:pod-network-blackhole-port
  - aws:eks:pod-network-latency
  - aws:eks:pod-network-packet-loss

  Le conteneur éphémère héritera de ses autorisations du contexte de sécurité du Pod cible. Si vous devez exécuter les conteneurs du Pod en tant qu'utilisateur non root, vous pouvez définir des contextes de sécurité distincts pour les conteneurs du Pod cible.
- Vous ne pouvez pas identifier les cibles de type aws:eks:pod dans votre modèle d'expérience à l'aide de ressources ou de balises de ressource ARNs . Vous devez identifier les cibles à l'aide des paramètres de ressources requis.

- Les actions aws:eks : pod-network-latency et aws:eks : ne pod-network-packet-loss doivent pas être exécutées en parallèle et cibler le même Pod. Selon la valeur du `maxErrors` paramètre que vous spécifiez, l'action peut se terminer en état terminé ou en échec :
  - Si la valeur `maxErrorsPercent` est 0 (valeur par défaut), l'action se terminera par un échec.
  - Dans le cas contraire, l'échec alourdira le `maxErrorsPercent` budget. Si le nombre d'injections échouées n'atteint pas le nombre indiqué`maxErrors`, l'action sera terminée.
  - Vous pouvez identifier ces défaillances à partir des journaux du conteneur éphémère injecté dans le Pod cible. Cela échouera avec`Exit Code: 16`.
- L'action aws:eks : ne pod-network-blackhole-port doit pas être exécutée en parallèle avec d'autres actions qui ciblent le même Pod et l'utilisent. `trafficType` Les actions parallèles utilisant différents types de trafic sont prises en charge.
- Le FIS ne peut surveiller l'état de l'injection de défauts que `securityContext` lorsque le pod cible est réglé sur. `readOnlyRootFilesystem: false` Sans cette configuration, toutes les actions du EKS Pod échoueront.

Prérequis

- Installez-le AWS CLI sur votre ordinateur. Cela n'est nécessaire que si vous comptez utiliser le AWS CLI pour créer des rôles IAM. Pour plus d'informations, consultez la section Installation ou mise à jour du AWS CLI.
- Installer kubectl sur votre ordinateur. Cela n'est nécessaire que pour interagir avec le cluster EKS afin de configurer ou de surveiller l'application cible. Pour plus d'informations, consultez https://kubernetes. io/docs/tasks/tools/.
- La version minimale prise en charge d'EKS est 1.23.

Création d'un rôle d'expérience

Pour exécuter un test, vous devez configurer un rôle IAM pour le test. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”. Les autorisations requises pour ce rôle dépendent de l'action que vous utilisez. Reportez-vous aux actions AWS FIS ciblées `aws:eks:pod` pour trouver les autorisations nécessaires à votre action.

Configuration du compte de service Kubernetes

Configurez un compte de service Kubernetes pour exécuter des tests avec des cibles dans l'espace de noms Kubernetes spécifié. Dans l'exemple suivant, le compte de service est *myserviceaccount* et l'espace de noms est*default*. Notez qu'il default s'agit de l'un des espaces de noms Kubernetes standard.

Pour configurer votre compte de service Kubernetes

1. Créez un fichier nommé `rbac.yaml` et ajoutez ce qui suit.

```
kind: ServiceAccount
apiVersion: v1
metadata:
  namespace: default
  name: myserviceaccount

---
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  namespace: default
  name: role-experiments
rules:
- apiGroups: [""]
  resources: ["configmaps"]
  verbs: [ "get", "create", "patch", "delete"]
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["create", "list", "get", "delete", "deletecollection"]
- apiGroups: [""]
  resources: ["pods/ephemeralcontainers"]
  verbs: ["update"]
- apiGroups: [""]
  resources: ["pods/exec"]
  verbs: ["create"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["get"]

---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
```

```
metadata:
  name: bind-role-experiments
  namespace: default
subjects:
- kind: ServiceAccount
  name: myserviceaccount
  namespace: default
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: fis-experiment
roleRef:
  kind: Role
  name: role-experiments
  apiGroup: rbac.authorization.k8s.io
```

2. Exécutez la commande suivante.

```
kubectl apply -f rbac.yaml
```

Accorder aux utilisateurs et aux rôles IAM l'accès à Kubernetes APIs

Suivez les étapes expliquées dans la section Associer les identités IAM aux autorisations Kubernetes dans la documentation. EKS

Option 1 : créer des entrées d'accès

Nous vous recommandons de l'utiliser Access Entries en suivant les étapes expliquées dans Accorder aux utilisateurs IAM l'accès à Kubernetes avec des entrées d'accès EKS.

```
aws eks create-access-entry \
                --principal-arn arn:aws:iam::123456789012:role/fis-experiment-role \
                --username fis-experiment \
                --cluster-name my-cluster
```

> **Important**
>
> Afin de tirer parti des entrées d'accès, le mode d'authentification du cluster EKS doit être configuré sur le `API` mode `API_AND_CONFIG_MAP` ou.

Option 2 : ajouter des entrées à l'aws-auth ConfigMap

Vous pouvez également utiliser la commande suivante pour créer un mappage d'identité. Pour plus d'informations, consultez la section Gérer les utilisateurs et les rôles IAM dans la documentation eksctl.

```
eksctl create iamidentitymapping \
                --arn arn:aws:iam::123456789012:role/fis-experiment-role \
                --username fis-experiment \
                --cluster my-cluster
```

> **Important**
>
> L'utilisation de la boîte à outils eksctl pour configurer les mappages d'identité entraînera la création d'entrées dans le. `aws-auth` ConfigMap Il est important de noter que ces entrées générées ne prennent pas en charge l'inclusion d'un composant de chemin. Par conséquent, l'ARN fourni en entrée ne doit pas contenir de segment de chemin (par exemple,`arn:aws:iam::123456789012:role/service-role/fis-experiment-role`).

Images du conteneur Pod

Les images du conteneur Pod fournies par AWS FIS sont hébergées sur Amazon ECR. Lorsque vous référencez une image depuis Amazon ECR, vous devez utiliser l'URI de l'image complète.

L'image du conteneur Pod est également disponible dans la galerie publique AWS ECR.

Région AWS	URI de l'image
USA Est (Ohio)	`051821878176.dkr.ecr.us-east-2.amazonaws.com/aws-fis-pod:0.1`
USA Est (Virginie du Nord)	`731367659002.dkr.ecr.us-east-1.amazonaws.com/aws-fis-pod:0.1`
USA Ouest (Californie du Nord)	`080694859247.dkr.ecr.us-west-1.amazonaws.com/aws-fis-pod:0.1`

Région AWS	URI de l'image
USA Ouest (Oregon)	864386544765.dkr.ecr.us-west-2.amazonaws.com/aws-fis-pod:0.1
Afrique (Le Cap)	056821267933.dkr.ecr.af-south-1.amazonaws.com/aws-fis-pod:0.1
Asie-Pacifique (Hong Kong)	246405402639.dkr.ecr.ap-east-1.amazonaws.com/aws-fis-pod:0.1
Asie-Pacifique (Mumbai)	524781661239.dkr.ecr.ap-south-1.amazonaws.com/aws-fis-pod:0.1
Asia Pacific (Seoul)	526524659354.dkr.ecr.ap-northeast-2.amazonaws.com/aws-fis-pod:0.1
Asie-Pacifique (Singapour)	316401638346.dkr.ecr.ap-southeast-1.amazonaws.com/aws-fis-pod:0.1
Asie-Pacifique (Sydney)	488104106298.dkr.ecr.ap-southeast-2.amazonaws.com/aws-fis-pod:0.1
Asie-Pacifique (Tokyo)	635234321696.dkr.ecr.ap-northeast-1.amazonaws.com/aws-fis-pod:0.1
Canada (Centre)	490658072207.dkr.ecr.ca-central-1.amazonaws.com/aws-fis-pod:0.1
Europe (Francfort)	713827034473.dkr.ecr.eu-central-1.amazonaws.com/aws-fis-pod:0.1
Europe (Irlande)	205866052826.dkr.ecr.eu-west-1.amazonaws.com/aws-fis-pod:0.1
Europe (Londres)	327424803546.dkr.ecr.eu-west-2.amazonaws.com/aws-fis-pod:0.1
Europe (Milan)	478809367036.dkr.ecr.eu-south-1.amazonaws.com/aws-fis-pod:0.1

Région AWS	URI de l'image
Europe (Paris)	154605889247.dkr.ecr.eu-west-3.amazonaws.com/aws-fis-pod:0.1
Europe (Espagne)	395402409451.dkr.ecr.eu-south-2.amazonaws.com/aws-fis-pod:0.1
Europe (Stockholm)	263175118295.dkr.ecr.eu-north-1.amazonaws.com/aws-fis-pod:0.1
Moyen-Orient (Bahreïn)	065825543785.dkr.ecr.me-south-1.amazonaws.com/aws-fis-pod:0.1
Amérique du Sud (São Paulo)	767113787785.dkr.ecr.sa-east-1.amazonaws.com/aws-fis-pod:0.1
AWS GovCloud (USA Est)	246533647532.dkr.ecr.us-gov-east-1.amazonaws.com/aws-fis-pod:0.1
AWS GovCloud (US-Ouest)	246529956514.dkr.ecr.us-gov-west-1.amazonaws.com/aws-fis-pod:0.1

Exemple de modèle d'expérience

Voici un exemple de modèle d'expérience pour l'the section called “aws:eks:pod-network-latency”action.

```
{
    "description": "Add latency and jitter to the network interface for the target EKS
 Pods",
    "targets": {
        "myPods": {
            "resourceType": "aws:eks:pod",
            "parameters": {
                "clusterIdentifier": "mycluster",
                "namespace": "default",
                "selectorType": "labelSelector",
                "selectorValue": "mylabel=mytarget"
            },
```

```
            "selectionMode": "COUNT(3)"
        }
    },
    "actions": {
        "EksPod-latency": {
            "actionId": "aws:eks:pod-network-latency",
            "description": "Add latency",
            "parameters": {
                "kubernetesServiceAccount": "myserviceaccount",
                "duration": "PT5M",
                "delayMilliseconds": "200",
                "jitterMilliseconds": "10",
                "sources": "0.0.0.0/0"
            },
            "targets": {
                "Pods": "myPods"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none",
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/fis-experiment-role",
    "tags": {
        "Name": "EksPodNetworkLatency"
    }
}
```

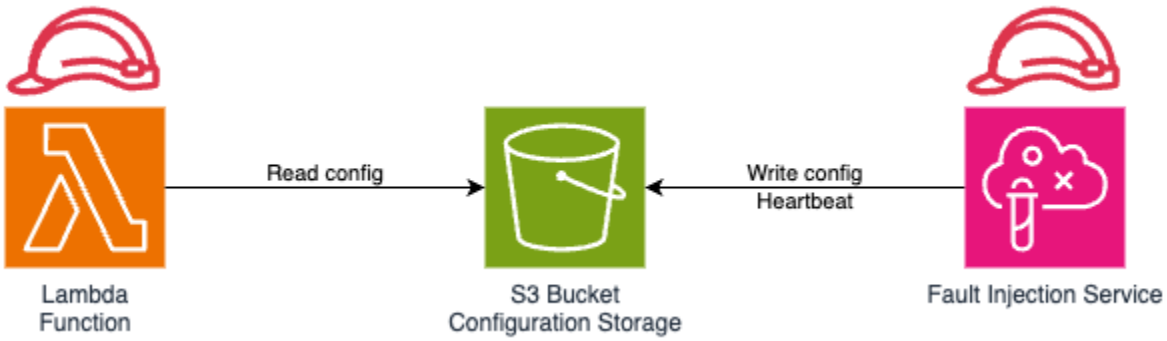
Utilisez les actions AWS FIS aws:lambda:function

Vous pouvez utiliser les actions aws:lambda:function pour injecter des erreurs dans les invocations de vos fonctions. AWS Lambda

Ces actions utilisent une extension AWS FIS gérée pour injecter des erreurs. Pour utiliser les actions aws:lambda:function, vous devez associer l'extension sous forme de couche à vos fonctions Lambda et configurer un compartiment Amazon S3 pour communiquer entre et l'extension. AWS FIS

Lorsque vous exécutez une AWS FIS expérience ciblant aws:lambda:function, que vous lisez la configuration AWS FIS Amazon S3 à partir de votre fonction Lambda et que vous écrivez les

informations relatives à l'injection d'erreurs à l'emplacement Amazon S3 spécifié, comme indiqué dans le schéma ci-dessous.



Actions

- the section called “aws:lambda:invocation-add-delay”
- the section called “aws:lambda:invocation-error”
- the section called “aws:lambda:invocation-http-integration-response”

Limites

- L'extension AWS FIS Lambda ne peut pas être utilisée avec des fonctions utilisant le streaming de réponses. Même si aucune erreur n'est appliquée, l'extension AWS FIS Lambda supprime les configurations de streaming. Pour plus d'informations, voir Streaming de réponses pour les fonctions Lambda dans le guide de l'AWS Lambda utilisateur.

Prérequis

Avant d'utiliser les actions AWS FIS Lambda, assurez-vous d'avoir effectué les tâches ponctuelles suivantes :

- Créez un compartiment Amazon S3 dans la région à partir de laquelle vous souhaitez démarrer une expérience - Vous pouvez utiliser un seul compartiment Amazon S3 pour plusieurs expériences et partager le compartiment entre plusieurs AWS comptes. Cependant, vous devez disposer d'un compartiment distinct pour chacun d'entre eux Région AWS.
- Créez une politique IAM pour accorder un accès en lecture pour l'extension Lambda au compartiment Amazon S3 - Dans le modèle suivant, `my-config-distribution-bucket`

remplacez-le par le nom du compartiment Amazon S3 que vous avez créé ci-dessus `FisConfigs` et par le nom d'un dossier de votre compartiment Amazon S3 que vous souhaitez utiliser.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Sid": "AllowListingConfigLocation",
          "Effect": "Allow",
          "Action": ["s3:ListBucket"],
          "Resource": ["arn:aws:s3:::my-config-distribution-bucket"],
          "Condition": {
              "StringLike": {
                  "s3:prefix": ["FisConfigs/*"]
              }
          }
      },
      {
          "Sid": "AllowReadingObjectFromConfigLocation",
          "Effect": "Allow",
          "Action": "s3:GetObject",
          "Resource": ["arn:aws:s3:::my-config-distribution-bucket/FisConfigs/*"]
      }
  ]
}
```

- Créez une politique IAM pour accorder un accès en écriture au compartiment Amazon S3 pour l' AWS FIS expérience - Dans le modèle suivant, remplacez-le `my-config-distribution-bucket` par le nom du compartiment Amazon S3 que vous avez créé ci-dessus et `FisConfigs` par le nom d'un dossier de votre compartiment Amazon S3 que vous souhaitez utiliser.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Sid": "AllowFisToWriteAndDeleteFaultConfigurations",
          "Effect": "Allow",
          "Action": [
              "s3:PutObject",
```

```
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::my-config-distribution-bucket/FisConfigs/*"
        },
        {
            "Sid": "AllowFisToInspectLambdaFunctions",
            "Effect": "Allow",
            "Action": [
                "lambda:GetFunction"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowFisToDoTagLookups",
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
}
```

Configuration des fonctions Lambda

Suivez les étapes ci-dessous pour chaque fonction Lambda que vous souhaitez influencer :

1. Associez la politique d'accès à la lecture Amazon S3 créée ci-dessus à la fonction Lambda.
2. Attachez l' AWS FIS extension sous forme de couche à la fonction. Pour plus d'informations sur la couche ARNs, consultezVersions disponibles de l' AWS FIS extension pour Lambda.
3. Définissez la `AWS_FIS_CONFIGURATION_LOCATION` variable sur l'ARN du dossier de configuration Amazon S3, par exemple`arn:aws:s3:::my-config-distribution-bucket/FisConfigs/`.
4. Définissez la variable `AWS_LAMBDA_EXEC_WRAPPER` sur `/opt/aws-fis/bootstrap`.

Configuration d'une AWS FIS expérience

Avant de lancer votre test, assurez-vous d'avoir associé la politique d'accès en écriture d'Amazon S3 que vous avez créée dans les conditions préalables aux rôles de test qui utiliseront les actions

AWS FIS Lambda. Pour plus d'informations sur la configuration d'une AWS FIS expérience, consultezGestion des AWS modèles d'expériences FIS.

Journalisation

L'extension AWS FIS Lambda écrit des journaux sur la console et CloudWatch des journaux. La journalisation peut être configurée à l'aide de la `AWS_FIS_LOG_LEVEL` variable. Les valeurs prises en charge sont `INFO`, `WARN` et `ERROR`. Les journaux seront écrits dans le format de journal configuré pour votre fonction Lambda.

Voici un exemple de journal au format texte :

```
2024-08-09T18:51:38.599984Z INFO AWS FIS EXTENSION - extension enabled 1.0.1
```

Voici un exemple de journal au format JSON :

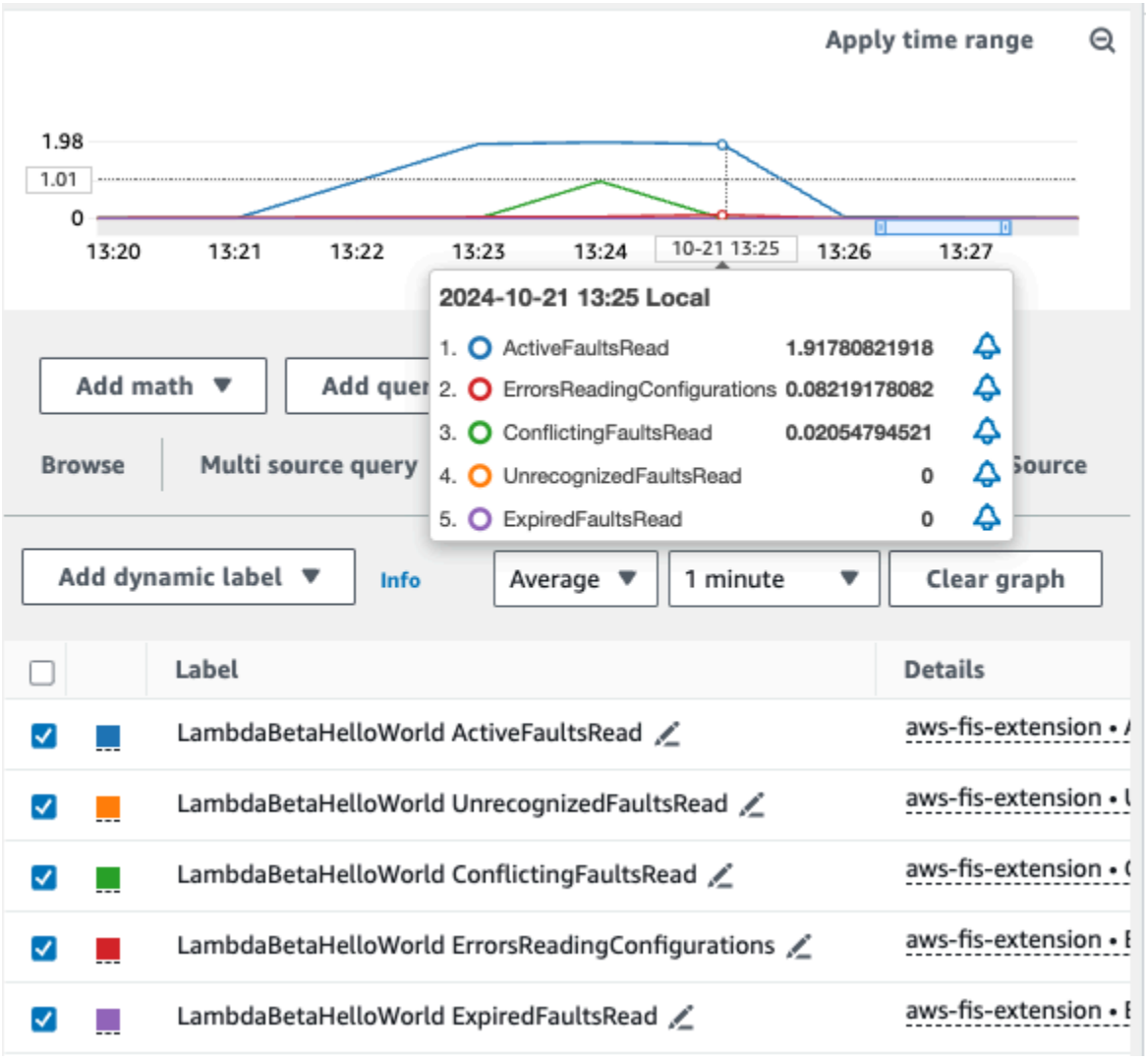
```
{
  "timestamp": "2024-10-08T17:15:36.953905Z",
  "level": "INFO",
  "fields": {
    "message": "AWS FIS EXTENSION - adding 5000 milliseconds of latency to function
 invocation",
    "requestId":"0608bf70-908f-4a17-bbfe-3782cd783d8b"
  }
}
```

Les journaux émis peuvent être utilisés avec les filtres CloudWatch métriques Amazon pour générer des métriques personnalisées. Pour plus d'informations sur les filtres métriques, consultez la section Création de métriques à partir d'événements de journal à l'aide de filtres dans le guide de l'utilisateur Amazon CloudWatch Logs.

Utilisation du format métrique CloudWatch intégré (EMF)

Vous pouvez configurer l'extension AWS FIS Lambda pour émettre des journaux EMF en définissant la `AWS_FIS_EXTENSION_METRICS` variable sur. `all` Par défaut, l'extension n'émet pas de journaux EMF et prend `AWS_FIS_EXTENSION_METRICS` par défaut la valeur. `none` Les journaux EMF sont publiés dans `aws-fis-extension namespace` la CloudWatch console.

Dans l'`aws-fis-extension`espace de noms, vous pouvez sélectionner certaines métriques à afficher dans un graphique. L'exemple ci-dessous montre certaines des métriques disponibles dans l'espace de `aws-fis-extension` noms.



Rubriques avancées

Cette section fournit des informations supplémentaires sur le AWS FIS fonctionnement de l'extension Lambda et des cas d'utilisation particuliers.

Rubriques

- Comprendre les sondages
- Comprendre la simultanéité
- Comprendre le pourcentage d'invocation
- Considérations spéciales pour SnapStart

- Considérations spéciales pour les fonctions rapides et peu fréquentes
- Configuration de plusieurs extensions à l'aide du proxy Lambda Runtime API
- Utilisation AWS FIS avec des environnements d'exécution de conteneurs
- AWS FIS Variables d'environnement Lambda

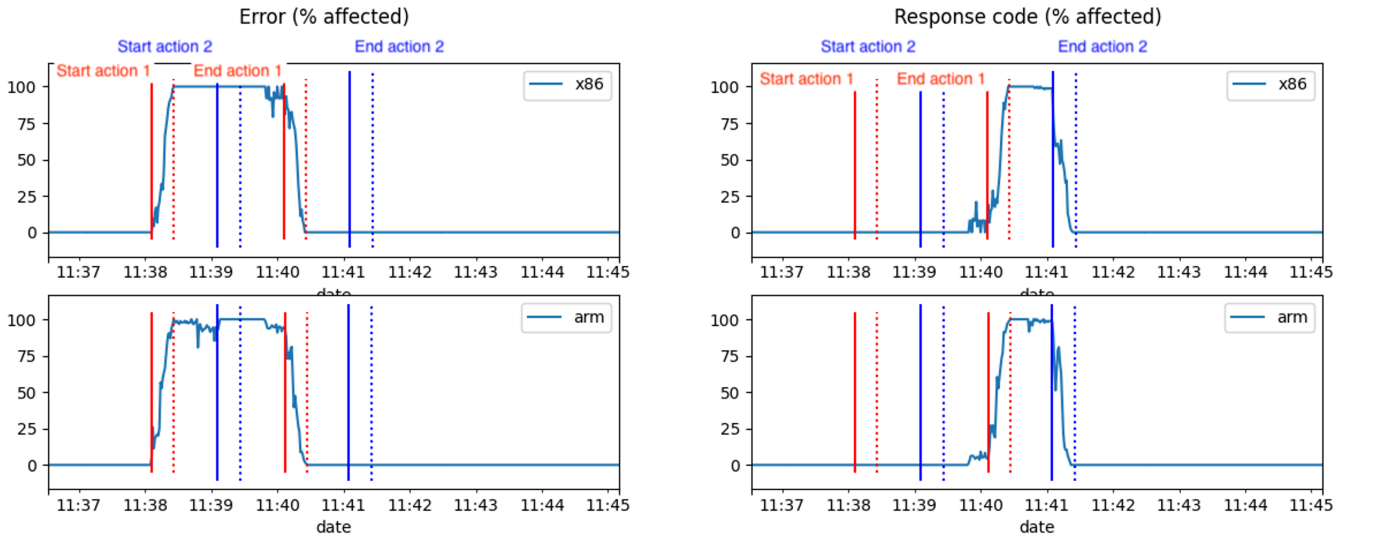
Comprendre les sondages

Vous remarquerez peut-être une période de montée en puissance allant jusqu'à 60 secondes avant que les défauts ne commencent à affecter toutes les invocations. Cela est dû au fait que l'extension Lambda interroge rarement les informations de configuration en attendant le début d'une expérience. Vous pouvez ajuster l'intervalle d'interrogation en définissant la variable d'`AWS_FIS_SLOW_POLL_INTERVAL_SECONDS`environnement (60 s par défaut). Une valeur inférieure entraînera des interrogations plus fréquentes, mais aura un impact plus important sur les performances et les coûts. Vous pouvez également remarquer une période de réduction allant jusqu'à 20 secondes après l'injection du défaut. Cela est dû au fait que l'extension interroge plus fréquemment pendant que les expériences sont en cours.

Comprendre la simultanéité

Vous pouvez cibler les mêmes fonctions Lambda avec plusieurs actions simultanément. Si les actions sont toutes différentes les unes des autres, toutes les actions seront appliquées. Par exemple, vous pouvez ajouter un délai initial avant de renvoyer une erreur. Si deux actions identiques ou contradictoires sont appliquées à la même fonction, seule l'action dont la date de début est la plus ancienne sera appliquée.

La figure ci-dessous montre deux actions contradictoires, aws:lambda:invocation-error et aws:lambda :, qui se chevauchent. invocation-http-integration-response Au départ, aws:lambda:invocation-error démarre à 11 h 38 et s'exécute pendant 2 minutes. Ensuite, aws:lambda : invocation-http-integration-response tente de démarrer à 11 h 39, mais n'entre en vigueur qu'à 11 h 40 après la fin de la première action. Pour maintenir le calendrier des expériences, aws:lambda : se termine invocation-http-integration-response toujours à l'heure initialement prévue, à 11:41.



Comprendre le pourcentage d'invocation

Les actions AWS Fault Injection Service Lambda utilisent une cible aws:lambda:function qui vous permet de sélectionner une ou plusieurs fonctions. AWS Lambda ARNs Grâce à celles-ci ARNs, les actions AWS Fault Injection Service Lambda peuvent injecter des erreurs à chaque appel de la fonction Lambda sélectionnée. Pour vous permettre d'injecter des erreurs dans une fraction des appels uniquement, chaque action vous permet de spécifier un `invocationPercentage` paramètre avec des valeurs comprises entre 0 et 100. À l'aide de ce `invocationPercentage` paramètre, vous pouvez vous assurer que les actions sont simultanées, même pour des pourcentages d'invocation inférieurs à 100 %.

Considérations spéciales pour SnapStart

AWS Lambda les fonctions SnapStart activées auront plus de chances d'attendre pendant toute la durée `AWS_FIS_SLOW_POLL_INTERVAL_SECONDS` avant de détecter la première configuration défectueuse, même si une expérience est déjà en cours. Cela est dû au fait que Lambda SnapStart utilise un seul instantané comme état initial pour plusieurs environnements d'exécution et conserve le stockage temporaire. Pour l'extension AWS Fault Injection Service Lambda, la fréquence des interrogations sera maintenue et la vérification de configuration initiale sera ignorée lors de l'initialisation de l'environnement d'exécution. Pour plus d'informations sur Lambda SnapStart, consultez la section Amélioration des performances de démarrage avec Lambda SnapStart dans le guide de l'utilisateur.AWS Lambda

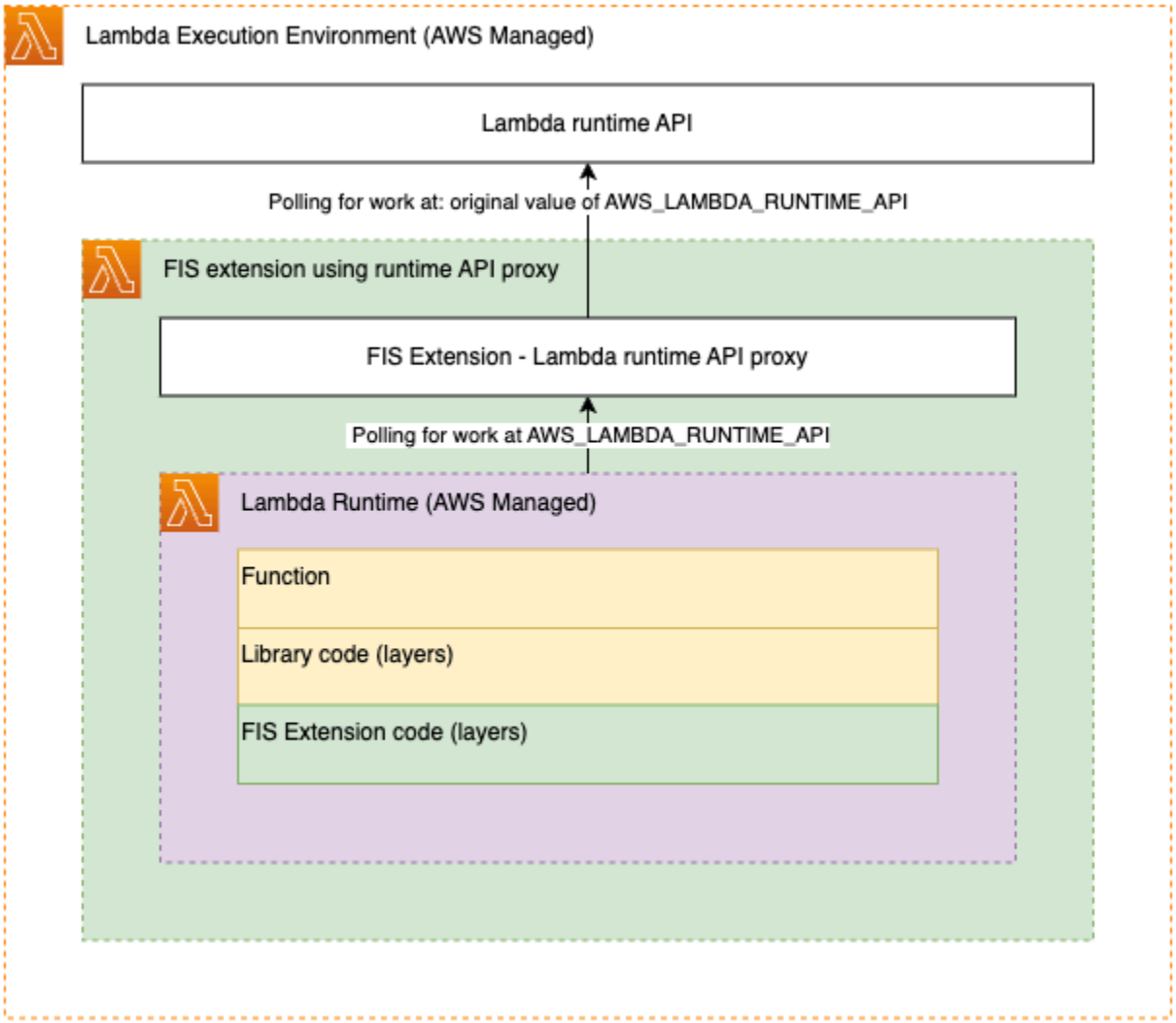
Considérations spéciales pour les fonctions rapides et peu fréquentes

Si votre fonction Lambda s'exécute pendant une durée inférieure à la durée moyenne d'interrogation de 70 millisecondes, le fil de sondage peut avoir besoin de plusieurs appels pour obtenir des configurations de panne. Si la fonction ne s'exécute pas fréquemment, par exemple une fois toutes les 15 minutes, le sondage ne sera jamais terminé. Pour vous assurer que le fil de sondage peut se terminer, définissez le `AWS_FIS_POLL_MAX_WAIT_MILLISECONDS` paramètre. L'extension attendra la durée que vous avez définie pour la fin d'un sondage en vol avant de démarrer la fonction. Notez que cela augmentera la durée de la fonction facturée et entraînera un délai supplémentaire pour certaines invocations.

Configuration de plusieurs extensions à l'aide du proxy Lambda Runtime API

L'extension Lambda utilise le proxy de l'API AWS Lambda Runtime pour intercepter les appels de fonctions avant qu'ils n'atteignent l'environnement d'exécution. Pour ce faire, il expose un proxy pour l'API AWS Lambda Runtime à l'environnement d'exécution et en annonçant son emplacement dans la `AWS_LAMBDA_RUNTIME_API` variable.

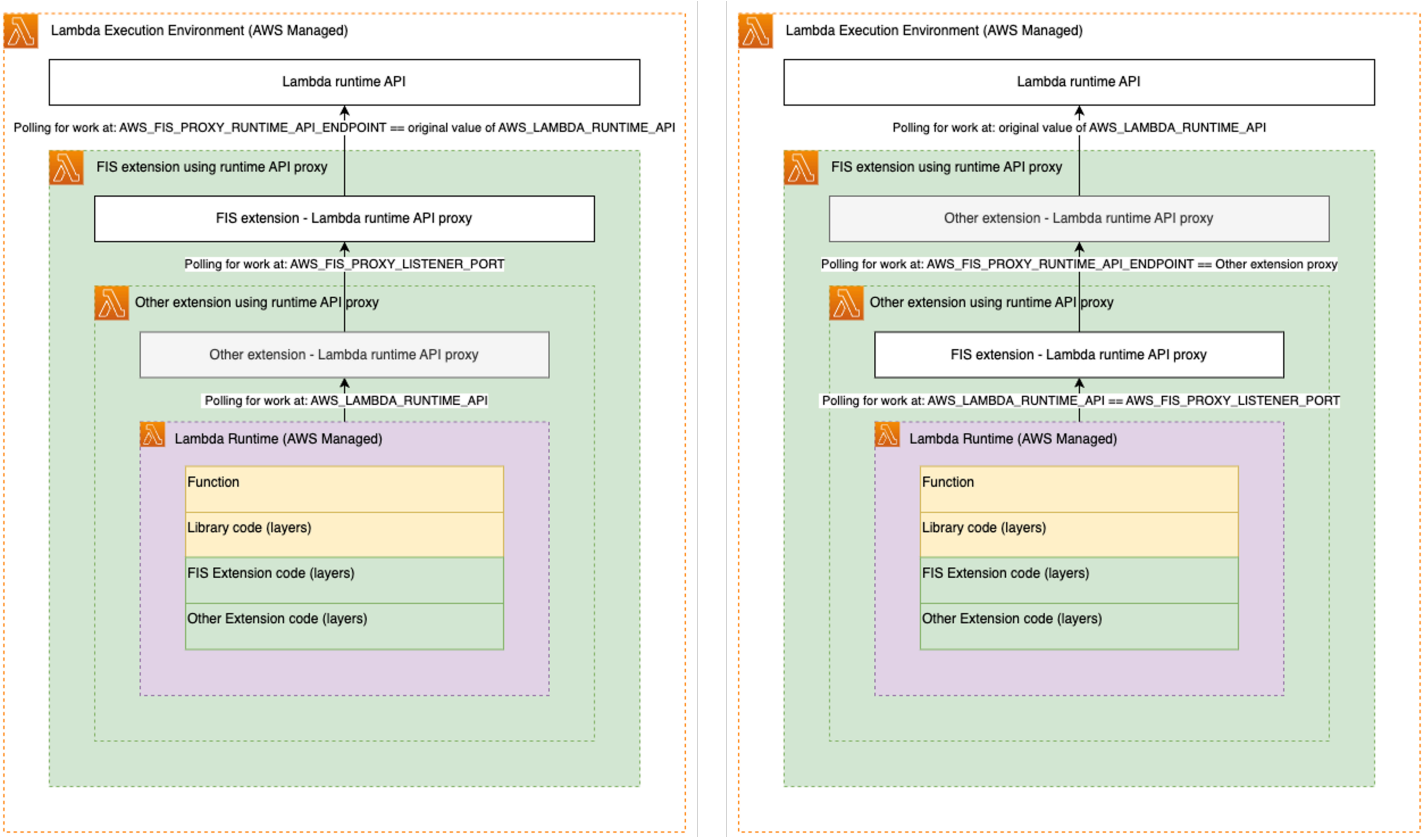
Le schéma suivant montre la configuration d'une seule extension à l'aide du proxy d'API Lambda Runtime :



Pour utiliser l'extension AWS FIS Lambda avec une autre extension utilisant le modèle de proxy AWS Lambda Runtime API, vous devez enchaîner les proxys à l'aide d'un script bootstrap personnalisé. L'extension AWS FIS Lambda accepte les variables d'environnement suivantes :

- `AWS_FIS_PROXY_RUNTIME_API_ENDPOINT`- Prend une chaîne sous la forme `127.0.0.1:9876` représentant l'adresse IP locale et le port d'écoute pour l'API AWS Lambda Runtime. Il peut s'agir de la valeur d'origine `AWS_LAMBDA_RUNTIME_API` ou de l'emplacement d'un autre proxy.
- `AWS_FIS_PROXY_LISTENER_PORT`- Prend un numéro de port sur lequel l' AWS FIS extension doit démarrer son propre proxy, par défaut`9100`.

Avec ces paramètres, vous pouvez enchaîner l' AWS FIS extension avec une autre extension à l'aide du proxy Lambda Runtime API dans deux ordres différents.



Pour plus d'informations sur le proxy AWS Lambda d'API Runtime, consultez les sections Amélioration de la sécurité et de la gouvernance des environnements d'exécution avec l'extension de proxy AWS Lambda Runtime API et Utilisation de l'API d'exécution Lambda pour des environnements d'exécution personnalisés dans le guide de l'AWS Lambda utilisateur.

Utilisation AWS FIS avec des environnements d'exécution de conteneurs

Pour les AWS Lambda fonctions utilisant des images de conteneur qui acceptent la variable d'`AWS_LAMBDA_RUNTIME_API`environnement, vous pouvez intégrer l'extension AWS FIS Lambda dans votre image de conteneur en suivant les étapes ci-dessous :

1. Déterminez l'ARN de la couche à partir de laquelle vous souhaitez extraire l'extension. Pour plus d'informations sur la recherche de l'ARN, consultezConfiguration des fonctions Lambda.
2. Utilisez la AWS Command Line Interface (CLI) pour demander des informations sur l'extension`aws lambda get-layer-version-by-arn --arn fis-extension-arn`. La réponse contiendra un `Location` champ contenant une URL pré-signée à partir de laquelle vous pourrez télécharger l'extension FIS sous forme de fichier ZIP.
3. Décompressez le contenu de l'extension dans votre système `/opt` de fichiers Docker. Voici un exemple de Dockerfile basé sur le runtime Lambda de NodeJS :

```
# extension installation #
FROM amazon/aws-lambda-nodejs:12 AS builder
COPY extension.zip extension.zip
RUN yum install -y unzip
RUN mkdir -p /opt
RUN unzip extension.zip -d /opt
RUN rm -f extension.zip
FROM amazon/aws-lambda-nodejs:12
WORKDIR /opt
COPY --from=builder /opt .
# extension installation finished #
# JS example. Modify as required by your runtime
WORKDIR ${LAMBDA_TASK_ROOT}
COPY index.js package.json .
RUN npm install
CMD [ "index.handler" ]
```

Pour plus d'informations sur les images de conteneur, voir Création d'une fonction Lambda à l'aide d'une image de conteneur dans le guide de l'AWS Lambda utilisateur.

AWS FIS Variables d'environnement Lambda

Voici une liste de variables d'environnement pour l'extension AWS FIS Lambda

- `AWS_FIS_CONFIGURATION_LOCATION`- Obligatoire. Emplacement où les configurations de défaut actives AWS FIS seront écrites et où l'extension lira les configurations de défaillance. Les emplacements doivent être au format Amazon S3 ARN, y compris un compartiment et un chemin. Par exemple, `arn:aws:s3:::my-fis-config-bucket/FisConfigs/`.
- `AWS_LAMBDA_EXEC_WRAPPER`- Obligatoire. Emplacement du script AWS Lambda wrapper utilisé pour configurer l'extension AWS FIS Lambda. Cela doit être défini sur le `/opt/aws-fis/bootstrap` script inclus dans l'extension.
- `AWS_FIS_LOG_LEVEL`- Facultatif. Niveau de journalisation des messages émis par l'extension AWS FIS Lambda. Les valeurs prises en charge sont `INFO`, `WARN` et `ERROR`. Si ce n'est pas le cas, AWS FIS l'extension sera définie par défaut sur`INFO`.
- `AWS_FIS_EXTENSION_METRICS`- Facultatif. Les valeurs possibles sont `all` et `none`. Si elle est définie sur `all` l'extension, elle émettra des métriques EMF sous le`aws-fis-extension namespace`.

- `AWS_FIS_SLOW_POLL_INTERVAL_SECONDS`- Facultatif. Si cette option est définie, l'intervalle d'interrogation (en secondes) sera dépassé pendant que l'extension n'injecte pas de défauts et attend qu'une configuration de défaut soit ajoutée à l'emplacement de configuration. La valeur par défaut est `60`.
- `AWS_FIS_PROXY_RUNTIME_API_ENDPOINT`- Facultatif. Si elle est définie, elle remplacera la valeur de `AWS_LAMBDA_RUNTIME_API` pour définir l'endroit où l' AWS FIS extension interagit avec l'API AWS Lambda d'exécution pour contrôler l'invocation des fonctions. Exige IP:PORT, par exemple,. `127.0.0.1:9000` Pour plus d'informations*`AWS_LAMBDA_RUNTIME_API`*, consultez la section Utilisation de l'API d'exécution Lambda pour des environnements d'exécution personnalisés dans le guide de l'AWS Lambda utilisateur.
- `AWS_FIS_PROXY_LISTENER_PORT`- Facultatif. Définit le port sur lequel l'extension AWS FIS Lambda expose un proxy AWS Lambda d'API d'exécution qui peut être utilisé par une autre extension ou par le moteur d'exécution. La valeur par défaut est `9100`.
- `AWS_FIS_POLL_MAX_WAIT_MILLISECONDS`- Facultatif. Si elle est définie sur une valeur différente de zéro, cette variable définit le nombre de millisecondes pendant lesquelles l'extension attendra la fin d'un sondage asynchrone en cours avant d'évaluer les configurations d'erreur et de lancer l'invocation du runtime. La valeur par défaut est `0`.

Versions disponibles de l' AWS FIS extension pour Lambda

Cette section contient des informations sur les versions de l'extension AWS FIS Lambda. L'extension prend en charge les fonctions Lambda développées pour les plateformes x86-64 et ARM64 (Graviton2). Votre fonction Lambda doit être configurée pour utiliser le nom de ressource Amazon (ARN) spécifique à l' Région AWS endroit où elle est actuellement hébergée. Vous pouvez consulter Région AWS les détails de l'ARN ci-dessous.

Rubriques

- AWS FIS Notes de mise à jour de l'extension Lambda
- Guide d'accès pour l'extension Lambda ARNs

AWS FIS Notes de mise à jour de l'extension Lambda

Le tableau suivant décrit les modifications apportées aux versions récentes de l'extension AWS FIS Lambda

Version	Date de lancement	Remarques
1.0.0	29/10/2024	Première version

Guide d'accès pour l'extension Lambda ARNs

Vous devez avoir au moins un paramètre dans votre Compte AWS et Région AWS avant de pouvoir rechercher des paramètres publics à l'aide de la console. Pour découvrir les paramètres publics, consultez la section Découverte des paramètres publics dans Parameter Store.

Accès à la console :

1. Ouvrez la AWS Systems Manager console à l'adresse https://console.aws.amazon.com/systems-manager/.
2. Dans le panneau de navigation, choisissez Stockage de paramètres.
3. Sélectionnez l'onglet Public parameters (Paramètres publics).
4. Sélectionnez le menu déroulant Select a service (Sélectionner un service) Dans les options de la liste déroulante, choisissez`fis`.
5. (Facultatif) Filtrez les paramètres que vous avez sélectionnés en saisissant plus d'informations dans la barre de recherche. Pour les architectures arm64, filtrez les paramètres en saisissant « arm64 ». Pour les architectures x86_64, filtrez les paramètres en saisissant « x86_64 ».
6. Sélectionnez le paramètre public à utiliser.
7. Dans les détails du paramètre, recherchez la valeur de l'ARN. Copiez l'ARN à utiliser pour configurer les extensions de couche sur vos fonctions Lambda cibles.

AWS CLI Accès :

Noms des paramètres SSM

Les noms de paramètres SSM suivants sont disponibles pour différentes architectures :

1. bras 64 : `/aws/service/fis/lambda-extension/AWS-FIS-extension-arm64/1.x.x`
2. x86_64 : `/aws/service/fis/lambda-extension/AWS-FIS-extension-x86_64/1.x.x`

AWS CLI Format de commande

Pour récupérer l'extension ARNs, utilisez le format de AWS CLI commande suivant où ParameterName est le nom de l'architecture et la région est la cible : Région AWS

```
aws ssm get-parameter --name parameterName --region region
```

Exemple d'utilisation

```
aws ssm get-parameter --name /aws/service/fis/lambda-extension/AWS-FIS-extension-
x86_64/1.x.x --region ap-southeast-2
```

Format de la réponse

La commande renvoie un objet JSON contenant les détails des paramètres, comme suit. L'ARN de la couche Lambda est inclus dans le champ Value de l'objet Parameter. Copiez l'ARN à utiliser pour configurer les extensions de couche sur vos fonctions Lambda cibles.

```
 {
     "Parameter":
        {
             "Name": "/aws/service/fis/lambda-extension/AWS-FIS-extension-
x86_64/1.x.x",
             "Type": "String",
             "Value": "arn:aws:lambda:ap-southeast-2:211125361907:layer:aws-fis-
extension-x86_64:9",
             "Version": 1,
             "LastModifiedDate": "2025-01-02T15:13:54.465000-05:00",
             "ARN": "arn:aws:ssm:ap-southeast-2::parameter/aws/service/fis/lambda-
extension/AWS-FIS-extension-x86_64/1.x.x",
             "DataType": "text"
        }
 }
```

Accès programmatique :

Récupérez ces paramètres publics par programmation lors de la création ou de la configuration de vos fonctions Lambda à l'aide de l'infrastructure en tant que code (IaC). Cette approche permet de maintenir vos fonctions Lambda avec l'ARN de la dernière version de couche sans nécessiter les

mises à jour manuelles du code qui seraient nécessaires si l'ARN de la couche d' AWS FIS extension était codé en dur. Les ressources suivantes montrent comment récupérer des paramètres publics à l'aide de plateformes IaC courantes :

- Obtenir des paramètres publics à l'aide du AWS SDK
- Obtenir des paramètres publics depuis AWS Systems Manager Parameter Store
- Obtenir des paramètres publics à l'aide de Terraform

Gestion des AWS modèles d'expériences FIS

Vous pouvez créer et gérer des modèles d'expériences à l'aide de la console AWS FIS ou de la ligne de commande. Un modèle d'expérience contient une ou plusieurs actions à exécuter sur des cibles spécifiées au cours d'une expérience. Il contient également les conditions d'arrêt qui empêchent l'expérience de sortir des limites. Pour plus d'informations sur les composants d'un modèle d'expérience, consultezComposants du modèle d'expérience. Après avoir créé un modèle de test, vous pouvez l'utiliser pour exécuter un test.

Tâches

- Création d'un modèle d'expérience
- Afficher les modèles d'expériences
- Génération d'un aperçu de la cible à partir d'un modèle d'expérience
- Lancer une expérience à partir d'un modèle
- Mettre à jour un modèle d'expérience
- Modèles d'expériences de tags
- Supprimer un modèle d'expérience
- Exemples de modèles d'expériences AWS FIS

Création d'un modèle d'expérience

Avant de commencer, effectuez les tâches suivantes :

- Planifiez votre expérience.
- Créez un rôle IAM qui accorde au service AWS FIS l'autorisation d'effectuer des actions en votre nom. Pour de plus amples informations, veuillez consulter Rôles IAM pour les expériences AWS FIS.
- Assurez-vous d'avoir accès au AWS FIS. Pour plus d'informations, consultez les exemples de politiques AWS FIS.

Pour créer un modèle d'expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.

3. Choisissez Créer un modèle d'expérience.
4. Pour l'étape 1, Spécifier les détails du modèle, procédez comme suit :

   a. Dans Description et nom, entrez une description du modèle, telle que`Amazon S3 Network Disrupt Connectivity`.

   b. (Facultatif) Pour le ciblage des comptes, choisissez Plusieurs comptes pour configurer un modèle d'expérience multi-comptes.

   c. Choisissez Suivant, puis passez à l'étape 2, Spécifier les actions et les cibles.
5. Pour Actions, spécifiez l'ensemble d'actions pour le modèle. Pour chaque action, choisissez Ajouter une action et effectuez les opérations suivantes :

   - Dans Nom, entrez le nom de l'action.

     Les caractères autorisés sont les caractères alphanumériques, les traits d'union (-) et les traits de soulignement (_). Le nom doit commencer par une lettre. Les espaces ne sont pas autorisés. Chaque nom d'action doit être unique dans ce modèle.
   - (Facultatif) Dans Description, entrez une description de l'action. La longueur maximale est de 512 caractères.
   - (Facultatif) Pour Démarrer après, sélectionnez une autre action définie dans ce modèle qui doit être terminée avant le début de l'action en cours. Dans le cas contraire, l'action s'exécute au début de l'expérience.
   - Pour Type d'action, choisissez l'action AWS FIS.
   - Pour Target, choisissez une cible que vous avez définie dans la section Cibles. Si vous n'avez pas encore défini de cible pour cette action, AWS FIS en crée une nouvelle pour vous.
   - Pour Paramètres d'action, spécifiez les paramètres de l'action. Cette section apparaît uniquement si l'action AWS FIS comporte des paramètres.
   - Choisissez Save (Enregistrer).
6. Pour les cibles, définissez les ressources cibles sur lesquelles effectuer les actions. Vous devez spécifier au moins un ID de ressource ou une balise de ressource comme cible. Choisissez Modifier pour modifier la cible que AWS FIS a créée pour vous à l'étape précédente, ou choisissez Ajouter une cible. Pour chaque cible, procédez comme suit :

   - Dans Nom, entrez le nom de la cible.

Les caractères autorisés sont les caractères alphanumériques, les traits d'union (-) et les traits de soulignement (_). Le nom doit commencer par une lettre. Les espaces ne sont pas autorisés. Chaque nom de cible doit être unique dans ce modèle.

- Pour Type de ressource, choisissez un type de ressource pris en charge pour l'action.
- Pour la méthode Target, effectuez l'une des opérations suivantes :
  - Choisissez Ressource, IDs puis choisissez ou ajoutez la ressource IDs.
  - Choisissez Balises, filtres et paramètres de ressource, puis ajoutez les balises et les filtres dont vous avez besoin. Pour de plus amples informations, veuillez consulter the section called “Identifier les ressources cibles”.
- Pour le mode sélection, choisissez Count pour exécuter l'action sur le nombre spécifié de cibles identifiées ou choisissez Percent pour exécuter l'action sur le pourcentage spécifié de cibles identifiées. Par défaut, l'action s'exécute sur toutes les cibles identifiées.
- Choisissez Save (Enregistrer).

7. Pour mettre à jour une action avec la cible que vous avez créée, recherchez l'action sous Actions, choisissez Modifier, puis mettez à jour la cible. Vous pouvez utiliser le même objectif pour plusieurs actions.
8. (Facultatif) Pour les options d'expérimentation, sélectionnez le comportement du mode de résolution cible vide.
9. Choisissez Suivant pour passer à l'étape 3, Configurer l'accès au service.
10. Pour l'accès aux services, choisissez Utiliser un rôle IAM existant, puis choisissez le rôle IAM que vous avez créé, comme décrit dans les conditions préalables de ce didacticiel. Si votre rôle n'est pas affiché, vérifiez qu'il possède la relation de confiance requise. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.
11. (Expériences multi-comptes uniquement) Pour les configurations de compte Target, ajoutez un ARN de rôle et une description facultative pour chaque compte cible. Pour télécharger le rôle ARNs du compte cible dans un fichier CSV, choisissez Télécharger le rôle ARNs pour tous les comptes cibles, puis choisissez Choisir un fichier .CSV
12. Choisissez Suivant pour passer à l'étape 4, Configurer les paramètres facultatifs.
13. (Facultatif) Pour les conditions d'arrêt, sélectionnez les CloudWatch alarmes Amazon pour les conditions d'arrêt. Pour de plus amples informations, veuillez consulter Conditions d'arrêt pour AWS FIS.
14. (Facultatif) Pour les journaux, configurez l'option de destination. Pour envoyer des journaux vers un compartiment S3, choisissez Envoyer vers un compartiment Amazon S3 et entrez le nom et

le préfixe du compartiment. Pour envoyer des CloudWatch journaux à Logs, choisissez Send to CloudWatch Logs et entrez le groupe de journaux.

15. (Facultatif) Pour les balises, choisissez Ajouter une nouvelle balise et spécifiez une clé de balise et une valeur de balise. Les balises que vous ajoutez sont appliquées à votre modèle d'expérience, et non aux expériences exécutées à l'aide du modèle.
16. Choisissez Suivant pour passer à l'étape 5, Réviser et créer.
17. Passez en revue le modèle et choisissez Créer un modèle d'expérience. Lorsque vous êtes invité à confirmer, entrez`create`, puis choisissez Créer un modèle d'expérience.

Pour créer un modèle d'expérience à l'aide de la CLI

Utilisez la commande create-experiment-template.

Vous pouvez charger un modèle d'expérience à partir d'un fichier JSON.

Utilisez le `--cli-input-json` paramètre.

```
aws fis create-experiment-template --cli-input-json fileb://<path-to-json-file>
```

Pour plus d'informations, consultez la section Génération d'un modèle de squelette de CLI dans le guide de AWS Command Line Interface l'utilisateur. Pour des exemples de modèles, voirExemples de modèles d'expériences AWS FIS.

Afficher les modèles d'expériences

Vous pouvez consulter les modèles d'expériences que vous avez créés.

Pour afficher un modèle d'expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Pour afficher les informations relatives à un modèle spécifique, sélectionnez l'ID du modèle d'expérience.
4. Dans la section Détails, vous pouvez consulter la description et les conditions d'arrêt du modèle.
5. Pour afficher les actions du modèle d'expérience, choisissez Actions.
6. Pour afficher les cibles du modèle d'expérience, choisissez Targets.
7. Pour afficher les balises du modèle d'expérience, choisissez Tags.

Pour afficher un modèle d'expérience à l'aide de la CLI

Utilisez la list-experiment-templatescommande pour obtenir une liste de modèles d'expériences et utilisez la get-experiment-templatecommande pour obtenir des informations sur un modèle d'expérience spécifique.

Génération d'un aperçu de la cible à partir d'un modèle d'expérience

Avant de commencer une expérience, vous pouvez générer un aperçu de la cible pour vérifier que votre modèle d'expérience est configuré pour cibler les ressources attendues. Les ressources ciblées lorsque vous commencez l'expérience réelle peuvent être différentes de celles de l'aperçu, car les ressources peuvent être supprimées, mises à jour ou échantillonnées de manière aléatoire. Lorsque vous générez un aperçu de la cible, vous lancez une expérience qui ignore toutes les actions.

> **Note**
>
> La génération d'un aperçu cible ne vous permet pas de vérifier que vous disposez des autorisations nécessaires pour effectuer des actions sur vos ressources.

Pour démarrer un aperçu de la cible à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Pour afficher les cibles du modèle d'expérience, choisissez Targets.
4. Pour vérifier vos ressources cibles pour le modèle d'expérience, choisissez Generate Preview. Lorsque vous exécutez un test, cet aperçu des cibles est automatiquement mis à jour avec les cibles du test le plus récent.

Pour démarrer un aperçu de la cible à l'aide de la CLI

- Exécutez la commande start-experiment suivante. Remplacez les valeurs en italique par vos propres valeurs.

```
aws fis start-experiment \
    --experiment-options actionsMode=skip-all \
```

```
    --experiment-template-id EXTxxxxxxxxx
```

Lancer une expérience à partir d'un modèle

Après avoir créé un modèle d'expérience, vous pouvez démarrer des expériences à l'aide de ce modèle.

Lorsque vous lancez un test, nous créons un instantané du modèle spécifié et nous utilisons cet instantané pour exécuter le test. Par conséquent, si le modèle d'expérience est mis à jour ou supprimé pendant l'exécution de l'expérience, ces modifications n'ont aucun impact sur l'expérience en cours.

Lorsque vous lancez une expérience, AWS FIS crée un rôle lié à un service en votre nom. Pour de plus amples informations, veuillez consulter Utiliser des rôles liés à un service pour le service d'injection de AWS défauts.

Après avoir démarré l'expérience, vous pouvez l'arrêter à tout moment. Pour de plus amples informations, veuillez consulter Arrêt d'une expérience.

Pour démarrer une expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Démarrer l'expérience.
4. (Facultatif) Pour ajouter une balise à votre expérience, choisissez Ajouter une nouvelle balise et entrez une clé de balise et une valeur de balise.
5. Sélectionnez Start experiment (Démarrer une expérience). Lorsque vous êtes invité à confirmer, entrez **`start`** et choisissez Démarrer l'expérience.

Pour démarrer une expérience à l'aide de la CLI

Utilisez la commande start-experiment.

Mettre à jour un modèle d'expérience

Vous pouvez mettre à jour un modèle d'expérience existant. Lorsque vous mettez à jour un modèle d'expérience, les modifications n'affectent pas les expériences en cours utilisant le modèle.

Pour mettre à jour un modèle d'expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Mettre à jour le modèle d'expérience.
4. Modifiez les détails du modèle selon vos besoins, puis choisissez Mettre à jour le modèle d'expérience.

Pour mettre à jour un modèle d'expérience à l'aide de la CLI

Utilisez la commande update-experiment-template.

Modèles d'expériences de tags

Vous pouvez appliquer vos propres balises aux modèles d'expérimentation pour vous aider à les organiser. Vous pouvez également implémenter des politiques IAM basées sur des balises pour contrôler l'accès aux modèles d'expériences.

Pour baliser un modèle d'expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience et choisissez Actions, Gérer les balises.
4. Pour ajouter une nouvelle balise, choisissez Ajouter une nouvelle balise, puis spécifiez une clé et une valeur.

   Pour supprimer une étiquette, choisissez Supprimer pour la balise.
5. Choisissez Save (Enregistrer).

Pour baliser un modèle d'expérience à l'aide de la CLI

Utilisez la commande tag-resource.

Supprimer un modèle d'expérience

Si vous n'avez plus besoin d'un modèle de test, vous pouvez le supprimer. Lorsque vous supprimez un modèle d'expérience, les expériences en cours utilisant le modèle ne sont pas affectées. L'expérience continue de se dérouler jusqu'à ce qu'elle soit terminée ou arrêtée. Toutefois, les modèles d'expériences supprimés ne peuvent pas être consultés sur la page Expériences de la console.

Pour supprimer un modèle d'expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Supprimer le modèle d'expérience.
4. Lorsque vous êtes invité à confirmer, entrez **`delete`** et choisissez Supprimer le modèle d'expérience.

Pour supprimer un modèle d'expérience à l'aide de la CLI

Utilisez la commande delete-experiment-template.

Exemples de modèles d'expériences AWS FIS

Si vous utilisez l'API AWS FIS ou un outil de ligne de commande pour créer un modèle d'expérience, vous pouvez créer le modèle en notation d' JavaScript objet (JSON). Pour plus d'informations sur les composants d'un modèle d'expérience, consultezAWS Composants du modèle d'expérience FIS.

Pour créer un test à l'aide de l'un des modèles d'exemple, enregistrez-le dans un fichier JSON (par exemple,`my-template.json`), remplacez les valeurs de l'espace réservé *`italics`* par vos propres valeurs, puis exécutez la create-experiment-templatecommande suivante.

```
aws fis create-experiment-template --cli-input-json file://my-template.json
```

Exemple de modèles

- Arrêter EC2 les instances en fonction de filtres
- Arrêter un nombre spécifié d' EC2 instances

- Exécuter un document AWS FIS SSM préconfiguré
- Exécuter un runbook d'automatisation prédéfini
- Limiter les actions d'API sur les EC2 instances dotées du rôle IAM cible
- Test de résistance du processeur des pods dans un cluster Kubernetes

Arrêter EC2 les instances en fonction de filtres

L'exemple suivant arrête toutes les EC2 instances Amazon en cours d'exécution dans la région spécifiée avec la balise spécifiée dans le VPC spécifié. Il les redémarre au bout de deux minutes.

```
{
    "tags": {
        "Name": "StopEC2InstancesWithFilters"
    },
    "description": "Stop and restart all instances in us-east-1b with the tag env=prod
 in the specified VPC",
    "targets": {
        "myInstances": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "env": "prod"
            },
            "filters": [
                {
                    "path": "Placement.AvailabilityZone",
                    "values": ["us-east-1b"]
                },
                {
                    "path": "State.Name",
                    "values": ["running"]
                },
                {
                    "path": "VpcId",
                    "values": [ "vpc-aabbcc11223344556"]
                }
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "StopInstances": {
```

```
            "actionId": "aws:ec2:stop-instances",
            "description": "stop the instances",
            "parameters": {
                "startInstancesAfterDuration": "PT2M"
            },
            "targets": {
                "Instances": "myInstances"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Arrêter un nombre spécifié d' EC2 instances

L'exemple suivant arrête trois instances avec la balise spécifiée. AWS Le FIS sélectionne les instances spécifiques à arrêter de manière aléatoire. Il redémarre ces instances au bout de deux minutes.

```
{
    "tags": {
        "Name": "StopEC2InstancesByCount"
    },
    "description": "Stop and restart three instances with the specified tag",
    "targets": {
        "myInstances": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "env": "prod"
            },
            "selectionMode": "COUNT(3)"
        }
    },
    "actions": {
        "StopInstances": {
            "actionId": "aws:ec2:stop-instances",
            "description": "stop the instances",
```

```
            "parameters": {
                "startInstancesAfterDuration": "PT2M"
            },
            "targets": {
                "Instances": "myInstances"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Exécuter un document AWS FIS SSM préconfiguré

L'exemple suivant exécute une injection de panne du processeur pendant 60 secondes sur l' EC2 instance spécifiée à l'aide d'un document AWS FIS SSM préconfiguré, -CPU-Stress. AWSFIS-Run AWS Le FIS surveille l'expérience pendant deux minutes.

```
{
    "tags": {
        "Name": "CPUStress"
    },
    "description": "Run a CPU fault injection on the specified instance",
    "targets": {
        "myInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": ["arn:aws:ec2:us-east-1:111122223333:instance/instance-
id"],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "CPUStress": {
            "actionId": "aws:ssm:send-command",
            "description": "run cpu stress using ssm",
            "parameters": {
                "duration": "PT2M",
                "documentArn": "arn:aws:ssm:us-east-1::document/AWSFIS-Run-CPU-Stress",
```

```
                "documentParameters": "{\"DurationSeconds\": \"60\",
 \"InstallDependencies\": \"True\", \"CPU\": \"0\"}"
            },
            "targets": {
                "Instances": "myInstance"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Exécuter un runbook d'automatisation prédéfini

L'exemple suivant publie une notification sur Amazon SNS à l'aide d'un runbook fourni par Systems Manager, AWS-Publish. SNSNotification Le rôle doit être autorisé à publier des notifications sur le sujet SNS spécifié.

```
{
    "description": "Publish event through SNS",
    "stopConditions": [
        {
            "source": "none"
        }
    ],
    "targets": {
    },
    "actions": {
        "sendToSns": {
            "actionId": "aws:ssm:start-automation-execution",
            "description": "Publish message to SNS",
            "parameters": {
                "documentArn": "arn:aws:ssm:us-east-1::document/AWS-
PublishSNSNotification",
                "documentParameters": "{\"Message\": \"Hello, world\", \"TopicArn\":
 \"arn:aws:sns:us-east-1:111122223333:topic-name\"}",
                "maxDuration": "PT1M"
            },
```

```
            "targets": {
            }
        }
    },
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Limiter les actions d'API sur les EC2 instances dotées du rôle IAM cible

L'exemple suivant limite 100 % des appels d'API spécifiés dans la définition d'action pour les appels d'API effectués par le ou les rôles IAM spécifiés dans la définition de cible.

> **Note**
>
> Si vous souhaitez cibler des EC2 instances membres d'un groupe Auto Scaling, utilisez l'action aws:ec2 : asg-insufficient-instance-capacity -error et ciblez plutôt par groupe Auto Scaling. Pour de plus amples informations, veuillez consulter aws:ec2:asg-insufficient-instance-capacity-error.

```
{
    "tags": {
        "Name": "ThrottleEC2APIActions"
    },
    "description": "Throttle the specified EC2 API actions on the specified IAM role",
    "targets": {
        "myRole": {
            "resourceType": "aws:iam:role",
            "resourceArns": ["arn:aws:iam::111122223333:role/role-name"],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "ThrottleAPI": {
            "actionId": "aws:fis:inject-api-throttle-error",
            "description": "Throttle APIs for 5 minutes",
            "parameters": {
                "service": "ec2",
                "operations": "DescribeInstances,DescribeVolumes",
                "percentage": "100",
                "duration": "PT2M"
```

```
            },
            "targets": {
                "Roles": "myRole"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:alarm-name"
        }
    ],
    "roleArn": "arn:aws:iam::111122223333:role/role-name"
}
```

Test de résistance du processeur des pods dans un cluster Kubernetes

L'exemple suivant utilise Chaos Mesh pour tester le stress du processeur des pods dans un cluster Amazon EKS Kubernetes pendant une minute.

```
{
    "description": "ChaosMesh StressChaos example",
    "targets": {
        "Cluster-Target-1": {
            "resourceType": "aws:eks:cluster",
            "resourceArns": [
                "arn:aws:eks:arn:aws::111122223333:cluster/cluster-id"
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "TestCPUStress": {
            "actionId": "aws:eks:inject-kubernetes-custom-resource",
            "parameters": {
                "maxDuration": "PT2M",
                "kubernetesApiVersion": "chaos-mesh.org/v1alpha1",
                "kubernetesKind": "StressChaos",
                "kubernetesNamespace": "default",
                "kubernetesSpec": "{\"selector\":{\"namespaces\":[\"default\"],
\"labelSelectors\":{\"run\":\"nginx\"}},\"mode\":\"all\",\"stressors\": {\"cpu\":
{\"workers\":1,\"load\":50}},\"duration\":\"1m\"}"
            },
```

```
            "targets": {
                "Cluster": "Cluster-Target-1"
            }
        }
    },
    "stopConditions": [{
        "source": "none"
    }],
    "roleArn": "arn:aws:iam::111122223333:role/role-name",
    "tags": {}
}
```

L'exemple suivant utilise Litmus pour tester le stress du processeur des pods dans un cluster Amazon EKS Kubernetes pendant une minute.

```
{
    "description": "Litmus CPU Hog",
    "targets": {
        "MyCluster": {
            "resourceType": "aws:eks:cluster",
            "resourceArns": [
                "arn:aws:eks:arn:aws::111122223333:cluster/cluster-id"
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "MyAction": {
            "actionId": "aws:eks:inject-kubernetes-custom-resource",
            "parameters": {
                "maxDuration": "PT2M",
                "kubernetesApiVersion": "litmuschaos.io/v1alpha1",
                "kubernetesKind": "ChaosEngine",
                "kubernetesNamespace": "litmus",
                "kubernetesSpec": "{\"engineState\":\"active\",\"appinfo\":
{\"appns\":\"default\",\"applabel\":\"run=nginx\",\"appkind\":\"deployment\"},
\"chaosServiceAccount\":\"litmus-admin\",\"experiments\":[{\"name\":\"pod-cpu-hog
\",\"spec\":{\"components\":{\"env\":[{\"name\":\"TOTAL_CHAOS_DURATION\",\"value\":
\"60\"},{\"name\":\"CPU_CORES\",\"value\":\"1\"},{\"name\":\"PODS_AFFECTED_PERC\",
\"value\":\"100\"},{\"name\":\"CONTAINER_RUNTIME\",\"value\":\"docker\"},{\"name\":
\"SOCKET_PATH\",\"value\":\"/var/run/docker.sock\"}]},\"probe\":[]}}],\"annotationCheck
\":\"false\"}"
            },
```

```
            "targets": {
                "Cluster": "MyCluster"
            }
        }
    },
    "stopConditions": [{
        "source": "none"
    }],
    "roleArn": "arn:aws:iam::111122223333:role/role-name",
    "tags": {}
}
```

Gestion de vos AWS expériences FIS

AWS FIS vous permet de réaliser des expériences d'injection de défauts sur vos charges AWS de travail. Pour commencer, créez un modèle d'expérience. Après avoir créé un modèle de test, vous pouvez l'utiliser pour démarrer un test.

Une expérience est terminée lorsque l'une des situations suivantes se produit :

- Toutes les actions du modèle ont été effectuées avec succès.
- Une condition d'arrêt est déclenchée.
- Impossible d'effectuer une action en raison d'une erreur. Par exemple, si la cible est introuvable.
- L'expérience est arrêtée manuellement.

Vous ne pouvez pas reprendre une expérience interrompue ou échouée. Vous ne pouvez pas non plus réexécuter un test terminé. Toutefois, vous pouvez démarrer une nouvelle expérience à partir du même modèle d'expérience. Vous pouvez éventuellement mettre à jour le modèle d'expérience avant de le spécifier à nouveau dans une nouvelle expérience.

Tâches

- Lancer une expérience
- Afficher vos expériences
- Marquer une expérience
- Arrêt d'une expérience
- Lister les cibles résolues

Lancer une expérience

Vous démarrez une expérience à partir d'un modèle d'expérience. Pour de plus amples informations, veuillez consulter Lancer une expérience à partir d'un modèle.

Vous pouvez planifier vos expériences sous forme de tâche ponctuelle ou de tâches récurrentes à l'aide de Amazon EventBridge. Pour de plus amples informations, veuillez consulter Tutoriel : planifier une expérience récurrente.

Vous pouvez suivre votre expérience à l'aide de l'une des fonctionnalités suivantes :

- Consultez vos expériences dans la console AWS FIS. Pour de plus amples informations, veuillez consulter Afficher vos expériences.
- Consultez CloudWatch les statistiques Amazon relatives aux ressources cibles de vos tests ou consultez les statistiques d'utilisation du AWS FIS. Pour de plus amples informations, veuillez consulter Surveiller en utilisant CloudWatch.
- Activez la journalisation des expériences pour capturer des informations détaillées sur votre expérience au fur et à mesure de son exécution. Pour de plus amples informations, veuillez consulter Enregistrement des expériences.

Afficher vos expériences

Vous pouvez consulter la progression d'une expérience en cours, ainsi que les expériences terminées, arrêtées ou ayant échoué.

Les tests interrompus, terminés ou échoués sont automatiquement supprimés de votre compte au bout de 120 jours.

Pour afficher les expériences à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Experiments.
3. Choisissez l'identifiant de l'expérience pour ouvrir sa page de détails.
4. Effectuez une ou plusieurs des actions suivantes :
   - Vérifiez les détails, l'État pour connaître l'état de l'expérience.
   - Cliquez sur l'onglet Actions pour obtenir des informations sur les actions de l'expérience.
   - Cliquez sur l'onglet Cibles pour obtenir des informations sur les cibles de l'expérience.
   - Choisissez l'onglet Chronologie pour obtenir une représentation visuelle des actions en fonction de leur heure de début et de fin.

Pour visualiser les expériences à l'aide de la CLI

Utilisez la commande list-experiments pour obtenir une liste d'expériences, et utilisez la commande get-experiment pour obtenir des informations sur une expérience spécifique.

États de l'expérience

Une expérience peut se trouver dans l'un des états suivants :

- en attente — L'expérience est en attente.
- lancement — L'expérience est sur le point de démarrer.
- en cours — L'expérience est en cours.
- terminé — Toutes les actions de l'expérience se sont terminées avec succès.
- arrêt — La condition d'arrêt a été déclenchée ou l'expérience a été arrêtée manuellement.
- stoppé — Toutes les actions en cours ou en attente dans le cadre de l'expérience sont arrêtées.
- échec — L'expérience a échoué en raison d'une erreur, telle que des autorisations insuffisantes ou une syntaxe incorrecte.
- annulé — L'expérience a été arrêtée ou empêchée de démarrer en raison d'un levier de sécurité enclenché.

États d'action

Une action peut présenter l'un des états suivants :

- en attente : l'action est en attente, soit parce que l'expérience n'a pas commencé, soit parce que l'action doit démarrer plus tard dans l'expérience.
- lancement — L'action est sur le point de démarrer.
- running — L'action est en cours d'exécution.
- terminé — L'action s'est terminée avec succès.
- annulé — L'expérience s'est arrêtée avant le début de l'action.
- ignoré — L'action a été ignorée.
- arrêt — L'action s'arrête.
- stoppé — Toutes les actions en cours ou en attente dans le cadre de l'expérience sont arrêtées.
- échec — L'action a échoué en raison d'une erreur du client, telle que des autorisations insuffisantes ou une syntaxe incorrecte.

Marquer une expérience

Vous pouvez appliquer des balises aux expériences pour vous aider à les organiser. Vous pouvez également implémenter des politiques IAM basées sur des balises pour contrôler l'accès aux expériences.

Pour étiqueter une expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Experiments.
3. Sélectionnez l'expérience et choisissez Actions, Gérer les balises.
4. Pour ajouter une nouvelle balise, choisissez Ajouter une nouvelle balise, puis spécifiez une clé et une valeur.

   Pour supprimer un tag, choisissez Supprimer pour le tag.
5. Choisissez Enregistrer.

Pour étiqueter une expérience à l'aide de la CLI

Utilisez la commande tag-resource.

Arrêt d'une expérience

Vous pouvez arrêter une expérience en cours à tout moment. Lorsque vous arrêtez un test, toutes les actions de publication qui n'ont pas été effectuées pour une action sont terminées avant l'arrêt du test. Vous ne pouvez pas reprendre une expérience interrompue.

Pour arrêter une expérience à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Experiments.
3. Sélectionnez l'expérience, puis cliquez sur Arrêter l'expérience.
4. Dans la boîte de dialogue de confirmation, choisissez Arrêter l'expérience.

Pour arrêter une expérience à l'aide de la CLI

Utilisez la commande stop-experiment.

Lister les cibles résolues

Vous pouvez consulter les informations relatives aux cibles résolues pour une expérience une fois la résolution cible terminée.

Pour afficher les cibles résolues à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Experiments.
3. Sélectionnez l'expérience, puis sélectionnez Rapport.
4. Consultez les informations sur les cibles résolues sous Ressources.

Pour afficher les cibles résolues à l'aide de la CLI

Utilisez la commande list-experiment-resolved-targets.

Tutoriels pour le service d'injection de AWS défauts

Les didacticiels suivants vous montrent comment créer et exécuter des expériences à l'aide du service d'injection de AWS défauts (AWS FIS).

Didacticiels

- Tutoriel : arrêt et démarrage de l'instance de test à l'aide AWS de FIS
- Tutoriel : Exécuter le stress du processeur sur une instance à l'aide de AWS FIS
- Tutoriel : tester les interruptions d'une instance Spot à l'aide AWS de FIS
- Tutoriel : Simuler un événement de connectivité
- Tutoriel : planifier une expérience récurrente

Tutoriel : arrêt et démarrage de l'instance de test à l'aide AWS de FIS

Vous pouvez utiliser le service d'injection de AWS défauts (AWS FIS) pour tester la façon dont vos applications gèrent l'arrêt et le démarrage des instances. Utilisez ce didacticiel pour créer un modèle d'expérience qui utilise l'`aws:ec2:stop-instances`action AWS FIS pour arrêter une instance, puis une seconde instance.

Prérequis

Pour terminer ce didacticiel, assurez-vous de suivre les étapes suivantes :

- Lancez deux EC2 instances de test dans votre compte. Après avoir lancé vos instances, notez IDs les deux instances.
- Créez un rôle IAM qui permet au service AWS FIS d'effectuer l'`aws:ec2:stop-instances`action en votre nom. Pour de plus amples informations, veuillez consulter Rôles IAM pour les expériences AWS FIS.
- Assurez-vous d'avoir accès au AWS FIS. Pour plus d'informations, consultez les exemples de politiques AWS FIS.

Étape 1 : Création d'un modèle d'expérience

Créez le modèle d'expérience à l'aide de la console AWS FIS. Dans le modèle, vous spécifiez deux actions qui s'exécuteront de manière séquentielle pendant trois minutes chacune. La première action arrête l'une des instances de test, que le AWS FIS choisit de manière aléatoire. La deuxième action arrête les deux instances de test.

Pour créer un modèle d'expérience

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Choisissez Créer un modèle d'expérience.
4. Pour l'étape 1, Spécifier les détails du modèle, procédez comme suit :
   a. Dans Description et nom, entrez une description du modèle, telle que`Amazon S3 Network Disrupt Connectivity`.
   b. Choisissez Suivant, puis passez à l'étape 2, Spécifier les actions et les cibles.
5. Pour Actions, procédez comme suit :
   a. Choisissez Add action.
   b. Entrez le nom de l'action. Par exemple, saisissez **`stopOneInstance`**.
   c. Pour Type d'action, choisissez aws:ec2:stop-instances.
   d. Pour Target, conservez la cible créée par AWS le FIS pour vous.
   e. Pour Paramètres d'action, Démarrer les instances après la durée, spécifiez 3 minutes (PT3M).
   f. Choisissez Save (Enregistrer).
6. Pour Targets (Cibles), procédez comme suit :
   a. Choisissez Modifier pour la cible que AWS FIS a automatiquement créée pour vous à l'étape précédente.
   b. Remplacez le nom par défaut par un nom plus descriptif. Par exemple, saisissez **`oneRandomInstance`**.
   c. Vérifiez que le type de ressource est aws:ec2:instance.
   d. Pour Méthode cible, choisissez Resource IDs, puis choisissez l' IDs une des deux instances de test.

e. Pour le mode de sélection, choisissez Count. Dans le champ Nombre de ressources, entrez**1**.

f. Choisissez Save (Enregistrer).

7. Choisissez Ajouter une cible et procédez comme suit :

a. Entrez le nom de la cible. Par exemple, saisissez **`bothInstances`**.

b. Pour Type de ressource, choisissez aws:ec2:instance.

c. Pour Méthode cible, choisissez Resource IDs, puis choisissez l' IDs une des deux instances de test.

d. Pour le mode de sélection, choisissez Tout.

e. Choisissez Save (Enregistrer).

8. Dans la section Actions, choisissez Ajouter une action. Procédez comme suit :

a. Dans Nom, entrez le nom de l'action. Par exemple, saisissez **`stopBothInstances`**.

b. Pour Type d'action, choisissez aws:ec2:stop-instances.

c. Pour Commencer après, choisissez la première action que vous avez ajoutée (**`stopOneInstance`**).

d. Pour Target, choisissez la deuxième cible que vous avez ajoutée (**`bothInstances`**).

e. Pour Paramètres d'action, Démarrer les instances après la durée, spécifiez 3 minutes (PT3M).

f. Choisissez Save (Enregistrer).

9. Choisissez Suivant pour passer à l'étape 3, Configurer l'accès au service.

10. Pour l'accès aux services, choisissez Utiliser un rôle IAM existant, puis choisissez le rôle IAM que vous avez créé, comme décrit dans les conditions préalables de ce didacticiel. Si votre rôle n'est pas affiché, vérifiez qu'il possède la relation de confiance requise. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.

11. Choisissez Suivant pour passer à l'étape 4, Configurer les paramètres facultatifs.

12. (Facultatif) Pour les balises, choisissez Ajouter une nouvelle balise et spécifiez une clé de balise et une valeur de balise. Les balises que vous ajoutez sont appliquées à votre modèle d'expérience, et non aux expériences exécutées à l'aide du modèle.

13. Choisissez Suivant pour passer à l'étape 5, Réviser et créer.

14. Passez en revue le modèle et choisissez Créer un modèle d'expérience. Lorsque vous êtes invité à confirmer, entrez`create`, puis choisissez Créer un modèle d'expérience.

(Facultatif) Pour afficher le modèle d'expérience JSON

Cliquez sur l'onglet Export (Exporter). Voici un exemple du JSON créé par la procédure de console précédente.

```
{
    "description": "Test instance stop and start",
    "targets": {
        "bothInstances": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": [
                "arn:aws:ec2:region:123456789012:instance/instance_id_1",
                "arn:aws:ec2:region:123456789012:instance/instance_id_2"
            ],
            "selectionMode": "ALL"
        },
        "oneRandomInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": [
                "arn:aws:ec2:region:123456789012:instance/instance_id_1",
                "arn:aws:ec2:region:123456789012:instance/instance_id_2"
            ],
            "selectionMode": "COUNT(1)"
        }
    },
    "actions": {
        "stopBothInstances": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "startInstancesAfterDuration": "PT3M"
            },
            "targets": {
                "Instances": "bothInstances"
            },
            "startAfter": [
                "stopOneInstance"
            ]
        },
        "stopOneInstance": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "startInstancesAfterDuration": "PT3M"
            },
            "targets": {
```

```
                "Instances": "oneRandomInstance"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none"
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/AllowFISEC2Actions",
    "tags": {}
}
```

Étape 2 : démarrer l'expérience

Lorsque vous avez fini de créer votre modèle de test, vous pouvez l'utiliser pour démarrer un test.

Pour démarrer une expérience

1. Vous devriez être sur la page de détails du modèle d'expérience que vous venez de créer. Sinon, choisissez Modèles d'expérience, puis sélectionnez l'ID du modèle d'expérience pour ouvrir la page de détails.
2. Sélectionnez Start experiment (Démarrer une expérience).
3. (Facultatif) Pour ajouter une balise à votre expérience, choisissez Ajouter une nouvelle balise et entrez une clé de balise et une valeur de balise.
4. Sélectionnez Start experiment (Démarrer une expérience). Lorsque vous êtes invité à confirmer, entrez **`start`** et choisissez Démarrer l'expérience.

Étape 3 : suivre la progression de l'expérience

Vous pouvez suivre la progression d'une expérience en cours jusqu'à ce qu'elle soit terminée, arrêtée ou échouée.

Pour suivre la progression d'une expérience

1. Vous devriez être sur la page de détails de l'expérience que vous venez de commencer. Sinon, choisissez Expériences, puis sélectionnez l'ID de l'expérience pour ouvrir la page de détails.
2. Pour voir l'état de l'expérience, cochez la case État dans le volet Détails. Pour plus d'informations, consultez la section États de l'expérience.

3. Lorsque l'état de l'expérience est en cours d'exécution, passez à l'étape suivante.

Étape 4 : vérifier le résultat de l'expérience

Vous pouvez vérifier que les instances ont été arrêtées et démarrées par l'expérience comme prévu.

Pour vérifier le résultat de l'expérience

1. Ouvrez la EC2 console Amazon https://console.aws.amazon.com/ec2/dans un nouvel onglet ou une nouvelle fenêtre de navigateur. Cela vous permet de continuer à suivre la progression de l'expérience dans la console AWS FIS tout en visualisant le résultat de l'expérience dans la EC2 console Amazon.
2. Dans le panneau de navigation, choisissez Instances.
3. Lorsque l'état de la première action passe de En attente à Exécution (console AWS FIS), l'état de l'une des instances cibles passe de Exécution à Arrêté (EC2 console Amazon).
4. Au bout de trois minutes, l'état de la première action passe à Terminé, l'état de la deuxième action passe à Exécuter et l'état de l'autre instance cible passe à Arrêté.
5. Au bout de trois minutes, l'état de la deuxième action passe à Terminé, l'état des instances cibles passe à Exécution et l'état de l'expérience passe à Terminé.

Étape 5 : nettoyer

Si vous n'avez plus besoin des EC2 instances de test que vous avez créées pour cette expérience, vous pouvez y mettre fin.

Pour résilier les instances

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation, choisissez Instances.
3. Sélectionnez les deux instances de test, choisissez Instance state) (État de l'instance, Terminate instance (Résilier l'instance).
4. Lorsque vous êtes invité à confirmer, choisissez Terminate (Mettre fin).

Si vous n'avez plus besoin du modèle d'expérience, vous pouvez le supprimer.

Pour supprimer un modèle d'expérience à l'aide de la AWS console FIS

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Supprimer le modèle d'expérience.
4. Lorsque vous êtes invité à confirmer, entrez **`delete`** puis choisissez Supprimer le modèle d'expérience.

Tutoriel : Exécuter le stress du processeur sur une instance à l'aide de AWS FIS

Vous pouvez utiliser le service d'injection de AWS défauts (AWS FIS) pour tester la façon dont vos applications gèrent le stress du processeur. Utilisez ce didacticiel pour créer un modèle d'expérience qui utilise AWS FIS pour exécuter un document SSM préconfiguré qui gère le stress du processeur sur une instance. Le didacticiel utilise une condition d'arrêt pour arrêter l'expérience lorsque l'utilisation du processeur de l'instance dépasse un seuil configuré.

Pour de plus amples informations, veuillez consulter the section called “Documents AWS FIS SSM préconfigurés”.

Prérequis

Avant de pouvoir utiliser AWS FIS pour gérer le stress du processeur, vous devez remplir les conditions préalables suivantes.

Créer un rôle IAM

Créez un rôle et associez une politique qui permet à AWS FIS d'utiliser l'`aws:ssm:send-command`action en votre nom. Pour de plus amples informations, veuillez consulter Rôles IAM pour les expériences AWS FIS.

Vérifier l'accès au AWS FIS

Assurez-vous d'avoir accès au AWS FIS. Pour plus d'informations, consultez les exemples de politiques AWS FIS.

Préparer une EC2 instance de test

- Lancez une EC2 instance à l'aide d'Amazon Linux 2 ou Ubuntu, comme l'exigent les documents SSM préconfigurés.
- L'instance doit être gérée par SSM. Pour vérifier que l'instance est gérée par SSM, ouvrez la console Fleet Manager. Si l'instance n'est pas gérée par SSM, vérifiez que l'agent SSM est installé et qu'un rôle IAM est attaché à l'instance conformément à la politique Amazon. SSMManaged InstanceCore Pour vérifier l'agent SSM installé, connectez-vous à votre instance et exécutez la commande suivante.

  Amazon Linux 2

  ```
  yum info amazon-ssm-agent
  ```

  Ubuntu

  ```
  apt list amazon-ssm-agent
  ```

- Activez la surveillance détaillée de l'instance. Cela fournit des données par périodes d'une minute, moyennant des frais supplémentaires. Sélectionnez l'instance et choisissez Actions, Surveiller et dépanner, Gérer la surveillance détaillée.

Étape 1 : créer une CloudWatch alarme pour une condition d'arrêt

Configurez une CloudWatch alarme afin de pouvoir arrêter l'expérience si l'utilisation du processeur dépasse le seuil que vous spécifiez. La procédure suivante définit le seuil à 50 % d'utilisation du processeur pour l'instance cible. Pour de plus amples informations, veuillez consulter Conditions d'arrêt.

Pour créer une alarme indiquant lorsque l'utilisation du processeur dépasse un seuil

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation, choisissez Instances.
3. Sélectionnez l'instance cible et choisissez Actions, Surveiller et dépanner, Gérer les CloudWatch alarmes.
4. Pour les notifications d'alarme, utilisez le bouton pour désactiver les notifications Amazon SNS.
5. Pour les seuils d'alarme, utilisez les paramètres suivants :

- Regrouper les échantillons par : Maximum
- Type de données à échantillonner : utilisation du processeur
- Pourcentage : **`50`**
- Période : **`1 Minute`**

6. Lorsque vous avez terminé de configurer l'alarme, choisissez Create.

Étape 2 : Création d'un modèle d'expérience

Créez le modèle d'expérience à l'aide de la console AWS FIS. Dans le modèle, vous spécifiez l'action suivante à exécuter : AWSFIS-Runaws:ssm:send-command/ -CPU-Stress.

Pour créer un modèle d'expérience

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Choisissez Créer un modèle d'expérience.
4. Pour l'étape 1, Spécifier les détails du modèle, procédez comme suit :
   a. Dans Description et nom, entrez une description pour le modèle.
   b. Choisissez Suivant, puis passez à l'étape 2, Spécifier les actions et les cibles.
5. Pour Actions, procédez comme suit :
   a. Choisissez Add action.
   b. Entrez le nom de l'action. Par exemple, saisissez **`runCpuStress`**.
   c. Pour Type d'action, choisissez AWSFIS-Runaws:ssm:send-command/ -CPU-Stress. Cela ajoute automatiquement l'ARN du document SSM à l'ARN du document.
   d. Pour Target, conservez la cible créée par AWS le FIS pour vous.
   e. Pour Paramètres d'action, Paramètres du document, entrez ce qui suit :

      ```
      {"DurationSeconds":"120"}
      ```

   f. Pour Paramètres d'action, Durée, spécifiez 5 minutes (PT5M).
   g. Choisissez Save (Enregistrer).
6. Pour Targets (Cibles), procédez comme suit :

a. Choisissez Modifier pour la cible que AWS FIS a automatiquement créée pour vous à l'étape précédente.

b. Remplacez le nom par défaut par un nom plus descriptif. Par exemple, saisissez **`testInstance`**.

c. Vérifiez que le type de ressource est aws:ec2:instance.

d. Pour Méthode cible, choisissez Resource IDs, puis choisissez l'ID de l'instance de test.

e. Pour le mode de sélection, choisissez Tout.

f. Choisissez Save (Enregistrer).

7. Choisissez Suivant pour passer à l'étape 3, Configurer l'accès au service.
8. Pour l'accès aux services, choisissez Utiliser un rôle IAM existant, puis choisissez le rôle IAM que vous avez créé, comme décrit dans les conditions préalables de ce didacticiel. Si votre rôle n'est pas affiché, vérifiez qu'il possède la relation de confiance requise. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.
9. Choisissez Suivant pour passer à l'étape 4, Configurer les paramètres facultatifs.
10. Pour les conditions d'arrêt, sélectionnez l' CloudWatch alarme que vous avez créée à l'étape 1.
11. (Facultatif) Pour les balises, choisissez Ajouter une nouvelle balise et spécifiez une clé de balise et une valeur de balise. Les balises que vous ajoutez sont appliquées à votre modèle d'expérience, et non aux expériences exécutées à l'aide du modèle.
12. Choisissez Suivant pour passer à l'étape 5, Réviser et créer.
13. Passez en revue le modèle et choisissez Créer un modèle d'expérience. Lorsque vous êtes invité à confirmer, entrez`create`, puis choisissez Créer un modèle d'expérience.

(Facultatif) Pour afficher le modèle d'expérience JSON

Cliquez sur l'onglet Export (Exporter). Voici un exemple du JSON créé par la procédure de console précédente.

```
{
    "description": "Test CPU stress predefined SSM document",
    "targets": {
        "testInstance": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": [
                "arn:aws:ec2:region:123456789012:instance/instance_id"
            ],
```

```
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "runCpuStress": {
            "actionId": "aws:ssm:send-command",
            "parameters": {
                "documentArn": "arn:aws:ssm:region::document/AWSFIS-Run-CPU-Stress",
                "documentParameters": "{\"DurationSeconds\":\"120\"}",
                "duration": "PT5M"
            },
            "targets": {
                "Instances": "testInstance"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:region:123456789012:alarm:awsec2-instance_id-
GreaterThanOrEqualToThreshold-CPUUtilization"
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/AllowFISSSMActions",
    "tags": {}
}
```

Étape 3 : démarrer l'expérience

Lorsque vous avez fini de créer votre modèle de test, vous pouvez l'utiliser pour démarrer un test.

Pour démarrer une expérience

1. Vous devriez être sur la page de détails du modèle d'expérience que vous venez de créer. Sinon, choisissez Modèles d'expérience, puis sélectionnez l'ID du modèle d'expérience pour ouvrir la page de détails.
2. Sélectionnez Start experiment (Démarrer une expérience).
3. (Facultatif) Pour ajouter une balise à votre expérience, choisissez Ajouter une nouvelle balise et entrez une clé de balise et une valeur de balise.
4. Sélectionnez Start experiment (Démarrer une expérience). À l'invite de confirmation, saisissez **`start`**. Sélectionnez Start experiment (Démarrer une expérience).

Étape 4 : suivre la progression de l'expérience

Vous pouvez suivre la progression d'une expérience en cours jusqu'à ce qu'elle se termine, s'arrête ou échoue.

Pour suivre la progression d'une expérience

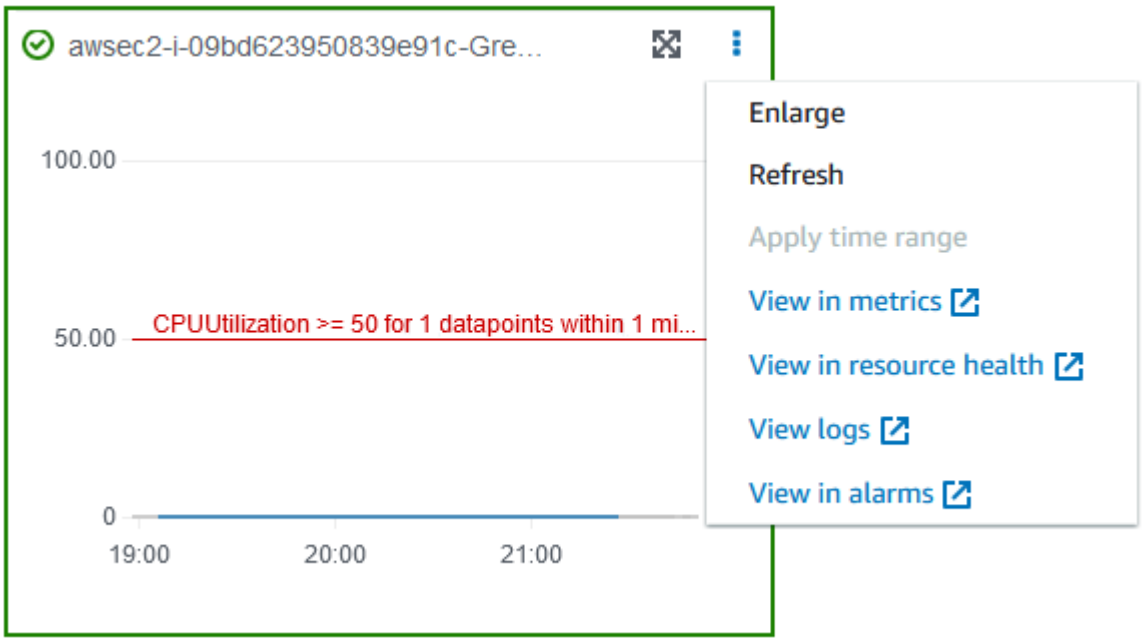
1. Vous devriez être sur la page de détails de l'expérience que vous venez de commencer. Sinon, choisissez Experiments, puis sélectionnez l'ID de l'expérience pour ouvrir la page de détails de l'expérience.
2. Pour voir l'état de l'expérience, cochez la case État dans le volet Détails. Pour plus d'informations, consultez la section États de l'expérience.
3. Lorsque l'état de l'expérience est en cours d'exécution, passez à l'étape suivante.

Étape 5 : Vérifiez les résultats de l'expérience

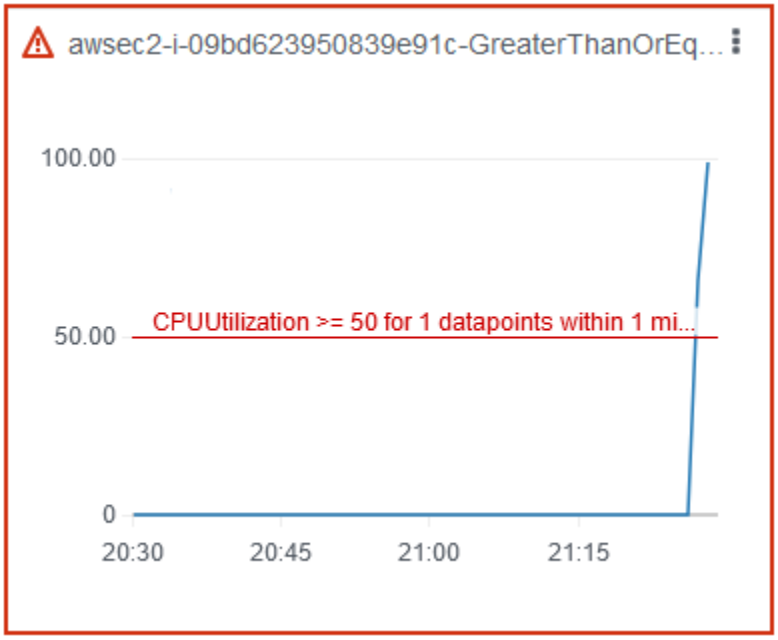
Vous pouvez surveiller l'utilisation du processeur de votre instance pendant que le test est en cours d'exécution. Lorsque l'utilisation du processeur atteint le seuil, l'alarme est déclenchée et l'expérience est interrompue par la condition d'arrêt.

Pour vérifier les résultats de l'expérience

1. Cliquez sur l'onglet Conditions d'arrêt. La bordure verte et l'icône en forme de coche verte indiquent que l'état initial de l'alarme est0K. La ligne rouge indique le seuil d'alarme. Si vous préférez un graphique plus détaillé, choisissez Agrandir dans le menu du widget.



2. Lorsque l'utilisation du processeur dépasse le seuil, la bordure rouge et l'icône du point d'exclamation rouge dans l'onglet Conditions d'arrêt indiquent que l'état de l'alarme est passé àALARM. Dans le volet Détails, l'état de l'expérience est Arrêté. Si vous sélectionnez l'état, le message affiché est « Expérience interrompue par une condition d'arrêt ».



3. Lorsque l'utilisation du processeur diminue en dessous du seuil, la bordure verte et l'icône en forme de coche verte indiquent que l'état de l'alarme est passé à0K.

4. (Facultatif) Choisissez Afficher dans les alarmes dans le menu du widget. Cela ouvre la page des détails de l'alarme dans la CloudWatch console, où vous pouvez obtenir plus de détails sur l'alarme ou modifier les paramètres de l'alarme.

Étape 6 : Nettoyer

Si vous n'avez plus besoin de l' EC2 instance de test que vous avez créée pour cette expérience, vous pouvez y mettre fin.

Pour résilier l'instance

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation, choisissez Instances.
3. Sélectionnez les instances de test, puis choisissez État de l'instance, Terminer l'instance.
4. Lorsque vous êtes invité à confirmer, choisissez Terminate (Mettre fin).

Si vous n'avez plus besoin du modèle d'expérience, vous pouvez le supprimer.

Pour supprimer un modèle d'expérience à l'aide de la AWS console FIS

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Supprimer le modèle d'expérience.
4. Lorsque vous êtes invité à confirmer, entrez **`delete`** puis choisissez Supprimer le modèle d'expérience.

Tutoriel : tester les interruptions d'une instance Spot à l'aide AWS de FIS

Les instances Spot utilisent la EC2 capacité inutilisée disponible, pour bénéficier d'une réduction allant jusqu'à 90 % par rapport à la tarification à la demande. Amazon EC2 peut toutefois interrompre vos instances Spot lorsqu'il a besoin de récupérer leur capacité. Lorsque vous utilisez des instances Spot, vous devez être prêt à faire face à d'éventuelles interruptions. Pour plus d'informations, consultez la section Interruptions des instances Spot dans le guide de EC2 l'utilisateur Amazon.

Vous pouvez utiliser le service d'injection de AWS défauts (AWS FIS) pour tester la manière dont vos applications gèrent une interruption d'instance Spot. Utilisez ce didacticiel pour créer un modèle d'expérience qui utilise l'`aws:ec2:send-spot-instance-interruptions`action AWS FIS pour interrompre l'une de vos instances Spot.

Sinon, pour lancer l'expérience à l'aide de la EC2 console Amazon, consultez la section Initiate a Spot Instance interruption dans le guide de EC2 l'utilisateur Amazon.

Prérequis

Avant de pouvoir utiliser AWS FIS pour interrompre une instance Spot, vous devez remplir les conditions préalables suivantes.

1. Créer un rôle IAM

Créez un rôle et associez une politique qui permet à AWS FIS d'effectuer l'`aws:ec2:send-spot-instance-interruptions`action en votre nom. Pour de plus amples informations, veuillez consulter Rôles IAM pour les expériences AWS FIS.

2. Vérifier l'accès au AWS FIS

Assurez-vous d'avoir accès au AWS FIS. Pour plus d'informations, consultez les exemples de politiques AWS FIS.

3. (Facultatif) Créez une demande d'instance Spot

Si vous souhaitez utiliser une nouvelle instance Spot pour cette expérience, utilisez la commande run-instances pour demander une instance Spot. Par défaut, les instances Spot interrompues sont résiliées. Si vous définissez le comportement d'interruption sur`stop`, vous devez également définir le type sur`persistent`. Pour ce didacticiel, ne définissez pas le comportement d'interruption sur`hibernate`, car le processus d'hibernation commence immédiatement.

```
aws ec2 run-instances \
    --image-id ami-0ab193018fEXAMPLE \
    --instance-type "t2.micro" \
    --count 1 \
    --subnet-id subnet-1234567890abcdef0 \
    --security-group-ids sg-111222333444aaab \
    --instance-market-options file://spot-options.json \
    --query Instances[*].InstanceId
```

Voici un exemple du fichier `spot-options.json`.

```
{
    "MarketType": "spot",
    "SpotOptions": {
        "SpotInstanceType": "persistent",
        "InstanceInterruptionBehavior": "stop"
    }
}
```

L'`--query`option de l'exemple de commande fait en sorte que la commande renvoie uniquement l'ID d'instance de l'instance Spot. Voici un exemple de sortie.

```
[
    "i-0abcdef1234567890"
]
```

4. Ajoutez une balise afin que AWS FIS puisse identifier l'instance Spot cible

Utilisez la commande create-tags pour ajouter le tag Name=interruptMe vers votre instance Spot cible.

```
aws ec2 create-tags \
    --resources i-0abcdef1234567890 \
    --tags Key=Name,Value=interruptMe
```

Étape 1 : Création d'un modèle d'expérience

Créez le modèle d'expérience à l'aide de la console AWS FIS. Dans le modèle, vous spécifiez l'action qui sera exécutée. L'action interrompt l'instance Spot avec la balise spécifiée. Si le tag est associé à plusieurs instances Spot, le AWS FIS choisit l'une d'entre elles au hasard.

Pour créer un modèle d'expérience

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Choisissez Créer un modèle d'expérience.
4. Pour l'étape 1, Spécifier les détails du modèle, procédez comme suit :
    a. Dans Description et nom, entrez une description et un nom pour le modèle.
    b. Choisissez Suivant, puis passez à l'étape 2, Spécifier les actions et les cibles.

5. Pour Actions, procédez comme suit :

   a. Choisissez Add action.

   b. Entrez le nom de l'action. Par exemple, saisissez **`interruptSpotInstance`**.

   c. Pour Type d'action, choisissez aws:ec2 :. send-spot-instance-interruptions

   d. Pour Target, conservez la cible créée par AWS le FIS pour vous.

   e. Pour Paramètres d'action, Durée avant interruption, spécifiez 2 minutes (PT2M).

   f. Choisissez Save (Enregistrer).

6. Pour Targets (Cibles), procédez comme suit :

   a. Choisissez Modifier pour la cible que AWS FIS a automatiquement créée pour vous à l'étape précédente.

   b. Remplacez le nom par défaut par un nom plus descriptif. Par exemple, saisissez **`oneSpotInstance`**.

   c. Vérifiez que le type de ressource est aws:ec2:spot-instance.

   d. Pour Méthode cible, sélectionnez Balises de ressources, filtres et paramètres.

   e. Pour les balises de ressource, choisissez Ajouter une nouvelle balise, puis entrez la clé et la valeur de la balise. Utilisez la balise que vous avez ajoutée à l'instance Spot pour l'interrompre, comme décrit dans les conditions préalables de ce didacticiel.

   f. Pour les filtres de ressources, choisissez Ajouter un nouveau filtre et entrez **`State.Name`** le chemin et **`running`** la valeur.

   g. Pour le mode de sélection, choisissez Count. Dans le champ Nombre de ressources, entrez**`1`**.

   h. Choisissez Save (Enregistrer).

7. Choisissez Suivant pour passer à l'étape 3, Configurer l'accès au service.

8. Pour l'accès aux services, choisissez Utiliser un rôle IAM existant, puis choisissez le rôle IAM que vous avez créé, comme décrit dans les conditions préalables de ce didacticiel. Si votre rôle n'est pas affiché, vérifiez qu'il possède la relation de confiance requise. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.

9. Choisissez Suivant pour passer à l'étape 4, Configurer les paramètres facultatifs.

10. (Facultatif) Pour les balises, choisissez Ajouter une nouvelle balise et spécifiez une clé de balise et une valeur de balise. Les balises que vous ajoutez sont appliquées à votre modèle d'expérience, et non aux expériences exécutées à l'aide du modèle.

11. Choisissez Suivant pour passer à l'étape 5, Réviser et créer.
12. Passez en revue le modèle et choisissez Créer un modèle d'expérience. Lorsque vous êtes invité à confirmer, entrez`create`, puis choisissez Créer un modèle d'expérience.

(Facultatif) Pour afficher le modèle d'expérience JSON

Cliquez sur l'onglet Export (Exporter). Voici un exemple du JSON créé par la procédure de console précédente.

```
{
    "description": "Test Spot Instance interruptions",
    "targets": {
        "oneSpotInstance": {
            "resourceType": "aws:ec2:spot-instance",
            "resourceTags": {
                "Name": "interruptMe"
            },
            "filters": [
                {
                    "path": "State.Name",
                    "values": [
                        "running"
                    ]
                }
            ],
            "selectionMode": "COUNT(1)"
        }
    },
    "actions": {
        "interruptSpotInstance": {
            "actionId": "aws:ec2:send-spot-instance-interruptions",
            "parameters": {
                "durationBeforeInterruption": "PT2M"
            },
            "targets": {
                "SpotInstances": "oneSpotInstance"
            }
        }
    },
    "stopConditions": [
        {
            "source": "none"
```

```
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/AllowFISSpotInterruptionActions",
    "tags": {
        "Name": "my-template"
    }
}
```

Étape 2 : démarrer l'expérience

Lorsque vous avez fini de créer votre modèle de test, vous pouvez l'utiliser pour démarrer un test.

Pour démarrer une expérience

1. Vous devriez être sur la page de détails du modèle d'expérience que vous venez de créer. Sinon, choisissez Modèles d'expérience, puis sélectionnez l'ID du modèle d'expérience pour ouvrir la page de détails.
2. Sélectionnez Start experiment (Démarrer une expérience).
3. (Facultatif) Pour ajouter une balise à votre expérience, choisissez Ajouter une nouvelle balise et entrez une clé de balise et une valeur de balise.
4. Sélectionnez Start experiment (Démarrer une expérience). Lorsque vous êtes invité à confirmer, entrez **`start`** et choisissez Démarrer l'expérience.

Étape 3 : suivre la progression de l'expérience

Vous pouvez suivre la progression d'une expérience en cours jusqu'à ce qu'elle soit terminée, arrêtée ou échouée.

Pour suivre la progression d'une expérience

1. Vous devriez être sur la page de détails de l'expérience que vous venez de commencer. Sinon, choisissez Expériences, puis sélectionnez l'ID de l'expérience pour ouvrir la page de détails.
2. Pour voir l'état de l'expérience, cochez la case État dans le volet Détails. Pour plus d'informations, consultez la section États de l'expérience.
3. Lorsque l'état de l'expérience est en cours d'exécution, passez à l'étape suivante.

Étape 4 : vérifier le résultat de l'expérience

Lorsque l'action de cette expérience est terminée, les événements suivants se produisent :

- L'instance Spot cible reçoit une recommandation de rééquilibrage d'instance.
- Un avis d'interruption de l'instance Spot est émis deux minutes avant qu'Amazon ne EC2 mette fin ou arrête votre instance.
- Au bout de deux minutes, l'instance Spot est résiliée ou arrêtée.
- Une instance Spot arrêtée par AWS FIS reste arrêtée jusqu'à ce que vous la redémarriez.

Pour vérifier que l'instance a été interrompue par l'expérience

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Depuis le panneau de navigation, ouvrez Spot Requests (Demandes Spot) et Instances dans différents onglets ou fenêtres de navigateur.
3. Pour Spot Requests (Demandes Spot), sélectionnez la demande d'instance Spot. L'état initial est `fulfilled`. Une fois l'expérience terminée, le statut change comme suit :
   - `terminate`- Le statut passe à`instance-terminated-by-experiment`.
   - `stop`- Le statut passe à `marked-for-stop-by-experiment` et ensuite`instance-stopped-by-experiment`.
4. Pour Instances, sélectionnez l'instance Spot. L'état initial est `Running`. Deux minutes après avoir reçu l'avis d'interruption de l'instance Spot, le statut change comme suit :
   - `stop`- Le statut passe à `Stopping` et ensuite`Stopped`.
   - `terminate`- Le statut passe à `Shutting-down` et ensuite`Terminated`.

Étape 5 : nettoyer

Si vous avez créé l'instance Spot de test pour cette expérience avec un comportement d'interruption de `stop` et que vous n'en avez plus besoin, vous pouvez annuler la demande d'instance Spot et mettre fin à l'instance Spot.

Pour annuler la demande et mettre fin à l'instance à l'aide du AWS CLI

1. Utilisez la cancel-spot-instance-requestscommande pour annuler la demande d'instance Spot.

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-ksie869j
```

2. Utilisez la commande terminate-instances pour mettre fin à l'instance.

```
aws ec2 terminate-instances --instance-ids i-0abcdef1234567890
```

Si vous n'avez plus besoin du modèle d'expérience, vous pouvez le supprimer.

Pour supprimer un modèle d'expérience à l'aide de la AWS console FIS

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Supprimer le modèle d'expérience.
4. Lorsque vous êtes invité à confirmer, entrez **`delete`** puis choisissez Supprimer le modèle d'expérience.

Tutoriel : Simuler un événement de connectivité

Vous pouvez utiliser le service d'injection de AWS défauts (AWS FIS) pour simuler divers événements de connectivité. AWS FIS simule les événements de connectivité en bloquant les connexions réseau de l'une des manières suivantes :

- `all`— Refuse tout le trafic entrant et sortant du sous-réseau. Notez que cette option autorise le trafic intra-sous-réseau, y compris le trafic à destination et en provenance des interfaces réseau du sous-réseau.
- `availability-zone`— Refuse le trafic intra-VPC à destination et en provenance de sous-réseaux dans d'autres zones de disponibilité.
- `dynamodb`— Refuse le trafic à destination et en provenance du point de terminaison régional pour DynamoDB dans la région actuelle.
- `prefix-list`— Refuse le trafic à destination et en provenance de la liste de préfixes spécifiée.
- `s3`— Refuse le trafic à destination et en provenance du point de terminaison régional pour Amazon S3 dans la région actuelle.
- `vpc`— Empêche le trafic entrant et sortant du VPC.

Utilisez ce didacticiel pour créer un modèle d'expérience qui utilise l'`aws:network:disrupt-connectivity`action AWS FIS pour introduire une perte de connectivité avec Amazon S3 dans un sous-réseau cible.

Rubriques

- Prérequis
- Étape 1 : Création d'un AWS modèle d'expérience FIS
- Étape 2 : envoyer un ping à un point de terminaison Amazon S3
- Étape 3 : Commencez votre AWS expérience FIS
- Étape 4 : Suivez la progression AWS de votre expérience FIS
- Étape 5 : vérifier l'interruption du réseau Amazon S3
- Étape 5 : nettoyer

Prérequis

Avant de commencer ce didacticiel, vous avez besoin d'un rôle doté des autorisations appropriées dans votre Compte AWS EC2 instance Amazon et d'une instance de test :

Un rôle doté d'autorisations dans votre Compte AWS

Créez un rôle et associez une politique qui permet à AWS FIS d'effectuer l'`aws:network:disrupt-connectivity`action en votre nom.

Votre rôle IAM nécessite la politique suivante :

- AWSFaultInjectionSimulatorNetworkAccess— Accorde l'autorisation de service AWS FIS sur le EC2 réseau Amazon et les autres services requis pour effectuer des actions AWS FIS liées à l'infrastructure réseau.

> **Note**
>
> Pour des raisons de simplicité, ce didacticiel utilise une politique AWS gérée. Pour une utilisation en production, nous vous recommandons de n'accorder que les autorisations minimales nécessaires à votre cas d'utilisation.
> Pour plus d'informations sur la création d'un rôle IAM, voir Rôles IAM pour les expériences AWS FIS (AWS CLI) ou Création d'un rôle IAM (console) dans le Guide de l'utilisateur IAM.

Une EC2 instance Amazon de test

Lancez et connectez-vous à une EC2 instance Amazon de test. Vous pouvez utiliser le didacticiel suivant pour lancer et vous connecter à une EC2 instance Amazon : Tutoriel : Commencez avec les instances Amazon EC2 Linux dans le guide de EC2 l'utilisateur Amazon.

Étape 1 : Création d'un AWS modèle d'expérience FIS

Créez le modèle d'expérience à l'aide du AWS FIS AWS Management Console. Un modèle AWS FIS est composé d'actions, de cibles, de conditions d'arrêt et d'un rôle d'expérience. Pour plus d'informations sur le fonctionnement des modèles, voir Modèles d'expériences pour AWS FIS.

Avant de commencer, assurez-vous que les éléments suivants sont prêts :

- Un rôle IAM doté des autorisations appropriées.
- Une EC2 instance Amazon.
- L'ID de sous-réseau de votre EC2 instance Amazon.

Pour créer un modèle d'expérience

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation de gauche, sélectionnez Modèles d'expériences.
3. Choisissez Créer un modèle d'expérience.
4. Pour l'étape 1, Spécifier les détails du modèle, procédez comme suit :
    a. Dans Description et nom, entrez une description du modèle, telle que`Amazon S3 Network Disrupt Connectivity`.
    b. Choisissez Suivant, puis passez à l'étape 2, Spécifier les actions et les cibles.
5. Sous Actions, sélectionnez Ajouter une action.
    a. Pour le nom, entrez`disruptConnectivity`.
    b. Pour Type d'action, sélectionnez aws:network:disrupt-connectivity.
    c. Sous Paramètres d'action, définissez la durée sur`2 minutes`.
    d. Sous Champ d'application, sélectionnez s3.
    e. En haut de la page, choisissez Enregistrer.
6. Sous Cibles, vous devriez voir la cible qui a été créée automatiquement. Choisissez Modifier.

a. Vérifiez que le type de ressource est`aws:ec2:subnet`.

b. Sous Méthode cible, sélectionnez Ressource IDs, puis choisissez le sous-réseau que vous avez utilisé lors de la création de votre EC2 instance Amazon dans les étapes Prérequis.

c. Vérifiez que le mode de sélection est défini sur Tous.

d. Choisissez Save (Enregistrer).

7. Choisissez Suivant pour passer à l'étape 3, Configurer l'accès au service.
8. Sous Accès aux services, sélectionnez le rôle IAM que vous avez créé, comme décrit dans les conditions préalables de ce didacticiel. Si votre rôle n'est pas affiché, vérifiez qu'il possède la relation de confiance requise. Pour de plus amples informations, veuillez consulter the section called "Rôle d'expérience".
9. Choisissez Suivant pour passer à l'étape 4, Configurer les paramètres facultatifs.
10. (Facultatif) Dans Conditions d'arrêt, vous pouvez sélectionner une CloudWatch alarme pour arrêter l'expérience si la condition se produit. Pour plus d'informations, consultez la section Conditions d'arrêt pour AWS FIS.
11. (Facultatif) Sous Logs, vous pouvez sélectionner un compartiment Amazon S3 ou envoyer des journaux CloudWatch pour votre expérience.
12. Choisissez Suivant pour passer à l'étape 5, Réviser et créer.
13. Passez en revue le modèle et choisissez Créer un modèle d'expérience. Lorsque vous êtes invité à confirmer, entrez`create`, puis choisissez Créer un modèle d'expérience.

Étape 2 : envoyer un ping à un point de terminaison Amazon S3

Vérifiez que votre EC2 instance Amazon est capable d'atteindre un point de terminaison Amazon S3.

1. Connectez-vous à l' EC2 instance Amazon que vous avez créée dans les étapes des prérequis.

   Pour résoudre les problèmes, consultez la section Résoudre les problèmes liés à la connexion à votre instance dans le guide de EC2 l'utilisateur Amazon.

2. Vérifiez Région AWS où se trouve votre instance. Vous pouvez le faire dans la EC2 console Amazon ou en exécutant la commande suivante.

```
hostname
```

Par exemple, si vous avez lancé une EC2 instance Amazon en`us-west-2`, vous verrez le résultat suivant.

```
[ec2-user@ip-172.16.0.0 ~]$ hostname
ip-172.16.0.0.us-west-2.compute.internal
```

3. Envoyez un ping à un point de terminaison Amazon S3 dans votre Région AWS. Remplacez *`Région AWS`* par votre région.

```
ping -c 1 s3.Région AWS.amazonaws.com
```

Pour la sortie, vous devriez voir un ping réussi avec 0 % de perte de paquets, comme illustré dans l'exemple suivant.

```
PING s3.us-west-2.amazonaws.com (x.x.x.x) 56(84) bytes of data.
64 bytes from s3-us-west-2.amazonaws.com (x.x.x.x: icmp_seq=1 ttl=249 time=1.30 ms

--- s3.us-west-2.amazonaws.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.306/1.306/1.306/0.000 ms
```

Étape 3 : Commencez votre AWS expérience FIS

Lancez un test avec le modèle de test que vous venez de créer.

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation de gauche, sélectionnez Modèles d'expériences.
3. Sélectionnez l'ID du modèle d'expérience que vous avez créé pour ouvrir sa page de détails.
4. Sélectionnez Start experiment (Démarrer une expérience).
5. (Facultatif) Sur la page de confirmation, ajoutez des balises pour votre expérience.
6. Sur la page de confirmation, choisissez Démarrer l'expérience.

Étape 4 : Suivez la progression AWS de votre expérience FIS

Vous pouvez suivre la progression d'une expérience en cours jusqu'à ce qu'elle soit terminée, arrêtée ou échouée.

1. Vous devriez être sur la page de détails de l'expérience que vous venez de commencer. Si ce n'est pas le cas, choisissez Expériences, puis sélectionnez l'ID de l'expérience pour ouvrir sa page de détails.
2. Pour voir l'état de l'expérience, vérifiez l'état dans le volet de détails. Pour plus d'informations, consultez la section États des expériences.
3. Lorsque l'état de l'expérience est en cours d'exécution, passez à l'étape suivante.

Étape 5 : vérifier l'interruption du réseau Amazon S3

Vous pouvez valider la progression de l'expérience en envoyant un ping au point de terminaison Amazon S3.

- Depuis votre EC2 instance Amazon, envoyez un ping au point de terminaison Amazon S3 dans votre Région AWS. Remplacez *Région AWS* par votre région.

  ```
  ping -c 1 s3.Région AWS.amazonaws.com
  ```

  Pour la sortie, vous devriez voir un ping infructueux avec une perte de paquets de 100 %, comme indiqué dans l'exemple suivant.

  ```
  ping -c 1 s3.us-west-2.amazonaws.com
  PING s3.us-west-2.amazonaws.com (x.x.x.x) 56(84) bytes of data.

  --- s3.us-west-2.amazonaws.com ping statistics ---
  1 packets transmitted, 0 received, 100% packet loss, time 0ms
  ```

Étape 5 : nettoyer

Si vous n'avez plus besoin de l' EC2 instance Amazon que vous avez créée pour cette expérience ou du modèle AWS FIS, vous pouvez les supprimer.

Pour supprimer l' EC2 instance Amazon

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation, choisissez Instances.
3. Sélectionnez l'instance de test, choisissez État de l'instance, puis Terminate instance.

4. Lorsque vous êtes invité à confirmer, choisissez Terminate (Mettre fin).

Pour supprimer le modèle d'expérience à l'aide de la AWS console FIS

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Supprimer le modèle d'expérience.
4. Lorsque vous êtes invité à confirmer, entrez`delete`, puis choisissez Supprimer le modèle d'expérience.

Tutoriel : planifier une expérience récurrente

Avec le service d'injection de AWS défauts (AWS FIS), vous pouvez réaliser des expériences d'injection de défauts sur vos charges AWS de travail. Ces expériences s'exécutent sur des modèles contenant une ou plusieurs actions à exécuter sur des cibles spécifiées. Lorsque vous utilisez également Amazon EventBridge, vous pouvez planifier vos expériences sous la forme d'une tâche ponctuelle ou de tâches récurrentes.

Utilisez ce didacticiel pour créer un EventBridge calendrier qui exécute un modèle d'expérience AWS FIS toutes les 5 minutes.

Tâches

- Prérequis
- Étape 1 : Création d'un rôle et d'une politique IAM
- Étape 2 : Création d'un Amazon EventBridge planificateur
- Étape 3 : Vérifiez votre expérience
- Étape 4 : nettoyer

Prérequis

Avant de commencer ce didacticiel, vous devez disposer d'un modèle d'expérience AWS FIS que vous souhaitez exécuter selon un calendrier. Si vous disposez déjà d'un modèle d'expérience de travail, notez l'ID du modèle et Région AWS. Sinon, vous pouvez créer un modèle en suivant les

instructions fournies dans ce didacticielthe section called “Arrêt et démarrage de l'instance de test”, puis en revenant à ce didacticiel.

Étape 1 : Création d'un rôle et d'une politique IAM

Pour créer un rôle et une politique IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de gauche, choisissez Rôles, puis Créer un rôle.
3. Choisissez Politique de confiance personnalisée, puis insérez l'extrait de code suivant pour permettre au Amazon EventBridge planificateur d'assumer le rôle en votre nom.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                  "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

   Choisissez Suivant.
4. Sous Ajouter des autorisations, choisissez Créer une politique.
5. Choisissez JSON, puis insérez la politique suivante. Remplacez la *your-experiment-template-id* valeur par l'ID du modèle de votre expérience dans les étapes des conditions préalables.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": [
                "arn:aws:fis:*:*:experiment-template/your-experiment-template-id",
```

```
                "arn:aws:fis:*:*:experiment/*"
            ]
        }
    ]
}
```

Vous pouvez limiter le planificateur pour qu'il n'exécute que les modèles d'expériences AWS FIS dotés d'une valeur de balise spécifique. Par exemple, la politique suivante autorise toutes les expériences AWS FIS, mais limite le planificateur à exécuter uniquement les modèles d'expériences balisés. `StartExperiment` `Purpose=Schedule`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment/*"
        },
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Schedule"
                }
            }
        }
    ]
}
```

Choisissez Suivant : Balises.

6. Choisissez Suivant : Vérification.
7. Sous Réviser la politique, nommez votre stratégie`FIS_RecurringExperiment`, puis choisissez Créer une politique.
8. Sous Ajouter des autorisations, ajoutez la nouvelle `FIS_RecurringExperiment` politique à votre rôle, puis choisissez Suivant.
9. Sous Nom, révision et création, nommez le rôle`FIS_RecurringExperiment_role`, puis choisissez Créer un rôle.

Étape 2 : Création d'un Amazon EventBridge planificateur

Pour créer un Amazon EventBridge planificateur

1. Ouvrez la EventBridge console Amazon à l'adresse https://console.aws.amazon.com/events/.
2. Dans le volet de navigation de gauche, choisissez Schedules.
3. Vérifiez que vous êtes dans le même modèle Région AWS que votre modèle d'expérience AWS FIS.
4. Choisissez Créer un planning, puis renseignez les champs suivants :
   - Sous Nom du calendrier, insérez`FIS_recurring_experiment_tutorial`.
   - Sous Modèle de planification, sélectionnez Planification récurrente.
   - Sous Type de planification, sélectionnez Planification basée sur le taux.
   - Sous Expression de débit, sélectionnez 5 minutes.
   - Sous Fenêtre horaire flexible, sélectionnez Désactivé.
   - (Facultatif) Sous Période, sélectionnez votre fuseau horaire.
   - Choisissez Suivant.
5. Sous Sélectionner la cible, choisissez Tout APIs, puis recherchez AWS FIS.
6. Choisissez AWS FIS, puis sélectionnez StartExperiment.
7. Sous Entrée, insérez la charge utile JSON suivante. Remplacez la *`your-experiment-template-id`* valeur par l'ID du modèle de votre expérience. `ClientToken`Il s'agit d'un identifiant unique pour le planificateur. Dans ce didacticiel, nous utilisons un mot clé de contexte autorisé par Amazon EventBridge Scheduler. Pour plus d'informations, consultez la section Ajout d'attributs de contexte dans le guide de EventBridge l'utilisateur Amazon.

   ```
   {
       "ClientToken": "<aws.scheduler.execution-id>",
       "ExperimentTemplateId": "your-experiment-template-id"
   }
   ```

   Choisissez Suivant.
8. (Facultatif) Sous Paramètres, vous pouvez définir la politique de nouvelle tentative, la file d'attente des lettres mortes (DLQ) et les paramètres de chiffrement. Vous pouvez également conserver les valeurs par défaut.

9. Sous Autorisations, sélectionnez Utiliser le rôle existant, puis recherchez`FIS_RecurringExperiment_role`.
10. Choisissez Suivant.
11. Sous Réviser et créer un calendrier, passez en revue les détails de votre planificateur, puis choisissez Créer un calendrier.

Étape 3 : Vérifiez votre expérience

Pour vérifier que votre expérience AWS FIS s'est déroulée dans les délais

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation de gauche, sélectionnez Experiments.
3. Cinq minutes après avoir créé votre planning, vous devriez voir votre test s'exécuter.

Étape 4 : nettoyer

Pour désactiver votre Amazon EventBridge planificateur

1. Ouvrez la EventBridge console Amazon à l'adresse https://console.aws.amazon.com/events/.
2. Dans le volet de navigation de gauche, choisissez Schedules.
3. Sélectionnez le planificateur que vous venez de créer, puis choisissez Désactiver.

Utilisation de la bibliothèque de AWS FIS scénarios

Les scénarios définissent des événements ou des conditions que les clients peuvent appliquer pour tester la résilience de leurs applications, tels que l'interruption des ressources informatiques sur lesquelles l'application est exécutée. Les scénarios sont créés et gérés par AWS. Ils minimisent le travail indifférencié en vous fournissant un groupe de cibles prédéfinies et d'actions d'erreur (par exemple, l'arrêt de 30 % des instances d'un groupe de dimensionnement automatique) pour les défaillances courantes des applications.

Les scénarios sont fournis via une bibliothèque de scénarios réservée à la console et exécutés à l'aide d'un modèle d' AWS FIS expérience. Pour exécuter un test à l'aide d'un scénario, vous devez sélectionner le scénario dans la bibliothèque, spécifier les paramètres correspondant aux détails de votre charge de travail et l'enregistrer en tant que modèle de test dans votre compte.

Rubriques

- Visualisation d'un scénario
- Utilisation d'un scénario
- Exporter un scénario
- Référence de scénarios

Visualisation d'un scénario

Pour afficher un scénario à l'aide de la console :

1. Ouvrez la AWS FIS console à l'adressehttps://console.aws.amazon.com/fis/.
2. Dans le volet de navigation, sélectionnez Bibliothèque de scénarios.
3. Pour afficher les informations relatives à un scénario spécifique, sélectionnez la carte de scénario pour afficher un panneau divisé.
   - Dans l'onglet Description du panneau divisé au bas de la page, vous pouvez consulter une brève description du scénario. Vous pouvez également trouver un bref résumé des prérequis contenant un résumé des ressources cibles requises et des mesures que vous devez prendre pour préparer les ressources à utiliser dans le cadre du scénario. Enfin, vous pouvez également consulter des informations supplémentaires sur les cibles et les actions du scénario, ainsi que sur la durée prévue pendant laquelle l'expérience s'exécute avec succès avec les paramètres par défaut.

- Dans l'onglet Contenu du panneau divisé en bas de page, vous pouvez prévisualiser une version partiellement remplie du modèle d'expérience qui sera créé à partir du scénario.
- Dans l'onglet Détails du panneau divisé en bas de page, vous trouverez une explication détaillée de la mise en œuvre du scénario. Cela peut contenir des informations détaillées sur la manière dont les différents aspects du scénario sont approximés. Le cas échéant, vous pouvez également en savoir plus sur les métriques à utiliser comme conditions d'arrêt et pour fournir une observabilité afin de tirer les leçons de l'expérience. Enfin, vous trouverez des recommandations sur la manière d'étendre le modèle d'expérience obtenu.

Utilisation d'un scénario

Pour utiliser un scénario à l'aide de la console :

1. Ouvrez la AWS FIS console à l'adressehttps://console.aws.amazon.com/fis/.
2. Dans le volet de navigation, sélectionnez Bibliothèque de scénarios.
3. Pour afficher des informations sur un scénario spécifique, sélectionnez la carte de scénario pour afficher un panneau divisé
4. Pour utiliser le scénario, sélectionnez la carte de scénario et choisissez Créer un modèle avec un scénario.
5. Dans la vue Créer un modèle d'expérience, renseignez les éléments manquants.
   a. Certains scénarios vous permettent de modifier en bloc des paramètres partagés entre plusieurs actions ou cibles. Cette fonctionnalité sera désactivée une fois que vous aurez apporté des modifications au scénario, y compris les modifications apportées par la modification groupée des paramètres. Pour utiliser cette fonctionnalité, cliquez sur le bouton Modifier les paramètres groupés. Modifiez les paramètres dans le modal et sélectionnez le bouton Enregistrer.
   b. Certains modèles d'expérience peuvent comporter des paramètres d'action ou de cible manquants, mis en évidence sur chaque action et sur chaque carte cible. Sélectionnez le bouton Modifier pour chaque carte, ajoutez les informations manquantes et sélectionnez le bouton Enregistrer sur la carte.
   c. Tous les modèles nécessitent un rôle d'exécution d'accès au service. Vous pouvez choisir un rôle existant ou en créer un nouveau pour ce modèle d'expérience.
   d. Nous vous recommandons de définir une ou plusieurs conditions d'arrêt facultatives en sélectionnant une CloudWatch alarme AWS existante. En savoir plus sur Conditions d'arrêt

pour AWS FIS. Si aucune alarme n'est encore configurée, vous pouvez suivre les instructions de la section Utilisation d'Amazon CloudWatch Alarms et mettre à jour le modèle de test ultérieurement.

e. Nous vous recommandons d'activer les journaux d'expérience facultatifs dans CloudWatch les journaux Amazon ou dans un compartiment Amazon S3. En savoir plus sur Enregistrement des expériences pour AWS FIS. Si les ressources appropriées ne sont pas encore configurées, vous pouvez mettre à jour le modèle d'expérience ultérieurement.

6. Dans le champ Créer un modèle d'expérience, sélectionnez Créer un modèle d'expérience.
7. Dans la vue Modèles d'expériences de la AWS FIS console, sélectionnez Démarrer l'expérience. En savoir plus sur Gestion des AWS modèles d'expériences FIS.

Exporter un scénario

Les scénarios sont une expérience réservée à la console. Bien que similaires aux modèles d'expérience, les scénarios ne sont pas des modèles d'expérience complets et ne peuvent pas être directement importés dans AWS FIS. Si vous souhaitez utiliser des scénarios dans le cadre de votre propre automatisation, vous pouvez utiliser l'une des deux méthodes suivantes :

1. Suivez les étapes décrites Utilisation d'un scénario pour créer un modèle d' AWS FIS expérience valide et exportez ce modèle.
2. Suivez les étapes de l'étape 3 Visualisation d'un scénario et de l'étape 3, dans l'onglet Contenu, copiez et enregistrez le contenu du scénario, puis ajoutez les paramètres manquants manuellement pour créer un modèle d'expérience valide.

Référence de scénarios

Les scénarios inclus dans la bibliothèque de scénarios sont conçus pour utiliser des balises dans la mesure du possible et chaque scénario décrit les balises requises dans les sections Prérequis et Fonctionnement de la description du scénario. Vous pouvez étiqueter vos ressources avec ces balises prédéfinies ou définir vos propres balises à l'aide de l'expérience de modification des paramètres en bloc (voirUtilisation d'un scénario).

Cette référence décrit les scénarios courants de la bibliothèque de scénarios AWS FIS. Vous pouvez également répertorier les scénarios pris en charge à l'aide de la console AWS FIS.

Pour de plus amples informations, veuillez consulter Utilisation de la bibliothèque de AWS FIS scénarios.

AWS FIS prend en charge les EC2 scénarios Amazon suivants. Ces scénarios ciblent les instances à l'aide de balises. Vous pouvez utiliser vos propres balises ou utiliser les balises par défaut incluses dans le scénario. Certains de ces scénarios utilisent des documents SSM.

- EC2 stress : échec d'instance - Explorez l'effet d'une défaillance d'instance en arrêtant une ou plusieurs EC2 instances.

  Ciblez les instances de la région actuelle auxquelles une balise spécifique est attachée. Dans ce scénario, nous arrêterons ces instances et les redémarrerons à la fin de la durée de l'action, par défaut 5 minutes.
- EC2 stress : Disk - Découvrez l'impact d'une utilisation accrue du disque sur votre application EC2 basée.

  Dans ce scénario, nous ciblerons EC2 les instances de la région actuelle auxquelles une balise spécifique est attachée. Dans ce scénario, vous pouvez personnaliser une quantité croissante d'utilisation du disque injectée sur des EC2 instances ciblées pendant la durée de l'action, par défaut 5 minutes pour chaque action de stress sur le disque.
- EC2 stress : CPU - Explorez l'impact de l'augmentation du processeur sur votre application EC2 basée.

  Dans ce scénario, nous ciblerons EC2 les instances de la région actuelle auxquelles une balise spécifique est attachée. Dans ce scénario, vous pouvez personnaliser une quantité croissante de stress du processeur injectée sur des EC2 instances ciblées pendant la durée de l'action, par défaut 5 minutes pour chaque action de stress du processeur.
- EC2 stress : Mémoire - Découvrez l'impact d'une utilisation accrue de la mémoire sur votre application EC2 basée.

  Dans ce scénario, nous ciblerons EC2 les instances de la région actuelle auxquelles une balise spécifique est attachée. Dans ce scénario, vous pouvez personnaliser une quantité croissante de stress mnésique injecté sur des EC2 instances ciblées pendant la durée de l'action, par défaut 5 minutes pour chaque action de stress mnésique.
- EC2 stress : latence du réseau - Explorez l'impact de l'augmentation de la latence du réseau sur votre application EC2 basée.

Dans ce scénario, nous ciblerons EC2 les instances de la région actuelle auxquelles une balise spécifique est attachée. Dans ce scénario, vous pouvez personnaliser une quantité croissante de latence réseau injectée sur les EC2 instances ciblées pendant la durée de l'action, par défaut 5 minutes pour chaque action de latence.

AWS FIS prend en charge les scénarios Amazon EKS suivants. Ces scénarios ciblent les pods EKS à l'aide d'étiquettes d'application Kubernetes. Vous pouvez utiliser vos propres étiquettes ou utiliser les étiquettes par défaut incluses dans le scénario. Pour plus d'informations sur EKS avec FIS, consultezActions du EKS Pod.

- Stress lié à l'EKS : suppression du module - Découvrez l'effet d'une défaillance du module EKS en supprimant un ou plusieurs modules.

  Dans ce scénario, nous ciblerons les pods de la région actuelle associés à une étiquette d'application. Dans ce scénario, nous mettrons fin à tous les pods correspondants. La recréation des pods sera contrôlée par la configuration de Kubernetes.
- Stress lié à l'EKS : processeur - Découvrez l'impact d'une augmentation du processeur sur votre application basée sur EKS.

  Dans ce scénario, nous ciblerons les pods de la région actuelle associés à une étiquette d'application. Dans ce scénario, vous pouvez personnaliser une quantité croissante de stress du processeur injectée sur les pods EKS ciblés pendant la durée de l'action, par défaut 5 minutes pour chaque action de stress du processeur.
- Stress lié à l'EKS : disque - Découvrez l'impact d'une utilisation accrue du disque sur votre application basée sur EKS.

  Dans ce scénario, nous ciblerons les pods de la région actuelle associés à une étiquette d'application. Dans ce scénario, vous pouvez personnaliser une quantité croissante de stress du disque injectée sur les pods EKS ciblés pendant la durée de l'action, par défaut 5 minutes pour chaque action de stress du processeur.
- Stress lié à l'EKS : mémoire - Découvrez l'impact d'une utilisation accrue de la mémoire sur votre application basée sur EKS.

  Dans ce scénario, nous ciblerons les pods de la région actuelle associés à une étiquette d'application. Dans ce scénario, vous pouvez personnaliser une quantité croissante de stress mnésique injecté sur les pods EKS ciblés pendant la durée de l'action, par défaut 5 minutes pour chaque action de stress mnésique.

- Stress lié à l'EKS : latence du réseau - Découvrez l'impact de l'augmentation de la latence du réseau sur votre application basée sur EKS.

  Dans ce scénario, nous ciblerons les pods de la région actuelle associés à une étiquette d'application. Dans ce scénario, vous pouvez personnaliser une quantité croissante de latence réseau injectée sur les pods EKS ciblés pendant la durée de l'action, par défaut 5 minutes pour chaque action de latence.

AWS FIS prend en charge les scénarios suivants pour les applications multi-AZ et multi-régions. Ces scénarios ciblent plusieurs types de ressources.

- AZ Availability: Power Interruption- Injectez les symptômes attendus d'une interruption complète de l'alimentation dans une zone de disponibilité (AZ). En savoir plus sur AZ Availability: Power Interruption.
- Cross-Region: Connectivity- Bloquez le trafic réseau de l'application entre la région d'essai et la région de destination et interrompez la réplication des données entre régions. En savoir plus sur l'utilisationCross-Region: Connectivity.

AZ Availability: Power Interruption

Vous pouvez utiliser le AZ Availability: Power Interruption scénario pour induire les symptômes attendus d'une interruption complète de l'alimentation dans une zone de disponibilité (AZ).

Ce scénario peut être utilisé pour démontrer que les applications multi-AZ fonctionnent comme prévu lors d'une seule coupure de courant AZ complète. Cela inclut la perte de calcul zonal (Amazon EC2, EKS et ECS), l'absence de redimensionnement du calcul dans l'AZ, la perte de connectivité des sous-réseaux, le basculement sur RDS, le basculement sur incident et le manque de réponse des volumes ElastiCache EBS. Par défaut, les actions pour lesquelles aucune cible n'a été trouvée seront ignorées.

Actions

Ensemble, les actions suivantes créent bon nombre des symptômes attendus d'une coupure de courant complète dans une seule AZ. Disponibilité de la zone AZ : L'interruption de courant n'affecte que les services qui devraient subir un impact lors d'une seule interruption de courant de la zone AZ. Par défaut, le scénario injecte les symptômes de coupure de courant pendant 30 minutes, puis, pendant 30 minutes supplémentaires, les symptômes susceptibles de survenir pendant le rétablissement.

Arrêter les instances

Lors d'une coupure de courant de l'AZ, les EC2 instances de l'AZ affectée s'éteignent. Une fois l'alimentation rétablie, les instances redémarrent. AZ Availability: Power Interruptioninclut aws:ec2:stop-instances pour arrêter toutes les instances de l'AZ affectée pendant la durée de l'interruption. Après cette durée, les instances sont redémarrées. L'arrêt EC2 des instances gérées par Amazon EKS entraîne la suppression des pods EKS dépendants. L'arrêt EC2 des instances gérées par Amazon ECS entraîne l'arrêt des tâches ECS dépendantes.

Cette action cible les EC2 instances exécutées dans l'AZ affectée. Par défaut, il cible les instances dotées d'une balise nommée `AzImpairmentPower` avec une valeur de`StopInstances`. Vous pouvez ajouter cette balise à vos instances ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucune instance valide n'est trouvée, cette action sera ignorée.

Arrêtez les instances ASG

Lors d'une coupure de courant de la zone AZ, EC2 les instances gérées par un groupe Auto Scaling dans la zone de distribution affectée s'arrêteront. Une fois l'alimentation rétablie, les instances redémarrent. AZ Availability: Power Interruptioninclut aws:ec2:stop-instances pour arrêter toutes les instances, y compris celles gérées par Auto Scaling, dans l'AZ concernée pendant la durée de l'interruption. Après cette durée, les instances sont redémarrées.

Cette action cible les EC2 instances exécutées dans l'AZ affectée. Par défaut, il cible les instances dotées d'une balise nommée `AzImpairmentPower` avec une valeur de`IceAsg`. Vous pouvez ajouter cette balise à vos instances ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucune instance valide n'est trouvée, cette action sera ignorée.

Interrompre les lancements d'instances

Lors d'une coupure de courant de l'AZ, les appels d' EC2 API visant à fournir de la capacité dans l'AZ échoueront. En particulier, les éléments suivants APIs seront affectés : `ec2:StartInstancesec2:CreateFleet`, et`ec2:RunInstances`. AZ Availability: Power Interruption includesinclut aws:ec2 : api-insufficient-instance-capacity -error pour empêcher le provisionnement de nouvelles instances dans l'AZ concernée.

Cette action cible les rôles IAM utilisés pour approvisionner des instances. Ils doivent être ciblés à l'aide d'un ARN. Par défaut, si aucun rôle IAM valide n'est trouvé, cette action sera ignorée.

Suspendre le dimensionnement ASG

Lors d'une coupure de courant dans l'AZ, les appels d' EC2 API effectués par le plan de contrôle Auto Scaling pour récupérer la capacité perdue dans l'AZ échoueront. En particulier, les éléments suivants APIs seront affectés : `ec2:StartInstancesec2:CreateFleet`, et`ec2:RunInstances`. AZ Availability: Power Interruptioninclut aws:ec2 : asg-insufficient-instance-capacity -error pour empêcher le provisionnement de nouvelles instances dans l'AZ concernée. Cela empêche également Amazon EKS et Amazon ECS de s'adapter à l'AZ concernée.

Cette action cible les groupes Auto Scaling. Par défaut, il cible les groupes Auto Scaling dotés d'une balise nommée `AzImpairmentPower` avec une valeur de`IceAsg`. Vous pouvez ajouter cette balise à vos groupes Auto Scaling ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucun groupe Auto Scaling valide n'est trouvé, cette action sera ignorée.

Suspendre la connectivité réseau

Lors d'une coupure de courant de l'AZ, le réseau de l'AZ ne sera pas disponible. Dans ce cas, la mise à jour du DNS de certains services AWS peut prendre jusqu'à quelques minutes afin de tenir compte du fait que les points de terminaison privés de l'AZ concernée ne sont pas disponibles. Pendant ce temps, les recherches DNS peuvent renvoyer des adresses IP inaccessibles. AZ Availability: Power Interruptioninclut aws:network:disrupt-connectivity pour bloquer toute connectivité réseau pour tous les sous-réseaux de l'AZ affectée pendant 2 minutes. Cela forcera les délais d'expiration et les actualisations du DNS pour la plupart des applications. L'arrêt de l'action au bout de 2 minutes permet la restauration ultérieure du DNS du service régional alors que l'AZ continue d'être indisponible.

Cette action cible les sous-réseaux. Par défaut, il cible les clusters dotés d'une balise nommée `AzImpairmentPower` avec une valeur de`DisruptSubnet`. Vous pouvez ajouter cette balise à vos sous-réseaux ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucun sous-réseau valide n'est trouvé, cette action sera ignorée.

Faillover RDS

Lors d'une coupure de courant de l'AZ, les nœuds RDS de l'AZ affectée s'arrêteront. Les nœuds AZ RDS individuels de l'AZ affectée seront totalement indisponibles. Pour les clusters multi-AZ, le nœud d'écriture basculera vers une zone de zone non affectée et les nœuds de lecture de la zone de référence affectée ne seront pas disponibles. Pour les clusters multi-AZ, AZ Availability: Power Interruption inclut aws:rds : failover-db-cluster pour basculer si le rédacteur se trouve dans l'AZ concerné.

Cette action cible les clusters RDS. Par défaut, il cible les clusters dotés d'une balise nommée `AzImpairmentPower` avec une valeur de`DisruptRds`. Vous pouvez ajouter cette balise à vos clusters ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucun cluster valide n'est trouvé, cette action sera ignorée.

Suspendre le groupe ElastiCache de réplication

Lors d'une coupure de courant AZ, ElastiCache les nœuds de l'AZ ne sont pas disponibles. AZ Availability: Power Interruptioninclut aws:elasticache : replicationgroup-interrupt-az-power pour terminer les ElastiCache nœuds de l'AZ affectée. Pendant toute la durée de l'interruption, les nouvelles instances ne seront pas mises en service dans l'AZ concernée, de sorte que le groupe de réplication conservera sa capacité réduite.

Cette action cible les groupes ElastiCache de réplication. Par défaut, il cible les groupes de réplication dotés d'une balise nommée `AzImpairmentPower` avec une valeur de`ElasticacheImpact`. Vous pouvez ajouter cette balise à vos groupes de réplication ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucun groupe de réplication valide n'est trouvé, cette action sera ignorée. Notez que seuls les groupes de réplication dont les nœuds d'écriture se trouvent dans l'AZ affectée seront considérés comme des cibles valides.

Démarrez ARC Zonal Autoshift

Cinq minutes après le début de la coupure de courant de l'AZ, l'action de restauration déplace `aws:arc:start-zonal-autoshift` automatiquement le trafic des ressources hors de l'AZ spécifié pendant les 25 minutes restantes de l'interruption de courant. Passé ce délai, le trafic revient à l'AZ d'origine. Notez que lors d'une panne de courant réelle en mode AZ, la panne AWS de courant est détectée et le trafic des ressources est décalé si le changement automatique est activé. Bien que le moment de ce changement varie, on estime qu'il se produira cinq minutes après le début de la déficience.

Cette action cible les ressources activées par le système Autoshift d'Amazon Application Recovery Controller (ARC). Par défaut, il cible les ressources avec la clé `AzImpairmentPower` et la valeur du tag`RecoverAutoshiftResources`. Vous pouvez ajouter cette balise à vos ressources ou remplacer la balise par défaut par la vôtre dans le modèle d'expérience. Par exemple, vous souhaiterez peut-être utiliser une balise spécifique à l'application. Par défaut, si aucune ressource valide n'est trouvée, cette action sera ignorée.

Suspendre les E/S EBS

Après une coupure de courant AZ, une fois l'alimentation rétablie, un très faible pourcentage d'instances peut rencontrer des volumes EBS qui ne répondent pas. AZ Availability: Power Interruptioninclut aws:ebs:pause-io pour laisser 1 volume EBS sans réponse.

Par défaut, seuls les volumes définis pour persister après la fermeture de l'instance sont ciblés. Cette action cible les volumes dotés d'une balise nommée `AzImpairmentPower` avec une valeur de`APIPauseVolume`. Vous pouvez ajouter cette balise à vos volumes ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucun volume valide n'est trouvé, cette action sera ignorée.

Limites

- Ce scénario n'inclut pas les conditions d'arrêt. Les conditions d'arrêt correctes pour votre application doivent être ajoutées au modèle d'expérience.
- Dans l'AZ cible, les Amazon EKS Pods exécutés sur des nœuds de EC2 travail EC2 seront interrompus et le démarrage de nouveaux EC2 nœuds sera bloqué. Toutefois, les Amazon EKS Pods exécutés sur AWS Fargate ne sont pas pris en charge.
- Dans l'AZ cible, les tâches Amazon ECS en cours d'exécution EC2 seront interrompues avec EC2 les nœuds de travail et le démarrage de nouveaux EC2 nœuds sera bloqué. Toutefois, les tâches Amazon ECS exécutées sur AWS Fargate ne sont pas prises en charge.
- Amazon RDS Multi-AZ avec deux instances de base de données de secours lisibles n'est pas pris en charge. Dans ce cas, les instances seront résiliées, le RDS basculera sur incident et la capacité sera immédiatement rétablie dans l'AZ concernée. Le mode veille lisible dans l'AZ concerné restera disponible.

Prérequis

- Ajoutez l'autorisation requise au rôle d'expérience AWS FIS.
- Les balises de ressources doivent être appliquées aux ressources qui doivent être ciblées par l'expérience. Ils peuvent utiliser votre propre convention de balisage ou les balises par défaut définies dans le scénario.

Autorisations

L'autoshift de zone ARC utilise un rôle lié au service IAM `AWSServiceRoleForZonalAutoshiftPracticeRun` pour effectuer le changement de zone en votre nom. Ce rôle utilise la politique `AWSZonalAutoshiftPracticeRunSLRPolicy`gérée par IAM. Il n'est pas nécessaire de créer le rôle manuellement. Lorsque vous créez un modèle d'expérience à partir du scénario AZ Power Interruption dans le AWS Management Console AWS CLI, le ou un AWS SDK, ARC crée le rôle lié au service pour vous. Pour plus d'informations, voir Utilisation du rôle lié au service pour le changement automatique de zone dans ARC.

La politique suivante accorde à AWS FIS les autorisations nécessaires pour exécuter un test avec le AZ Availability: Power Interruption scénario. Cette politique doit être associée au rôle d'expérimentation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowFISExperimentLoggingActionsCloudwatch",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:network-acl/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateNetworkAcl",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateNetworkAcl",
```

```
            "Resource": "arn:aws:ec2:*:*:network-acl/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkAclEntry",
                "ec2:DeleteNetworkAcl"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-acl/*",
                "arn:aws:ec2:*:*:vpc/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateNetworkAcl",
            "Resource": "arn:aws:ec2:*:*:vpc/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcs",
                "ec2:DescribeManagedPrefixLists",
                "ec2:DescribeSubnets",
                "ec2:DescribeNetworkAcls"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:ReplaceNetworkAclAssociation",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:network-acl/*"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "rds:FailoverDBCluster"
            ],
            "Resource": [
                "arn:aws:rds:*:*:cluster:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "rds:RebootDBInstance"
            ],
            "Resource": [
                "arn:aws:rds:*:*:db:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:DescribeReplicationGroups",
                "elasticache:InterruptClusterAzPower"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ]
        },
        {
            "Sid": "TargetResolutionByTags",
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
```

```
            "Resource": "arn:aws:ec2:*:*:instance/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant"
            ],
            "Resource": [
                "arn:aws:kms:*:*:key/*"
            ],
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "ec2.*.amazonaws.com"
                },
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:PauseVolumeIO"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*"
        },
        {
            "Sid": "AllowInjectAPI",
            "Effect": "Allow",
            "Action": [
```

```
                "ec2:InjectApiError"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "ec2:FisActionId": [
                        "aws:ec2:api-insufficient-instance-capacity-error",
                        "aws:ec2:asg-insufficient-instance-capacity-error"
                    ]
                }
            }
        },
        {
            "Sid": "DescribeAsg",
            "Effect": "Allow",
            "Action": [
                "autoscaling:DescribeAutoScalingGroups"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Contenu du scénario

Le contenu suivant définit le scénario. Ce JSON peut être enregistré et utilisé pour créer un modèle d'expérience à l'aide de la create-experiment-templatecommande de l'AWS Command Line Interface (AWS CLI). Pour obtenir la version la plus récente du scénario, consultez la bibliothèque de scénarios de la console FIS.

```
{
    "targets": {
        "IAM-role": {
            "resourceType": "aws:iam:role",
            "resourceArns": [],
            "selectionMode": "ALL"
        },
        "EBS-Volumes": {
```

```
            "resourceType": "aws:ec2:ebs-volume",
            "resourceTags": {
                "AzImpairmentPower": "ApiPauseVolume"
            },
            "selectionMode": "COUNT(1)",
            "parameters": {
                "availabilityZoneIdentifier": "us-east-1a"
            },
            "filters": [
                {
                    "path": "Attachments.DeleteOnTermination",
                    "values": [
                        "false"
                    ]
                }
            ]
        },
        "EC2-Instances": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "AzImpairmentPower": "StopInstances"
            },
            "filters": [
                {
                    "path": "State.Name",
                    "values": [
                        "running"
                    ]
                },
                {
                    "path": "Placement.AvailabilityZone",
                    "values": [
                        "us-east-1a"
                    ]
                }
            ],
            "selectionMode": "ALL"
        },
        "ASG": {
            "resourceType": "aws:ec2:autoscaling-group",
            "resourceTags": {
                "AzImpairmentPower": "IceAsg"
            },
            "selectionMode": "ALL"
```

```
        },
        "ASG-EC2-Instances": {
            "resourceType": "aws:ec2:instance",
            "resourceTags": {
                "AzImpairmentPower": "IceAsg"
            },
            "filters": [
                {
                    "path": "State.Name",
                    "values": [
                        "running"
                    ]
                },
                {
                    "path": "Placement.AvailabilityZone",
                    "values": [
                        "us-east-1a"
                    ]
                }
            ],
            "selectionMode": "ALL"
        },
        "Subnet": {
            "resourceType": "aws:ec2:subnet",
            "resourceTags": {
                "AzImpairmentPower": "DisruptSubnet"
            },
            "filters": [
                {
                    "path": "AvailabilityZone",
                    "values": [
                        "us-east-1a"
                    ]
                }
            ],
            "selectionMode": "ALL",
            "parameters": {}
        },
        "RDS-Cluster": {
            "resourceType": "aws:rds:cluster",
            "resourceTags": {
                "AzImpairmentPower": "DisruptRds"
            },
            "selectionMode": "ALL",
```

```
            "parameters": {
                "writerAvailabilityZoneIdentifiers": "us-east-1a"
            }
        },
        "ElastiCache-Cluster": {
            "resourceType": "aws:elasticache:replicationgroup",
            "resourceTags": {
                "AzImpairmentPower": "DisruptElasticache"
            },
            "selectionMode": "ALL",
            "parameters": {
                "availabilityZoneIdentifier": "us-east-1a"
            }
        }
    },
    "actions": {
        "Pause-Instance-Launches": {
            "actionId": "aws:ec2:api-insufficient-instance-capacity-error",
            "parameters": {
                "availabilityZoneIdentifiers": "us-east-1a",
                "duration": "PT30M",
                "percentage": "100"
            },
            "targets": {
                "Roles": "IAM-role"
            }
        },
        "Pause-EBS-IO": {
            "actionId": "aws:ebs:pause-volume-io",
            "parameters": {
                "duration": "PT30M"
            },
            "targets": {
                "Volumes": "EBS-Volumes"
            },
            "startAfter": [
                "Stop-Instances",
                "Stop-ASG-Instances"
            ]
        },
        "Stop-Instances": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "completeIfInstancesTerminated": "true",
```

```
                "startInstancesAfterDuration": "PT30M"
            },
            "targets": {
                "Instances": "EC2-Instances"
            }
        },
        "Pause-ASG-Scaling": {
            "actionId": "aws:ec2:asg-insufficient-instance-capacity-error",
            "parameters": {
                "availabilityZoneIdentifiers": "us-east-1a",
                "duration": "PT30M",
                "percentage": "100"
            },
            "targets": {
                "AutoScalingGroups": "ASG"
            }
        },
        "Stop-ASG-Instances": {
            "actionId": "aws:ec2:stop-instances",
            "parameters": {
                "completeIfInstancesTerminated": "true",
                "startInstancesAfterDuration": "PT30M"
            },
            "targets": {
                "Instances": "ASG-EC2-Instances"
            }
        },
        "Pause-network-connectivity": {
            "actionId": "aws:network:disrupt-connectivity",
            "parameters": {
                "duration": "PT2M",
                "scope": "all"
            },
            "targets": {
                "Subnets": "Subnet"
            }
        },
        "Failover-RDS": {
            "actionId": "aws:rds:failover-db-cluster",
            "parameters": {},
            "targets": {
                "Clusters": "RDS-Cluster"
            }
        },
```

```
        "Pause-ElastiCache": {
            "actionId": "aws:elasticache:replicationgroup-interrupt-az-power",
            "parameters": {
                "duration": "PT30M"
            },
            "targets": {
                "ReplicationGroups": "ElastiCache-Cluster"
            }
        }
    },
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": ""
        }
    ],
    "roleArn": "",
    "tags": {
        "Name": "AZ Impairment: Power Interruption"
    },
    "logConfiguration": {
        "logSchemaVersion": 2
    },
    "experimentOptions": {
        "accountTargeting": "single-account",
        "emptyTargetResolutionMode": "skip"
    },
    "description": "Affect multiple resource types in a single AZ, targeting by tags
 and explicit ARNs, to approximate power interruption in one AZ."
}
```

Cross-Region: Connectivity

Vous pouvez utiliser Cross-Region: Connectivity ce scénario pour bloquer le trafic réseau de l'application entre la région d'essai et la région de destination et suspendre la réplication entre régions pour Amazon S3 et Amazon DynamoDB. Interrégional : la connectivité affecte le trafic applicatif sortant de la région dans laquelle vous exécutez le test (région d'essai). Le trafic entrant apatride en provenance de la région que vous souhaitez isoler de la région de test (région de destination) peut ne pas être bloqué. Le trafic provenant des services gérés AWS ne peut pas être bloqué.

Ce scénario peut être utilisé pour démontrer que les applications multirégionales fonctionnent comme prévu lorsque les ressources de la région de destination ne sont pas accessibles depuis la région d'expérimentation. Cela inclut le blocage du trafic réseau entre la région expérimentale et la région de destination en ciblant les passerelles de transit et les tables de routage. Il interrompt également la réplication entre régions pour S3 et DynamoDB. Par défaut, les actions pour lesquelles aucune cible n'a été trouvée seront ignorées.

Actions

Ensemble, les actions suivantes bloquent la connectivité entre régions pour les services AWS inclus. Les actions sont exécutées en parallèle. Par défaut, le scénario bloque le trafic pendant 3 heures, que vous pouvez augmenter jusqu'à une durée maximale de 12 heures.

Perturber la connectivité de Transit Gateway

Cross Region: Connectivityinclut aws:network : transit-gateway-disrupt-cross -region-connectivity pour bloquer le trafic réseau interrégional entre la région d' VPCs essai et la région de destination connectée par une passerelle de transit. VPCs Cela n'affecte pas l'accès aux points de terminaison VPC au sein de la région d'expérience, mais bloquera le trafic en provenance de la région d'expérience destiné à un point de terminaison VPC dans la région de destination.

Cette action cible les passerelles de transit reliant la région expérimentale à la région de destination. Par défaut, il cible les passerelles de transit avec une balise nommée `DisruptTransitGateway` avec une valeur de`Allowed`. Vous pouvez ajouter cette balise à vos passerelles de transport en commun ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucune passerelle de transit valide n'est trouvée, cette action sera ignorée.

Interrompre la connectivité des sous-réseaux

Cross Region: Connectivityinclut aws:network : route-table-disrupt-cross -region-connectivity pour bloquer le trafic réseau interrégional depuis la région d'essai vers VPCs les blocs IP AWS publics de la région de destination. Ces blocs IP publics incluent les points de terminaison des services AWS dans la région de destination, par exemple le point de terminaison régional S3, et les blocs IP AWS pour les services gérés, par exemple les adresses IP utilisées pour les équilibreurs de charge et Amazon API Gateway. Cette action bloque également la connectivité réseau via les connexions d'appairage VPC interrégionales entre la région d'essai et la région de destination. Cela n'affecte pas l'accès aux points de terminaison VPC dans la région d'expérimentation, mais bloque le trafic en provenance de la région d'expérience destiné à un point de terminaison VPC dans la région de destination.

Cette action cible les sous-réseaux de la région d'expérimentation. Par défaut, il cible les sous-réseaux dotés d'une balise nommée `DisruptSubnet` avec une valeur de`Allowed`. Vous pouvez ajouter cette balise à vos sous-réseaux ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucun sous-réseau valide n'est trouvé, cette action sera ignorée.

Suspendre la réplication S3

Cross Region: Connectivityinclut aws:s3 : bucket-pause-replication pour suspendre la réplication S3 de la région d'expérience vers la région de destination pour les buckets ciblés. La réplication de la région de destination vers la région d'expérimentation ne sera pas affectée. Une fois le scénario terminé, la réplication des compartiments reprendra à partir du point où elle avait été interrompue. Notez que le temps nécessaire à la réplication pour synchroniser tous les objets varie en fonction de la durée de l'expérience et du taux de chargement des objets vers le compartiment.

Cette action cible les compartiments S3 de la région d'expérience avec la réplication entre régions (CRR) activée vers un compartiment S3 de la région de destination. Par défaut, il cible les compartiments dotés d'une balise nommée `DisruptS3` avec une valeur de`Allowed`. Vous pouvez ajouter cette balise à vos compartiments ou remplacer la balise par défaut par la vôtre dans le modèle d'expérience. Par défaut, si aucun compartiment valide n'est trouvé, cette action sera ignorée.

Suspendre la réplication DynamoDB

Cross-Region: Connectivityinclut aws:dynamodb : global-table-pause-replication pour suspendre la réplication entre la région expérimentale et toutes les autres régions, y compris la région de destination. Cela empêche la réplication vers et hors de la région d'expérience, mais n'affecte pas la réplication entre les autres régions. Une fois le scénario terminé, la réplication des tables reprendra à partir du point où elle avait été interrompue. Notez que le temps nécessaire à la réplication pour synchroniser toutes les données varie en fonction de la durée de l'expérience et du taux de modifications apportées à la table.

Cette action cible les tables globales DynamoDB de la région d'expérimentation. Par défaut, il cible les tables dotées d'une balise nommée `DisruptDynamoDb` avec une valeur de`Allowed`. Vous pouvez ajouter cette balise à vos tableaux ou remplacer la balise par défaut par votre propre balise dans le modèle d'expérience. Par défaut, si aucune table globale valide n'est trouvée, cette action sera ignorée.

Limites

- Ce scénario n'inclut pas les conditions d'arrêt. Les conditions d'arrêt correctes pour votre application doivent être ajoutées au modèle d'expérience.

Prérequis

- Ajoutez l'autorisation requise au rôle d'expérience AWS FIS.
- Les balises de ressources doivent être appliquées aux ressources qui doivent être ciblées par l'expérience. Ils peuvent utiliser votre propre convention de balisage ou les balises par défaut définies dans le scénario.

Autorisations

La politique suivante accorde à AWS FIS les autorisations nécessaires pour exécuter un test avec le Cross-Region: Connectivity scénario. Cette politique doit être associée au rôle d'expérimentation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RouteTableDisruptConnectivity1",
            "Effect": "Allow",
            "Action": "ec2:CreateRouteTable",
            "Resource": "arn:aws:ec2:*:*:route-table/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity2",
            "Effect": "Allow",
            "Action": "ec2:CreateRouteTable",
            "Resource": "arn:aws:ec2:*:*:vpc/*"
        },
        {
            "Sid": "RouteTableDisruptConnectivity21",
            "Effect": "Allow",
```

```
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:route-table/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateRouteTable",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity3",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateNetworkInterface",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity4",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateManagedPrefixList",
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity5",
            "Effect": "Allow",
            "Action": "ec2:DeleteRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:route-table/*",
                "arn:aws:ec2:*:*:vpc/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
```

```
            }
        }
    },
    {
        "Sid": "RouteTableDisruptConnectivity6",
        "Effect": "Allow",
        "Action": "ec2:CreateRoute",
        "Resource": "arn:aws:ec2:*:*:route-table/*",
        "Condition": {
            "StringEquals": {
                "ec2:ResourceTag/managedByFIS": "true"
            }
        }
    },
    {
        "Sid": "RouteTableDisruptConnectivity7",
        "Effect": "Allow",
        "Action": "ec2:CreateNetworkInterface",
        "Resource": "arn:aws:ec2:*:*:network-interface/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/managedByFIS": "true"
            }
        }
    },
    {
        "Sid": "RouteTableDisruptConnectivity8",
        "Effect": "Allow",
        "Action": "ec2:CreateNetworkInterface",
        "Resource": [
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:security-group/*"
        ]
    },
    {
        "Sid": "RouteTableDisruptConnectivity9",
        "Effect": "Allow",
        "Action": "ec2:DeleteNetworkInterface",
        "Resource": "arn:aws:ec2:*:*:network-interface/*",
        "Condition": {
            "StringEquals": {
                "ec2:ResourceTag/managedByFIS": "true"
            }
        }
```

```
        },
        {
            "Sid": "RouteTableDisruptConnectivity10",
            "Effect": "Allow",
            "Action": "ec2:CreateManagedPrefixList",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity11",
            "Effect": "Allow",
            "Action": "ec2:DeleteManagedPrefixList",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity12",
            "Effect": "Allow",
            "Action": "ec2:ModifyManagedPrefixList",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity13",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcs",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeManagedPrefixLists",
                "ec2:DescribeSubnets",
                "ec2:DescribeRouteTables",
```

```
                "ec2:DescribeVpcEndpoints"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RouteTableDisruptConnectivity14",
            "Effect": "Allow",
            "Action": "ec2:ReplaceRouteTableAssociation",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:route-table/*"
            ]
        },
        {
            "Sid": "RouteTableDisruptConnectivity15",
            "Effect": "Allow",
            "Action": "ec2:GetManagedPrefixListEntries",
            "Resource": "arn:aws:ec2:*:*:prefix-list/*"
        },
        {
            "Sid": "RouteTableDisruptConnectivity16",
            "Effect": "Allow",
            "Action": "ec2:AssociateRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:route-table/*"
            ]
        },
        {
            "Sid": "RouteTableDisruptConnectivity17",
            "Effect": "Allow",
            "Action": "ec2:DisassociateRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:route-table/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity18",
            "Effect": "Allow",
```

```
            "Action": "ec2:DisassociateRouteTable",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*"
            ]
        },
        {
            "Sid": "RouteTableDisruptConnectivity19",
            "Effect": "Allow",
            "Action": "ec2:ModifyVpcEndpoint",
            "Resource": [
                "arn:aws:ec2:*:*:route-table/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/managedByFIS": "true"
                }
            }
        },
        {
            "Sid": "RouteTableDisruptConnectivity20",
            "Effect": "Allow",
            "Action": "ec2:ModifyVpcEndpoint",
            "Resource": [
                "arn:aws:ec2:*:*:vpc-endpoint/*"
            ]
        },
        {
            "Sid": "TransitGatewayDisruptConnectivity1",
            "Effect": "Allow",
            "Action": [
                "ec2:DisassociateTransitGatewayRouteTable",
                "ec2:AssociateTransitGatewayRouteTable"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:transit-gateway-route-table/*",
                "arn:aws:ec2:*:*:transit-gateway-attachment/*"
            ]
        },
        {
            "Sid": "TransitGatewayDisruptConnectivity2",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTransitGatewayPeeringAttachments",
                "ec2:DescribeTransitGatewayAttachments",
```

```
                "ec2:DescribeTransitGateways"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3CrossRegion1",
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3CrossRegion2",
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3CrossRegion3",
            "Effect": "Allow",
            "Action": [
                "s3:PauseReplication"
            ],
            "Resource": "arn:aws:s3:::*",
            "Condition": {
                "StringLike": {
                    "s3:DestinationRegion": "*"
                }
            }
        },
        {
            "Sid": "S3CrossRegion4",
            "Effect": "Allow",
            "Action": [
                "s3:GetReplicationConfiguration",
                "s3:PutReplicationConfiguration"
            ],
            "Resource": "arn:aws:s3:::*",
            "Condition": {
                "BoolIfExists": {
                    "s3:isReplicationPauseRequest": "true"
```

```
                }
            }
        },
        {
            "Sid": "DdbCrossRegion1",
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DdbCrossRegion",
            "Effect": "Allow",
            "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:PutResourcePolicy",
                "dynamodb:GetResourcePolicy",
                "dynamodb:DeleteResourcePolicy"
            ],
            "Resource": [
                "arn:aws:dynamodb:*:*:table/*",
            ]
        }
    ]
}
```

Contenu du scénario

Le contenu suivant définit le scénario. Ce JSON peut être enregistré et utilisé pour créer un modèle d'expérience à l'aide de la create-experiment-templatecommande de l'AWS Command Line Interface (AWS CLI). Pour obtenir la version la plus récente du scénario, consultez la bibliothèque de scénarios de la console FIS.

```
{
        "targets": {
                "Transit-Gateway": {
                        "resourceType": "aws:ec2:transit-gateway",
                        "resourceTags": {
                                "TgwTag": "TgwValue"
                        },
                        "selectionMode": "ALL"
```

```
                },
                "Subnet": {
                        "resourceType": "aws:ec2:subnet",
                        "resourceTags": {
                                "SubnetKey": "SubnetValue"
                        },
                        "selectionMode": "ALL",
                        "parameters": {}
                },
                "S3-Bucket": {
                        "resourceType": "aws:s3:bucket",
                        "resourceTags": {
                                "S3Impact": "Allowed"
                        },
                        "selectionMode": "ALL"
                },
                "DynamoDB-Global-Table": {
                        "resourceType": "aws:dynamodb:global-table",
                        "resourceTags": {
                                "DisruptDynamoDb": "Allowed"
                        },
                        "selectionMode": "ALL"
                }
        },
        "actions": {
                "Disrupt-Transit-Gateway-Connectivity": {
                        "actionId": "aws:network:transit-gateway-disrupt-cross-region-
connectivity",
                        "parameters": {
                                "duration": "PT3H",
                                "region": "eu-west-1"
                        },
                        "targets": {
                                "TransitGateways": "Transit-Gateway"
                        }
                },
                "Disrupt-Subnet-Connectivity": {
                        "actionId": "aws:network:route-table-disrupt-cross-region-
connectivity",
                        "parameters": {
                                "duration": "PT3H",
                                "region": "eu-west-1"
                        },
                        "targets": {
```

```
                                "Subnets": "Subnet"
                        }
                },
                "Pause-S3-Replication": {
                        "actionId": "aws:s3:bucket-pause-replication",
                        "parameters": {
                                "duration": "PT3H",
                                "region": "eu-west-1"
                        },
                        "targets": {
                                "Buckets": "S3-Bucket"
                        }
                },
                "Pause-DynamoDB-Replication": {
                        "actionId": "aws:dynamodb:global-table-pause-replication",
                        "parameters": {
                                "duration": "PT3H"
                        },
                        "targets": {
                                "Tables": "DynamoDB-Global-Table"
                        }
                }
        },
        "stopConditions": [
                {
                        "source": "none"
                }
        ],
        "roleArn": "",
        "logConfiguration": {
                "logSchemaVersion": 2
        },
        "tags": {
                "Name": "Cross-Region: Connectivity"
        },
        "experimentOptions": {
                "accountTargeting": "single-account",
                "emptyTargetResolutionMode": "skip"
        },
        "description": "Block application network traffic from experiment Region to
 target Region and pause cross-Region replication"
}
```

Travailler avec des expériences multi-comptes pour AWS FIS

Vous pouvez créer et gérer des modèles d'expériences multi-comptes à l'aide de la AWS FIS console ou de la ligne de commande. Vous créez un test multi-comptes en spécifiant l'option de test de ciblage des comptes sous `"multi-account"` la forme et en ajoutant des configurations de compte cible. Après avoir créé un modèle de test multi-comptes, vous pouvez l'utiliser pour exécuter un test.

Grâce à un test multi-comptes, vous pouvez configurer et exécuter des scénarios de défaillance réels sur une application qui couvre plusieurs AWS comptes au sein d'une même région. Vous exécutez des tests multi-comptes à partir d'un compte orchestrateur qui ont un impact sur les ressources de plusieurs comptes cibles.

Lorsque vous effectuez un test multi-comptes, les comptes cibles dont les ressources sont affectées sont avertis via leurs tableaux de bord AWS Health, afin de sensibiliser les utilisateurs des comptes cibles. Grâce aux expériences multi-comptes, vous pouvez :

- Exécutez des scénarios de défaillance réels sur des applications couvrant plusieurs comptes grâce aux commandes centralisées et aux garde-fous fournis. AWS FIS
- Contrôlez les effets d'une expérience multi-comptes à l'aide de rôles IAM dotés d'autorisations et de balises précises pour définir la portée de chaque cible.
- Visualisez de manière centralisée les AWS FIS actions entreprises dans chaque compte à partir des AWS FIS journaux AWS Management Console et via ceux-ci.
- Surveillez et auditez les appels AWS FIS d'API effectués dans chaque compte avec AWS CloudTrail.

Cette section vous aide à démarrer des expériences multi-comptes.

Rubriques

- Concepts pour les expériences multi-comptes
- Bonnes pratiques pour les expériences multi-comptes
- Conditions préalables pour les expériences multi-comptes
- Création d'un modèle d'expérience multi-comptes
- Mettre à jour la configuration d'un compte cible

- Supprimer une configuration de compte cible

Concepts pour les expériences multi-comptes

Les concepts clés des expériences multi-comptes sont les suivants :

- Compte d'orchestrateur - Le compte d'orchestrateur agit comme un compte central pour configurer et gérer l'expérience dans la AWS FIS console, ainsi que pour centraliser la journalisation. Le compte de l'orchestrateur est propriétaire du modèle AWS FIS d'expérience et de l'expérience.
- Comptes cibles - Un compte cible est un compte AWS individuel dont les ressources peuvent être affectées par une expérience AWS FIS multi-comptes.
- Configurations de compte cible - Vous définissez les comptes cibles qui font partie d'un test en ajoutant des configurations de compte cible au modèle de test. Une configuration de compte cible est un élément du modèle d'expérience requis pour les expériences multi-comptes. Vous en définissez un pour chaque compte cible en définissant un ID de AWS compte, un rôle IAM et une description facultative.

Bonnes pratiques pour les expériences multi-comptes

Les meilleures pratiques relatives à l'utilisation d'expériences multi-comptes sont les suivantes :

- Lorsque vous configurez des cibles pour des tests multi-comptes, nous vous recommandons de cibler avec des balises de ressources cohérentes sur tous les comptes cibles. Une AWS FIS expérience permettra de résoudre les ressources dotées de balises cohérentes dans chaque compte cible. Une action doit résoudre au moins une ressource cible dans un compte cible, sinon elle échouera, sauf pour les expériences avec la valeur `emptyTargetResolutionMode` définie sur`skip`. Des quotas d'action s'appliquent par compte. Si vous souhaitez cibler les ressources par ressource ARNs, la même limite de compte unique par action s'applique.
- Lorsque vous ciblez des ressources dans une ou plusieurs zones de disponibilité à l'aide de paramètres ou de filtres, vous devez spécifier un ID AZ, et non un nom AZ. L'AZ ID est un identifiant unique et cohérent pour une zone de disponibilité pour tous les comptes. Pour savoir comment trouver l'ID AZ des zones de disponibilité de votre compte, consultez la section Zone de disponibilité IDs de vos ressources AWS.

Conditions préalables pour les expériences multi-comptes

Pour utiliser les conditions d'arrêt pour une expérience multi-comptes, vous devez d'abord configurer les alarmes entre comptes. Les rôles IAM sont définis lorsque vous créez un modèle d'expérience multi-comptes. Vous pouvez créer les rôles IAM nécessaires avant de créer le modèle.

Contenu

- Autorisations pour les expériences multi-comptes
- Conditions d'arrêt pour les expériences multi-comptes (facultatif)
- Leviers de sécurité pour les expériences multi-comptes (en option)

Autorisations pour les expériences multi-comptes

Les expériences multi-comptes utilisent le chaînage des rôles IAM pour accorder des autorisations permettant d' AWS FIS effectuer des actions sur les ressources des comptes cibles. Pour les expériences multi-comptes, vous configurez des rôles IAM dans chaque compte cible et dans le compte d'orchestrateur. Ces rôles IAM nécessitent une relation de confiance entre les comptes cibles et le compte de l'orchestrateur, et entre le compte de l'orchestrateur et. AWS FIS

Les rôles IAM pour les comptes cibles contiennent les autorisations requises pour agir sur les ressources et sont créés pour un modèle d'expérience en ajoutant des configurations de comptes cibles. Vous allez créer un rôle IAM pour le compte d'orchestrateur avec l'autorisation d'assumer les rôles de comptes cibles et d'établir une relation de confiance avec. AWS FIS Ce rôle IAM est utilisé comme modèle `roleArn` d'expérience.

Pour en savoir plus sur le chaînage des rôles, voir Termes et concepts relatifs aux rôles. dans le guide de l'utilisateur d'IAM

Dans l'exemple suivant, vous allez configurer des autorisations pour qu'un compte d'orchestrateur A puisse exécuter un test `aws:ebs:pause-volume-io` dans le compte cible B.

1. Dans le compte B, créez un rôle IAM avec les autorisations requises pour exécuter l'action. Pour connaître les autorisations requises pour chaque action, consultezRéférence des actions. L'exemple suivant montre les autorisations accordées par un compte cible pour exécuter l'action the section called “aws:ebs:pause-volume-io” EBS Pause Volume IO.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:PauseVolumeIO"
            ],
            "Resource": "arn:aws:ec2:region:accountIdB:volume/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Ajoutez ensuite une politique de confiance dans le compte B qui crée une relation de confiance avec le compte A. Choisissez un nom pour le rôle IAM du compte A, que vous allez créer à l'étape 3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
           "Effect": "Allow",
            "Principal": {
                "AWS": "AccountIdA"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringLike":{
                    "sts:ExternalId": "arn:aws:fis:region:accountIdA:experiment/*"
                },
                "ArnEquals": {
```

```
                    "aws:PrincipalArn": "arn:aws:iam::accountIdA:role/role_name"
                }
            }
        }
    ]
}
```

3. Dans le compte A, créez un rôle IAM. Ce nom de rôle doit correspondre au rôle que vous avez spécifié dans la politique de confiance à l'étape 2. Pour cibler plusieurs comptes, vous accordez à l'orchestrateur l'autorisation d'assumer chaque rôle. L'exemple suivant montre les autorisations permettant au compte A d'assumer le compte B. Si vous avez des comptes cibles supplémentaires, vous ajouterez un rôle supplémentaire ARNs à cette politique. Vous ne pouvez avoir qu'un seul ARN de rôle par compte cible.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::accountIdB:role/role_name"
            ]
        }
    ]
}
```

4. Ce rôle IAM pour le compte A est utilisé comme modèle `roleArn` d'expérience. L'exemple suivant montre la politique de confiance requise dans le rôle IAM qui accorde des AWS FIS autorisations pour assumer le compte A, le compte de l'orchestrateur.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "fis.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
```

```
        }
    ]
}
```

Vous pouvez également utiliser Stacksets pour attribuer plusieurs rôles IAM à la fois. Pour l'utiliser CloudFormation StackSets, vous devez configurer les StackSet autorisations nécessaires dans vos AWS comptes. Pour en savoir plus, consultez la section Travailler avec AWS CloudFormation StackSets.

Conditions d'arrêt pour les expériences multi-comptes (facultatif)

Une condition d'arrêt est un mécanisme permettant d'arrêter une expérience si elle atteint un seuil que vous définissez comme une alarme. Pour configurer une condition d'arrêt pour votre expérience multi-comptes, vous pouvez utiliser des alarmes entre comptes. Vous devez activer le partage dans chaque compte cible pour que l'alarme soit accessible au compte de l'orchestrateur à l'aide d'autorisations en lecture seule. Une fois partagées, vous pouvez combiner les statistiques de différents comptes cibles à l'aide de Metric Math. Vous pouvez ensuite ajouter cette alarme comme condition d'arrêt de l'expérience.

Pour en savoir plus sur les tableaux de bord multicomptes, consultez la section Activation de la fonctionnalité multicomptes dans. CloudWatch

Leviers de sécurité pour les expériences multi-comptes (en option)

Les leviers de sécurité sont utilisés pour arrêter toutes les expériences en cours et empêcher le démarrage de nouvelles expériences. Vous pouvez utiliser le levier de sécurité pour empêcher les expériences FIS pendant certaines périodes ou en réponse à des alarmes relatives à l'état de l'application. Chaque AWS compte est doté d'un levier de sécurité Région AWS. Lorsqu'un levier de sécurité est activé, cela a un impact sur toutes les expériences exécutées dans le même compte et dans la même région que le levier de sécurité. Pour arrêter et empêcher les expériences multi-comptes, le levier de sécurité doit être activé dans le même compte et dans la même région que ceux où les expériences sont en cours.

Création d'un modèle d'expérience multi-comptes

Pour savoir comment créer un modèle d'expérience à l'aide du AWS Management Console

Consultez Création d'un modèle d'expérience.

Pour créer un modèle d'expérience à l'aide de la CLI

1. Ouvrez le AWS Command Line Interface
2. Pour créer un test à partir d'un fichier JSON enregistré avec l'option de test de ciblage de compte définie sur `"multi-account"` (par exemple,`my-template.json`), remplacez les valeurs de l'espace réservé par vos propres valeurs, puis exécutez la create-experiment-templatecommande suivante. *italics*

   ```
   aws fis create-experiment-template --cli-input-json file://my-template.json
   ```

   Cela renverra le modèle d'expérience dans la réponse. Copiez le `id` depuis la réponse, qui est l'ID du modèle d'expérience.
3. Exécutez la create-target-account-configurationcommande pour ajouter une configuration de compte cible au modèle d'expérience. Remplacez les valeurs de l'espace réservé *italics* par vos propres valeurs, en utilisant l'`id`étape 2 comme valeur du `--experiment-template-id` paramètre, puis exécutez ce qui suit. Le paramètre `--description` est facultatif. Répétez cette étape pour chaque compte cible.

   ```
   aws fis create-target-account-configuration --experiment-template-id EXTxxxxxxxxx
    --account-id 111122223333 --role-arn arn:aws:iam::111122223333:role/role-name --
   description "my description"
   ```
4. Exécutez la get-target-account-configurationcommande pour récupérer les détails d'une configuration de compte cible spécifique.

   ```
   aws fis get-target-account-configuration --experiment-template-id EXTxxxxxxxxx --
   account-id 111122223333
   ```
5. Une fois que vous avez ajouté toutes les configurations de votre compte cible, vous pouvez exécuter la list-target-account-configurationscommande pour vérifier que les configurations de votre compte cible ont été créées.

   ```
   aws fis list-target-account-configurations --experiment-template-id EXTxxxxxxxxx
   ```

   Vous pouvez également vérifier que vous avez ajouté des configurations de compte cible en exécutant la get-experiment-templatecommande. Le modèle renverra un champ en lecture seule représentant le `targetAccountConfigurationsCount` décompte de toutes les configurations de compte cible sur le modèle d'expérience.

6. Lorsque vous êtes prêt, vous pouvez exécuter le modèle d'expérience à l'aide de la commande start-experiment.

```
aws fis start-experiment --experiment-template-id EXTxxxxxxxxx
```

Mettre à jour la configuration d'un compte cible

Vous pouvez mettre à jour la configuration d'un compte cible existant si vous souhaitez modifier l'ARN ou la description du rôle du compte. Lorsque vous mettez à jour la configuration d'un compte cible, les modifications n'affectent pas les tests en cours utilisant le modèle.

Pour mettre à jour la configuration d'un compte cible à l'aide du AWS Management Console

1. Ouvrez la AWS FIS console à l'adresse https://console.aws.amazon.com/fis/.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Mettre à jour le modèle d'expérience.
4. Dans le panneau latéral, choisissez Étape 3, Configurer l'accès au service.
5. Modifiez les configurations du compte cible, puis choisissez Mettre à jour le modèle d'expérience.
6. Sélectionnez l'étape 5, Réviser et créer.

Pour mettre à jour la configuration d'un compte cible à l'aide de la CLI

Exécutez update-target-account-configurationcommande par commande, en remplaçant les valeurs de l'espace réservé *italics* par vos propres valeurs. Les `--description` paramètres `--role-arn` et sont facultatifs et ne seront pas mis à jour s'ils ne sont pas inclus.

```
aws fis update-target-account-configuration --experiment-template-id EXTxxxxxxxxx
 --account-id 111122223333 --role-arn arn:aws:iam::111122223333:role/role-name --
description "my description"
```

Supprimer une configuration de compte cible

Si vous n'avez plus besoin d'une configuration de compte cible, vous pouvez la supprimer. Lorsque vous supprimez une configuration de compte cible, les expériences en cours utilisant le modèle ne sont pas affectées. L'expérience continue de se dérouler jusqu'à ce qu'elle soit terminée ou arrêtée.

Pour supprimer une configuration de compte cible à l'aide du AWS Management Console

1. Ouvrez la AWS FIS console à l'adresse https://console.aws.amazon.com/fis/.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Mettre à jour.
4. Dans le panneau latéral, choisissez Étape 3, Configurer l'accès au service.
5. Sous Configurations du compte cible, sélectionnez Supprimer pour le rôle ARN du compte cible que vous souhaitez supprimer.
6. Sélectionnez l'étape 5, Réviser et créer.
7. Passez en revue le modèle et choisissez Mettre à jour le modèle d'expérience. Lorsque vous êtes invité à confirmer, entrez `update` et choisissez Mettre à jour le modèle d'expérience.

Pour supprimer une configuration de compte cible à l'aide de la CLI

Exécutez la delete-target-account-configurationcommande en remplaçant les valeurs de l'espace réservé *`italics`* par vos propres valeurs.

```
aws fis update-target-account-configuration --experiment-template-id EXTxxxxxxxxx --
account-id 111122223333
```

Planification des expériences

Avec AWS Fault Injection Service (FIS), vous pouvez réaliser des expériences d'injection de défauts sur vos charges de travail AWS. Ces expériences s'exécutent sur des modèles contenant une ou plusieurs actions à exécuter sur des cibles spécifiées. Vous pouvez désormais planifier vos expériences sous forme de tâche ponctuelle ou de tâches récurrentes de manière native à partir de la console FIS. Outre les règles planifiées, le FIS propose désormais une nouvelle fonctionnalité de planification. FIS s'intègre désormais à EventBridge Scheduler et crée des règles en votre nom. EventBridge Le planificateur est un planificateur sans serveur qui vous permet de créer, d'exécuter et de gérer des tâches à partir d'un service géré centralisé.

> **Important**
>
> Le planificateur d'expériences n' AWS Fault Injection Service est pas disponible dans AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Rubriques

- Création d'un rôle de planificateur
- Création d'un calendrier d'expériences
- Mettre à jour le calendrier d'une expérience
- Désactiver ou supprimer un calendrier d'expérimentation

Création d'un rôle de planificateur

Un rôle d'exécution est un rôle IAM qui AWS FIS , pour interagir avec le EventBridge planificateur et pour permettre au planificateur Event Bridge, de démarrer l'expérience FIS. Vous associez des politiques d'autorisation à ce rôle pour autoriser le EventBridge planificateur à invoquer FIS Experiment. Les étapes suivantes décrivent comment créer un nouveau rôle d'exécution et une politique EventBridge permettant de démarrer une expérience.

Création d'un rôle de planificateur à l'aide de l'interface de ligne de commande AWS

Ce rôle IAM est nécessaire pour qu'Event Bridge puisse planifier une expérience pour le compte du client.

1. Copiez la politique JSON de prise de rôle suivante et enregistrez-la localement sous le nom de `fis-execution-role.json`. Cette politique de confiance permet à EventBridge Scheduler d'assumer le rôle en votre nom.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. À partir de l'interface de ligne de commande AWS (AWS CLI), entrez la commande suivante pour créer un nouveau rôle. `FisSchedulerExecutionRole`Remplacez-le par le nom que vous souhaitez attribuer à ce rôle.

```
aws iam create-role --role-name FisSchedulerExecutionRole --assume-role-policy-
document file://fis-execution-role.json
```

   En cas de succès, vous verrez le résultat suivant :

```
{
    "Role": {
        "Path": "/",
        "RoleName": "FisSchedulerExecutionRole",
        "RoleId": "AROAZL22PDN5A6WKRBQNU",
        "Arn": "arn:aws:iam::123456789012:role/FisSchedulerExecutionRole",
        "CreateDate": "2023-08-24T17:23:05+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "scheduler.amazonaws.com"
                    },
```

```
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

3. Pour créer une nouvelle politique permettant au EventBridge Scheduler d'invoquer l'expérience, copiez le code JSON suivant et enregistrez-le localement sous le nom de. `fis-start-experiment-permissions.json` La politique suivante permet au EventBridge planificateur de lancer l'`fis:StartExperiment`action sur tous les modèles de test de votre compte. Remplacez le `*` à la fin de `"arn:aws:fis:*:*:experiment-template/*"` par l'ID de votre modèle d'expérience si vous souhaitez limiter le rôle à un seul modèle d'expérience.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": [
                "arn:aws:fis:*:*:experiment-template/*",
                "arn:aws:fis:*:*:experiment/*"
            ]
        }
    ]
}
```

4. Exécutez la commande suivante pour créer la nouvelle politique d'autorisation. `FisSchedulerPolicy`Remplacez-le par le nom que vous souhaitez donner à cette politique.

```
aws iam create-policy --policy-name FisSchedulerPolicy --policy-document file://fis-start-experiment-permissions.json
```

En cas de succès, vous verrez le résultat suivant. Notez l'ARN de la politique. Vous utiliserez cet ARN à l'étape suivante pour associer la politique à notre rôle d'exécution.

```
{
    "Policy": {
```

```
        "PolicyName": "FisSchedulerPolicy",
        "PolicyId": "ANPAZL22PDN5ESVUWXLBD",
        "Arn": "arn:aws:iam::123456789012:policy/FisSchedulerPolicy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-08-24T17:34:45+00:00",
        "UpdateDate": "2023-08-24T17:34:45+00:00"
    }
}
```

5. Exécutez la commande suivante pour associer la politique à votre rôle d'exécution. `your-policy-arn`Remplacez-le par l'ARN de la politique que vous avez créée à l'étape précédente. `FisSchedulerExecutionRole`Remplacez-le par le nom de votre rôle d'exécution.

```
aws iam attach-role-policy --policy-arn your-policy-arn --role-name
 FisSchedulerExecutionRole
```

   L'`attach-role-policy`opération ne renvoie pas de réponse sur la ligne de commande.

6. Vous pouvez limiter le planificateur pour qu'il n'exécute que des modèles d' AWS FIS expériences dotés d'une valeur de balise spécifique. Par exemple, la politique suivante accorde l' `fis:StartExperiment`autorisation pour toutes les AWS FIS expériences, mais limite le planificateur à n'exécuter que les modèles d'expériences balisés. `Purpose=Schedule`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment/*"
        },
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment-template/*",
            "Condition": {
                "StringEquals": {
```

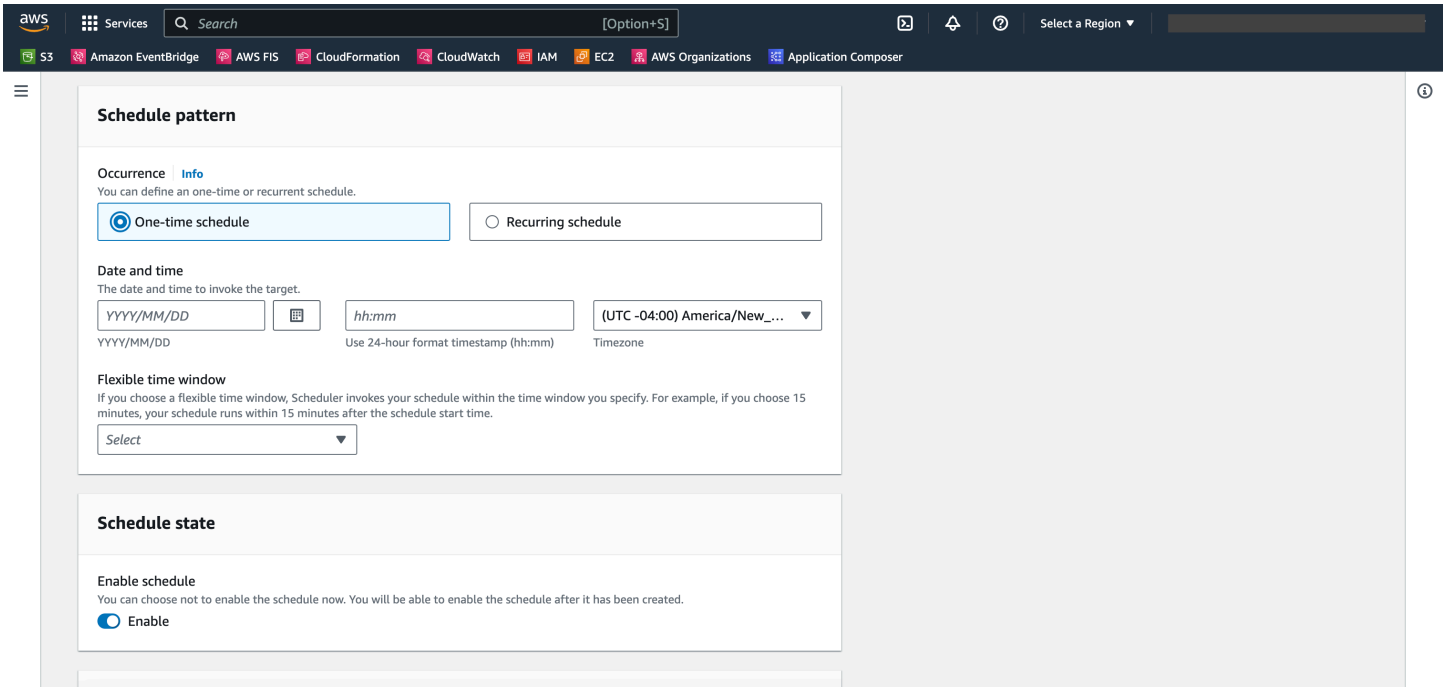
```
                    "aws:ResourceTag/Purpose": "Schedule"
                }
            }
        }
    ]
}
```

Création d'un calendrier d'expériences

Avant de planifier une expérience, vous devez en invoquer une ou plusieurs Composants du modèle d'expérience dans votre planning. Vous pouvez utiliser une ressource AWS existante ou en créer une nouvelle.

Une fois le modèle d'expérience créé, cliquez sur Actions et sélectionnez Planifier l'expérience. Vous serez redirigé vers la page de planification des expériences. Le nom de l'horaire sera renseigné pour vous.

Suivez la section sur le modèle de planification et choisissez un calendrier ponctuel ou récurrent. Renseignez les champs de saisie obligatoires et accédez aux autorisations.



L'état du planning sera activé par défaut. Remarque : si vous désactivez l'état de planification, l'expérience ne sera pas planifiée même si vous créez une planification.

AWS FIS Le planificateur d'expériences est basé sur le planificateur. EventBridge Vous pouvez consulter la documentation pour connaître les différents types de planification pris en charge.

Pour mettre à jour le calendrier à l'aide de la console

1. Ouvrez la AWS FIS console.
2. Dans le volet de navigation de gauche, choisissez Experiment Templates.
3. Choisissez le modèle d'expérience pour lequel vous souhaitez créer le calendrier.
4. Cliquez sur Actions, puis sélectionnez Planifier un test dans le menu déroulant.
   a. Sous Nom du calendrier, le nom est renseigné automatiquement.
   b. Sous Modèle de planification, sélectionnez Planification récurrente.
   c. Sous Type de planification, vous pouvez sélectionner une planification basée sur le taux, voir les types de planification.
   d. Sous Expression du débit, choisissez un taux plus lent que le temps d'exécution de votre expérience, par exemple 5 minutes.
   e. Sous Période, sélectionnez votre fuseau horaire.
   f. Sous Date et heure de début, spécifiez une date et une heure de début.
   g. Sous Date et heure de fin, spécifiez une date et une heure de fin
   h. Sous État du calendrier, activez l'option Activer le calendrier.
   i. Sous Autorisations, sélectionnez Utiliser le rôle existant, puis recherchez `FisSchedulerExecutionRole`.
   j. Choisissez Suivant.
5. Sélectionnez Réviser et créer un calendrier, passez en revue les détails de votre planificateur, puis choisissez Créer un calendrier.

Mettre à jour le calendrier d'une expérience

Vous pouvez mettre à jour le calendrier d'une expérience afin qu'elle ait lieu à la date et à l'heure qui vous conviennent.

Pour mettre à jour l'exécution d'un test à l'aide de la console

1. Ouvrez la console Amazon FIS.
2. Dans le volet de navigation, choisissez Experiment Templates.

3. Choisissez le type de ressource : Modèle d'expérience pour lequel un calendrier a déjà été créé.
4. Cliquez sur l'identifiant de l'expérience pour le modèle. Accédez ensuite à l'onglet des horaires.
5. Vérifiez s'il existe un calendrier associé à l'expérience. Sélectionnez le planning associé et cliquez sur le bouton Mettre à jour le planning.

Désactiver ou supprimer un calendrier d'expérimentation

Pour arrêter l'exécution ou le déroulement d'une expérience selon un calendrier, vous pouvez supprimer ou désactiver la règle. Les étapes suivantes expliquent comment supprimer ou désactiver une exécution d'expérience à l'aide de la AWS console.

Pour supprimer ou désactiver une règle

1. Ouvrez la console Amazon FIS.
2. Dans le volet de navigation, choisissez Experiment Templates.
3. Choisissez le type de ressource : Modèle d'expérience pour lequel un calendrier a déjà été créé.
4. Cliquez sur l'identifiant de l'expérience pour le modèle. Accédez ensuite à l'onglet des horaires.
5. Vérifiez s'il existe un calendrier associé à l'expérience. Sélectionnez le planning associé et cliquez sur le bouton Mettre à jour le planning.
6. Effectuez l'une des actions suivantes :
   a. Pour supprimer le planning, cliquez sur le bouton situé à côté de la règle Supprimer le planning. Tapez `delete` et cliquez sur le bouton Supprimer le calendrier.
   b. Pour désactiver le calendrier, cliquez sur le bouton situé à côté de la règle Désactiver le calendrier. Tapez `disable` et cliquez sur le bouton Désactiver le calendrier.

Leviers de sécurité pour AWS FIS

Les leviers de sécurité sont utilisés pour arrêter toutes les expériences en cours et empêcher le démarrage de nouvelles expériences. Vous pouvez utiliser le levier de sécurité pour empêcher les expériences FIS pendant certaines périodes ou en réponse à des alarmes relatives à l'état de l'application. Chaque AWS compte est doté d'un levier de sécurité Région AWS.

Pour les expériences en cours qui sont arrêtées par le levier de sécurité, vous ne payez que pour la durée de l'action exécutée avant l'arrêt de l'expérience. Les expériences dont le démarrage est empêché n'entraîneront aucun coût. Les sections suivantes fournissent des informations sur la façon de commencer à utiliser les leviers de sécurité.

Rubriques

- Concepts de leviers de sécurité
- Utilisation de leviers de sécurité

Concepts de leviers de sécurité

Un levier de sécurité peut être enclenché ou débrayé.

- En cas de désactivation, les expériences FIS sont autorisées. Par défaut, les leviers de sécurité sont désengagés.
- Si elles sont activées, les expériences en cours sont arrêtées et aucune nouvelle expérience n'est autorisée à démarrer.

Une expérience affectée par un levier de sécurité se terminera dans l'un des états suivants :

- Arrêté, si l'expérience était en cours lorsque le levier de sécurité était enclenché.
- Annulé, si l'expérience a débuté alors que le levier de sécurité était déjà enclenché.

Vous ne pouvez pas reprendre ou relancer un test qui a été arrêté ou annulé. Cependant, vous pouvez démarrer une nouvelle expérience en utilisant le même modèle d'expérience une fois le levier de sécurité débrayé.

Ressources sur les leviers de sécurité

Le levier de sécurité est une ressource définie par le Amazon Resource Name (ARN). Les leviers de sécurité incluent les paramètres suivants :

- Statut, qui est soit engagé, soit désengagé.
- Raison, qui est une chaîne saisie par l'utilisateur pour enregistrer les raisons pour lesquelles l'état du levier de sécurité a été modifié.

Utilisation de leviers de sécurité

Cette section explique comment afficher, activer et désactiver les leviers de sécurité à l'aide de la AWS FIS console ou de la ligne de commande.

Visualisation d'un levier de sécurité

Vous pouvez consulter l'état de votre levier de sécurité pour votre compte et votre région en suivant les étapes ci-dessous.

Pour visualiser un levier de sécurité à l'aide de la console

1. Ouvrez la AWS FIS console
2. Dans le volet de navigation, sélectionnez Experiments.
3. Si le levier de sécurité est activé, vous verrez une bannière d'alerte en haut de la page. S'il n'y a pas de bandeau d'alerte, le levier de sécurité est débrayé.

Pour visualiser un levier de sécurité à l'aide de la CLI

- Utilisez la commande suivante :

```
aws fis get-safety-lever --id "default"
```

Un levier de sécurité peut se trouver dans l'un des états suivants :

- Désengagé - Le levier de sécurité n'a aucune incidence sur les expériences. Les expériences peuvent se dérouler librement. Les leviers de sécurité sont désactivés par défaut.

- Engagement - Le levier de sécurité passe de débrayé à enclenché. Il se peut qu'il y ait encore des expériences qui n'ont pas encore été arrêtées. Le levier de sécurité ne peut pas être changé dans cet état.
- Engagé - Le levier de sécurité est actif et aucune expérience n'est en cours. Toute nouvelle expérience qui tente de démarrer alors que le levier de sécurité est enclenché sera annulée.

Activation d'un levier de sécurité

Pour actionner un levier de sécurité à l'aide de la console

1. Ouvrez la AWS FIS console
2. Dans le volet de navigation, sélectionnez Experiments.
3. Cliquez sur le bouton Arrêter toutes les expériences.
4. Entrez la raison pour laquelle vous avez actionné le levier de sécurité.
5. Choisissez Confirmer.

Pour activer un levier de sécurité à l'aide de la CLI

- Utilisez la commande suivante de l'. Remplissez le champ du motif avec votre propre réponse.

```
aws fis  update-safety-lever-state --id "default" --state
 "status=engaged,reason=xxxxx"
```

Débranchement d'un levier de sécurité

Pour désengager un levier de sécurité à l'aide de la console

1. Ouvrez la AWS FIS console
2. Dans le volet de navigation, sélectionnez Experiments.
3. Cliquez sur le bouton Désenclencher le levier de sécurité.
4. Entrez le motif du désengagement du levier de sécurité.
5. Choisissez Confirmer.

Pour déconnecter un levier de sécurité à l'aide de la CLI

- Utilisez la commande suivante :

```
aws fis  update-safety-lever-state --id "default" --state
 "status=disengaged,reason=recovered"
```

Surveillance des AWS expériences FIS

Vous pouvez utiliser les outils suivants pour suivre la progression et l'impact de vos expériences avec le service d'injection de AWS défauts (AWS FIS).

AWS console FIS et AWS CLI

Utilisez la console AWS FIS ou le AWS CLI pour suivre la progression d'une expérience en cours. Vous pouvez consulter le statut de chaque action dans le test, ainsi que les résultats de chaque action. Pour de plus amples informations, veuillez consulter the section called “Afficher vos expériences”.

CloudWatch mesures d'utilisation et alarmes

Utilisez les statistiques CloudWatch d'utilisation pour obtenir une visibilité sur l'utilisation des ressources par votre compte. AWS Les métriques d'utilisation du FIS correspondent aux quotas AWS de service. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d'un quota de service. Pour de plus amples informations, veuillez consulter Surveiller en utilisant CloudWatch.

Vous pouvez également créer des conditions d'arrêt pour vos expériences AWS FIS en créant des CloudWatch alarmes qui définissent le moment où une expérience dépasse les limites. Lorsque l'alarme est déclenchée, l'expérience s'arrête. Pour de plus amples informations, veuillez consulter Conditions d'arrêt. Pour plus d'informations sur la création d' CloudWatch alarmes, consultez les sections Création CloudWatch d'une alarme basée sur un seuil statique et Création CloudWatch d'une alarme basée sur la détection d'anomalies dans le guide de l' CloudWatch utilisateur Amazon.

AWS Enregistrement des expériences FIS

Activez la journalisation des expériences pour capturer des informations détaillées sur votre expérience au fur et à mesure de son exécution. Pour plus d'informations, voir Enregistrement des expériences.

Expérimentez les événements de changement d'état

Amazon vous EventBridge permet de répondre automatiquement aux événements du système ou aux modifications des ressources. AWS Le FIS émet une notification lorsque l'état d'une expérience change. Vous pouvez créer des règles pour les événements qui vous intéressent et qui spécifient l'action automatique à effectuer lorsqu'un événement correspond à une règle. Par

exemple, envoyer une notification à une rubrique Amazon SNS ou invoquer une fonction Lambda. Pour de plus amples informations, veuillez consulter Surveiller en utilisant EventBridge.

CloudTrail journaux

AWS CloudTrail À utiliser pour capturer des informations détaillées sur les appels passés à l'API AWS FIS et les stocker sous forme de fichiers journaux dans Amazon S3. CloudTrail enregistre également les appels passés au service APIs pour les ressources sur lesquelles vous effectuez des tests. Vous pouvez utiliser ces CloudTrail journaux pour déterminer quels appels ont été passés, l'adresse IP source d'où provient l'appel, qui a effectué l'appel, quand l'appel a été passé, etc.

AWS Notifications du tableau de bord de santé

AWS Health fournit une visibilité continue sur les performances de vos ressources et sur la disponibilité de vos AWS services et comptes. Lorsque vous démarrez une expérience, le AWS FIS envoie une notification à votre AWS Health Dashboard. La notification est présente pendant toute la durée de l'expérience dans chaque compte contenant des ressources ciblées dans une expérience, y compris les expériences multi-comptes. Les expériences multi-comptes comportant uniquement des actions n'incluant pas de cibles, telles que `aws:ssm:start-automation-execution` et`aws:fis:wait`, n'émettent pas de notification. Les informations relatives au rôle utilisé pour autoriser l'expérience seront répertoriées sous Ressources concernées. Pour en savoir plus sur le tableau de bord AWS Health, consultez le tableau de bord AWS Health dans le guide de l'utilisateur d'AWS Health.

> **Note**
>
> AWS Health organise des événements dans la mesure du possible.

Surveillez les statistiques d'utilisation du AWS FIS à l'aide d'Amazon CloudWatch

Vous pouvez utiliser Amazon CloudWatch pour surveiller l'impact des expériences AWS FIS sur les cibles. Vous pouvez également surveiller votre utilisation AWS du FIS.

Pour plus d'informations sur l'affichage de l'état d'une expérience, consultezAfficher vos expériences.

Surveiller AWS les expériences FIS

Lorsque vous planifiez vos expériences AWS FIS, identifiez les CloudWatch métriques que vous pouvez utiliser pour identifier la base de référence ou « l'état d'équilibre » pour les types de ressources cibles pour l'expérience. Une fois que vous avez démarré une expérience, vous pouvez surveiller ces CloudWatch mesures pour les cibles sélectionnées via le modèle d'expérience.

Pour plus d'informations sur les CloudWatch métriques disponibles pour un type de ressource cible pris en charge par le AWS FIS, consultez les rubriques suivantes :

- Surveillez vos instances à l'aide de CloudWatch
- CloudWatch Métriques Amazon ECS
- Surveillance des métriques Amazon RDS à l'aide de CloudWatch
- Surveillance des métriques Run Command à l'aide de CloudWatch

AWS Métriques d'utilisation du FIS

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.

AWS Les métriques d'utilisation du FIS correspondent aux quotas AWS de service. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d'un quota de service. Pour plus d'informations sur les CloudWatch alarmes, consultez le guide de CloudWatch l'utilisateur Amazon.

AWS FIS publie la métrique suivante dans l'espace de noms AWS/Usage.

Métrique	Description
`ResourceCount`	Nombre total des ressources spécifiées exécutées sur votre compte. La ressource est définie par les dimensions associées à la métrique.

Les dimensions suivantes sont utilisées pour affiner les métriques d'utilisation publiées par le AWS FIS.

Dimension	Description
`Service`	Nom du AWS service contenant la ressource. Pour les métriques d'utilisation du AWS FIS, la valeur de cette dimension est`FIS`.
`Type`	Type d'entité faisant l'objet d'un rapport. Actuellement, la seule valeur valide pour les métriques d'utilisation du AWS FIS est`Resource`.
`Resource`	Type de ressource en cours d'exécution. Les valeurs possibles concernent `Experimen tTemplates` les modèles d'expériences et `ActiveExperiments` les expériences actives.
`Class`	Cette dimension est réservée à une utilisation future.

Surveillez les expériences AWS FIS à l'aide d'Amazon EventBridge

Lorsque l'état d'une expérience change, le AWS FIS émet une notification. Ces notifications sont mises à disposition sous forme d'événements via Amazon EventBridge (anciennement CloudWatch Events). AWS La FIS diffuse ces événements dans la mesure du possible. Les événements sont diffusés EventBridge en temps quasi réel.

Avec EventBridge, vous pouvez créer des règles qui déclenchent des actions programmatiques en réponse à un événement. Par exemple, vous pouvez configurer une règle qui invoque une rubrique SNS pour envoyer une notification par e-mail ou qui invoque une fonction Lambda pour effectuer une action.

Pour plus d'informations EventBridge, consultez Getting started with Amazon EventBridge dans le guide de EventBridge l'utilisateur Amazon.

Voici la syntaxe d'un événement de changement d'état d'une expérience :

```
{
```

```
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "FIS Experiment State Change",
    "source": "aws.fis",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "region",
    "resources": [
        "arn:aws:fis:region:account_id:experiment/experiment-id"
    ],
    "detail": {
        "experiment-id": "EXPaBCD1efg2HIJkL3",
        "experiment-template-id": "EXTa1b2c3de5f6g7h",
        "new-state": {
            "status": "new_value",
            "reason": "reason_string"
        },
        "old-state": {
           "status": "old_value",
           "reason": "reason_string"
        }
    }
}
```

`experiment-id`

ID de l'expérience dont l'état a changé.

`experiment-template-id`

ID du modèle d'expérience utilisé par l'expérience.

`new_value`

Le nouvel état de l'expérience. Les valeurs possibles sont :

- `completed`
- `failed`
- `initiating`
- `running`
- `stopped`
- `stopping`

`old_value`

État précédent de l'expérience. Les valeurs possibles sont :

- `initiating`
- `pending`
- `running`
- `stopping`

Enregistrement des expériences pour AWS FIS

Vous pouvez utiliser la journalisation des expériences pour saisir des informations détaillées sur votre expérience au fur et à mesure de son exécution.

La journalisation des expériences vous est facturée en fonction des coûts associés à chaque type de destination de journal. Pour plus d'informations, consultez les CloudWatch tarifs Amazon (sous Paid Tier, Logs, Vended Logs) et Amazon S3 Pricing.

Autorisations

Vous devez accorder des autorisations AWS FIS pour envoyer des journaux à chaque destination de journal que vous configurez. Pour plus d'informations, consultez les informations suivantes dans le guide de l'utilisateur d'Amazon CloudWatch Logs :

- Logs envoyés à CloudWatch Logs
- Journaux envoyés à Amazon S3

Schéma du journal

Le schéma utilisé pour la journalisation des expériences est le suivant. La version actuelle du schéma est 2. Les champs pour `details` dépendent de la valeur de`log_type`. Les champs pour `resolved_targets` dépendent de la valeur de`target_type`. Pour de plus amples informations, veuillez consulter the section called “Exemples d'enregistrements de journal”.

```
{
    "id": "EXP123abc456def789",
    "log_type": "experiment-start | target-resolution-start | target-resolution-detail
 | target-resolution-end | action-start | action-error | action-end | experiment-end",
    "event_timestamp": "yyyy-mm-ddThh:mm:ssZ",
```

```
    "version": "2",
    "details": {
        "account_id":"123456789012",
        "action_end_time": "yyyy-mm-ddThh:mm:ssZ",
        "action_id": "String",
        "action_name": "String",
        "action_start_time": "yyyy-mm-ddThh:mm:ssZ",
        "action_state": {
            "status": "pending | initiating | running | completed | cancelled |
 stopping | stopped | failed",
            "reason": "String"
        },
        "action_targets": "String to string map",
        "error_information": "String",
        "experiment_end_time": "yyyy-mm-ddThh:mm:ssZ",
        "experiment_state": {
            "status": "pending | initiating | running | completed | stopping | stopped
 | failed",
            "reason": "String"
        },
        "experiment_start_time": "yyyy-mm-ddThh:mm:ssZ",
        "experiment_template_id": "String",
        "page": Number,
        "parameters": "String to string map",
        "resolved_targets": [
            {
               "field": "value"
            }
        ],
        "resolved_targets_count": Number,
        "status": "failed | completed",
        "target_name": "String",
        "target_resolution_end_time": "yyyy-mm-ddThh:mm:ssZ",
        "target_resolution_start_time": "yyyy-mm-ddThh:mm:ssZ",
        "target_type": "String",
        "total_pages": Number,
        "total_resolved_targets_count": Number

    }
}
```

Notes de mise à jour

- La version 2 introduit :
  - Le `target_type` champ fait passer le `resolved_targets` champ d'une liste d'objets ARNs à une liste d'objets. Les champs valides pour l'`resolved_targets`objet dépendent de la valeur de`target_type`, qui est le type de ressource des cibles.
  - Les types `target-resolution-detail` d'événements `action-error` et qui ajoutent le `account_id` champ.
- La version 1 est la version initiale.

Enregistrer les destinations

AWS FIS prend en charge la livraison de journaux vers les destinations suivantes :

- Un compartiment Amazon S3
- Un groupe de CloudWatch journaux Amazon Logs

Livraison du journal S3

Les journaux sont livrés à l'emplacement suivant.

```
bucket-and-optional-prefix/AWSLogs/account-id/fis/region/experiment-
id/YYYY/MM/DD/account-id_awsfislogs_region_experiment-id_YYYYMMDDHHMMZ_hash.log
```

Plusieurs minutes peuvent s'écouler avant que les journaux ne soient livrés au bucket.

CloudWatch Logs et livraison de journaux

Les journaux sont transmis à un flux de journauxnamed /aws/fis/*experiment-id*.

Les journaux sont envoyés au groupe de journaux en moins d'une minute.

Exemples d'enregistrements de journal

Voici des exemples d'enregistrements de journal pour une expérience qui exécute l'aws:ec2:reboot-instancesaction sur une EC2 instance sélectionnée au hasard.

Enregistrements

- démarrage de l'expérience
- target-resolution-start
- target-resolution-detail
- target-resolution-end
- action-start
- fin de l'action
- action-erreur
- fin de l'expérience

démarrage de l'expérience

Voici un exemple d'enregistrement pour l'`experiment-start`événement.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "experiment-start",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "experiment_template_id": "EXTCDh1M8HHkhxoaQ",
        "experiment_start_time": "2023-05-31T18:50:43Z"
    }
}
```

target-resolution-start

Voici un exemple d'enregistrement pour l'`target-resolution-start`événement.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "target-resolution-start",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "target_resolution_start_time": "2023-05-31T18:50:45Z",
```

```
        "target_name": "EC2InstancesToReboot"
    }
}
```

target-resolution-detail

Voici un exemple d'enregistrement pour l'`target-resolution-detail`événement. Si la résolution cible échoue, l'enregistrement inclut également le `error_information` champ.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "target-resolution-detail",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "target_resolution_end_time": "2023-05-31T18:50:45Z",
        "target_name": "EC2InstancesToReboot",
        "target_type": "aws:ec2:instance",
        "account_id": "123456789012",
        "resolved_targets_count": 2,
        "status": "completed"

    }
}
```

target-resolution-end

Si la résolution cible échoue, l'enregistrement inclut également le `error_information` champ. S'il `total_pages` est supérieur à 1, le nombre de cibles résolues a dépassé la limite de taille pour un enregistrement. Des `target-resolution-end` enregistrements supplémentaires contiennent les cibles résolues restantes.

Voici un exemple d'enregistrement de l'`target-resolution-end`événement associé à une EC2 action.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "target-resolution-end",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
```

```
    "details": {
        "target_resolution_end_time": "2023-05-31T18:50:46Z",
        "target_name": "EC2InstanceToReboot",
        "target_type": "aws:ec2:instance",
        "resolved_targets": [
            {
                "arn": "arn:aws:ec2:us-east-1:123456789012:instance/
i-0f7ee2abffc330de5"
            }
        ],
        "page": 1,
        "total_pages": 1
    }
}
```

Voici un exemple d'enregistrement de l'`target-resolution-end`événement associé à une action EKS.

```
{
    "id": "EXP24YfiucfyVPJpEJn",
    "log_type": "target-resolution-end",
    "event_timestamp": "2023-05-31T18:50:45Z",
    "version": "2",
    "details": {
        "target_resolution_end_time": "2023-05-31T18:50:46Z",
        "target_name": "myPods",
        "target_type": "aws:eks:pod",
        "resolved_targets": [
            {
                "pod_name": "example-696fb6498b-sxhw5",
                "namespace": "default",
                "cluster_arn": "arn:aws:eks:us-east-1:123456789012:cluster/fis-demo-
cluster",
                "target_container_name": "example"
            }
        ],
        "page": 1,
        "total_pages": 1
    }
}
```

action-start

Voici un exemple d'enregistrement pour l'`action-start`événement. Si le modèle d'expérience spécifie les paramètres de l'action, l'enregistrement inclut également le `parameters` champ.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "action-start",
    "event_timestamp": "2023-05-31T18:50:56Z",
    "version": "2",
    "details": {
        "action_name": "Reboot",
        "action_id": "aws:ec2:reboot-instances",
        "action_start_time": "2023-05-31T18:50:56Z",
        "action_targets": {"Instances":"EC2InstancesToReboot"}
    }
}
```

action-erreur

Voici un exemple d'enregistrement pour l'`action-error`événement. Cet événement n'est renvoyé qu'en cas d'échec d'une action. Il est renvoyé pour chaque compte sur lequel l'action échoue.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "action-error",
    "event_timestamp": "2023-05-31T18:50:56Z",
    "version": "2",
    "details": {
        "action_name": "pause-io",
        "action_id": "aws:ebs:pause-volume-io",
        "account_id": "123456789012",
        "action_state": {
            "status": "failed",
            "reason":"Unable to start Pause Volume IO. Target volumes must be attached
 to an instance type based on the Nitro system. VolumeId(s): [vol-1234567890abcdef0]:"
        }
    }
}
```

fin de l'action

Voici un exemple d'enregistrement pour l'`action-end`événement.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "action-end",
    "event_timestamp": "2023-05-31T18:50:56Z",
    "version": "2",
    "details": {
        "action_name": "Reboot",
        "action_id": "aws:ec2:reboot-instances",
        "action_end_time": "2023-05-31T18:50:56Z",
        "action_state": {
            "status": "completed",
            "reason": "Action was completed."
        }
    }
}
```

fin de l'expérience

Voici un exemple d'enregistrement pour l'`experiment-end`événement.

```
{
    "id": "EXPhjAXCGY78HV2a4A",
    "log_type": "experiment-end",
    "event_timestamp": "2023-05-31T18:50:57Z",
    "version": "2",
    "details": {
        "experiment_end_time": "2023-05-31T18:50:57Z",
        "experiment_state": {
            "status": "completed",
            "reason": "Experiment completed"
        }
    }
}
```

Activer la journalisation des expériences

La journalisation des expériences est désactivée par défaut. Pour recevoir des journaux d'expériences pour une expérience, vous devez créer l'expérience à partir d'un modèle d'expérience avec la journalisation activée. La première fois que vous exécutez un test configuré pour utiliser une destination qui n'a pas été utilisée auparavant pour la journalisation, nous retardons le test pour configurer la livraison du journal vers cette destination, ce qui prend environ 15 secondes.

Pour activer la journalisation des expériences à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Mettre à jour le modèle d'expérience.
4. Pour les journaux, configurez les options de destination. Pour envoyer des journaux vers un compartiment S3, choisissez Envoyer vers un compartiment Amazon S3 et entrez le nom et le préfixe du compartiment. Pour envoyer des CloudWatch journaux à Logs, choisissez Send to CloudWatch Logs et entrez le groupe de journaux.
5. Choisissez Mettre à jour le modèle d'expérience.

Pour activer la journalisation des expériences à l'aide du AWS CLI

Utilisez la update-experiment-templatecommande et spécifiez une configuration de journal.

Désactiver la journalisation des expériences

Si vous ne souhaitez plus recevoir les journaux de vos expériences, vous pouvez désactiver la journalisation des expériences.

Pour désactiver la journalisation des expériences à l'aide de la console

1. Ouvrez la console AWS FIS à https://console.aws.amazon.com/fis/l'adresse.
2. Dans le volet de navigation, sélectionnez Modèles d'expériences.
3. Sélectionnez le modèle d'expérience, puis choisissez Actions, Mettre à jour le modèle d'expérience.
4. Pour les journaux, décochez Envoyer vers un compartiment Amazon S3 et Envoyer vers CloudWatch les journaux.
5. Choisissez Mettre à jour le modèle d'expérience.

Pour désactiver la journalisation des expériences à l'aide du AWS CLI

Utilisez la update-experiment-templatecommande et spécifiez une configuration de journal vide.

Enregistrez les appels d'API avec AWS CloudTrail

AWS Le service d'injection de défauts (AWS FIS) est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans AWS FIS. CloudTrail capture tous les appels d'API pour AWS FIS sous forme d'événements. Les appels capturés incluent les appels provenant de la console AWS FIS et les appels de code vers les opérations de l'API AWS FIS. Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour AWS FIS. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été faite à AWS FIS, l'adresse IP à partir de laquelle la demande a été faite, qui a fait la demande, quand elle a été faite et des détails supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Utiliser CloudTrail

CloudTrail est activé sur votre compte Compte AWS lorsque vous créez le compte. Lorsqu'une activité se produit dans AWS FIS, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre Compte AWS. Pour plus d'informations, consultez la section Affichage des événements à l'aide de l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre région Compte AWS, y compris des événements pour la AWS FIS, créez un parcours. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un parcours dans la console, celui-ci s'applique à toutes les AWS régions. Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour plus d'informations, consultez les ressources suivantes :

- Créez un parcours pour votre AWS compte
- CloudTrail Services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les actions AWS FIS sont enregistrées CloudTrail et documentées dans la référence de l'API du service d'injection de AWS défauts. Pour les actions d'expérimentation effectuées sur une ressource cible, consultez la documentation de référence de l'API pour le service propriétaire de la ressource. Par exemple, pour les actions effectuées sur une EC2 instance Amazon, consultez le Amazon EC2 API Reference.

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 les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour plus d'informations, consultez la section Élément userIdentity CloudTrail .

Comprendre les AWS entrées du fichier journal FIS

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

Voici un exemple d'entrée de CloudTrail journal pour un appel à l'`StopExperiment`action AWS FIS.


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AKIAIOSFODNN7EXAMPLE:jdoe",
    "arn": "arn:aws:sts::111122223333:assumed-role/example/jdoe",
    "accountId": "111122223333",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
```

```
        "arn": "arn:aws:iam::111122223333:role/example",
        "accountId": "111122223333",
        "userName": "example"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2020-12-03T09:40:42Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2020-12-03T09:44:20Z",
  "eventSource": "fis.amazonaws.com",
  "eventName": "StopExperiment",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.51.100.25",
  "userAgent": "Boto3/1.22.9 Python/3.8.13 Linux/5.4.186-113.361.amzn2int.x86_64
Botocore/1.25.9",
  "requestParameters": {
    "clientToken": "1234abc5-6def-789g-012h-ijklm34no56p",
    "experimentTemplateId": "ABCDE1fgHIJkLmNop",
    "tags": {}
  },
  "responseElements": {
    "experiment": {
      "actions": {
        "exampleAction1": {
          "actionId": "aws:ec2:stop-instances",
          "duration": "PT10M",
          "state": {
            "reason": "Initial state",
            "status": "pending"
          },
          "targets": {
            "Instances": "exampleTag1"
          }
        },
        "exampleAction2": {
          "actionId": "aws:ec2:stop-instances",
          "duration": "PT10M",
          "state": {
            "reason": "Initial state",
            "status": "pending"
          },
```

```
          "targets": {
            "Instances": "exampleTag2"
          }
        }
      },
      "creationTime": 1605788649.95,
      "endTime": 1606988660.846,
      "experimentTemplateId": "ABCDE1fgHIJkLmNop",
      "id": "ABCDE1fgHIJkLmNop",
      "roleArn": "arn:aws:iam::111122223333:role/AllowFISActions",
      "startTime": 1605788650.109,
      "state": {
        "reason": "Experiment stopped",
        "status": "stopping"
      },
      "stopConditions": [
        {
          "source": "aws:cloudwatch:alarm",
          "value": "arn:aws:cloudwatch:us-east-1:111122223333:alarm:example"
        }
      ],
      "tags": {},
      "targets": {
        "ExampleTag1": {
          "resourceTags": {
            "Example": "tag1"
          },
          "resourceType": "aws:ec2:instance",
          "selectionMode": "RANDOM(1)"
        },
        "ExampleTag2": {
          "resourceTags": {
            "Example": "tag2"
          },
          "resourceType": "aws:ec2:instance",
          "selectionMode": "RANDOM(1)"
        }
      }
    }
  },
  "requestID": "1abcd23e-f4gh-567j-klm8-9np01q234r56",
  "eventID": "1234a56b-c78d-9e0f-g1h2-34jk56m7n890",
  "readOnly": false,
  "eventType": "AwsApiCall",
```

```
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"

}
```

Voici un exemple d'entrée de CloudTrail journal pour une action d'API invoquée par le AWS FIS dans le cadre d'une expérience incluant l'action `aws:ssm:send-command` AWS FIS. L'`userIdentity`élément reflète une demande faite avec des informations d'identification temporaires obtenues en assumant un rôle. Le nom du rôle assumé apparaît dans`userName`. L'identifiant de l'expérience, EXP21n T17 WMz A6DNugz, apparaît dans et en `principalId` tant que partie intégrante de l'ARN du rôle assumé.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROATZZZ4JPIXUEXAMPLE:EXP21nT17WMzA6dnUgz",
        "arn": "arn:aws:sts::111122223333:assumed-role/AllowActions/
EXP21nT17WMzA6dnUgz",
        "accountId": "111122223333",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROATZZZ4JPIXUEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/AllowActions",
                "accountId": "111122223333",
                "userName": "AllowActions"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-05-30T13:23:19Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "fis.amazonaws.com"
    },
    "eventTime": "2022-05-30T13:23:19Z",
    "eventSource": "ssm.amazonaws.com",
    "eventName": "ListCommands",
    "awsRegion": "us-east-2",
```

```
    "sourceIPAddress": "fis.amazonaws.com",
    "userAgent": "fis.amazonaws.com",
    "requestParameters": {
        "commandId": "51dab97f-489b-41a8-a8a9-c9854955dc65"
    },
    "responseElements": null,
    "requestID": "23709ced-c19e-471a-9d95-cf1a06b50ee6",
    "eventID": "145fe5a6-e9d5-45cc-be25-b7923b950c83",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Résolution des problèmes AWS FIS

Pour résoudre les erreurs, AWS FIS renvoie les erreurs détaillées à partir des journaux d'expériences de l'`GetExperiment`API et du FIS. Les erreurs sont renvoyées dans le cadre de l'état de l'expérience en cas d'échec de l'expérience. Lorsque plusieurs actions échouent, la première action échouée est renvoyée sous forme d'erreur expérimentale. Vous pouvez consulter vos journaux d'expériences FIS pour détecter toute autre erreur. Pour savoir comment enregistrer et surveiller les AWS FIS expériences, voirSurveillance des AWS expériences FIS.

Selon le type de panne, l'un des messages d'erreur suivants peut s'afficher :

- Motif : description détaillée de la panne spécifique. Les valeurs de raison ne doivent pas être utilisées pour l'automatisation, car elles sont sujettes à modification.
- Code : type de panne. Les valeurs de code ne doivent pas être utilisées pour l'automatisation, car elles sont sujettes à modification, sauf indication contraire dans le tableau ci-dessous.
- Emplacement : contexte de la section du modèle d'expérience qui a échoué, telle que l'action ou la cible.
- Identifiant du compte : le AWS compte sur lequel l'échec s'est produit.

Codes d'erreur

Code d'erreur	Description du code
`ConfigurationFailure`	L'action, la cible, l'expérience ou le journal n'est pas configuré correctement. Vérifiez l'erreur `location` et assurez-vous que les paramètres et les configurations sont corrects.
`DependentServiceFailure`	Un autre AWS service est tombé en panne. Réessayez de lancer l'expérience.
`InternalFailure`	Une erreur interne s'est produite lors de l'exécution de l'expérience. Vous pouvez automatiser en fonction de ce code d'erreur.

Code d'erreur	Description du code
`InvalidTarget`	Une cible n'a pas pu être résolue pendant la résolution de la cible ou au début d'une action. Cela peut être dû à l'une des raisons suivantes : • La cible n'existe pas, par exemple si elle a été supprimée ou si l'ARN est incorrect. • Il existe un tag pour votre cible qui ne résout aucune ressource. • Il existe une action qui n'est pas liée à une cible. Pour résoudre le problème, consultez vos journaux afin d'identifier les cibles non résolues. Vérifiez que toutes les actions sont liées à des cibles et que votre identifiant de ressource ou vos balises existent et n'ont pas été mal orthographiés.

Code d'erreur	Description du code
`AuthorizationFailure`	Deux causes principales peuvent expliquer l'échec d'une expérience en raison d'erreurs d'autorisation : • Le rôle IAM que vous ciblez ne dispose pas des autorisations appropriées pour résoudre des cibles ou agir sur vos ressources. Pour corriger cette erreur, passez en revue les autorisations requises pour vos actions dans la référence des actions FIS et ajoutez-les à votre rôle IAM d'expérience. • La création d'un rôle AWS lié au service (SLR) pour FIS a été refusée par une politique de contrôle des services (SCP) de votre organisation. Le FIS utilise le SLR pour gérer la surveillance et la sélection des ressources pour les expériences. Pour de plus amples informations, veuillez consulter Autorisations de rôle liées au service pour FIS AWS.
`QuotaExceededFailure`	Le quota pour le type de ressource a été dépassé. Pour déterminer si le quota peut être augmenté, voirQuotas et limites pour le service d'injection de AWS défauts.

Sécurité dans le service d'injection de AWS défauts

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez de centres de données et d'architectures 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 cela comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de AWS conformité Programmes de de conformité. Pour en savoir plus sur les programmes de conformité qui s'appliquent au service d'injection de AWS défauts, voir AWS Services concernés par programme de conformitéAWS .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation du AWS FIS. Les rubriques suivantes expliquent comment configurer le AWS FIS pour répondre à vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources AWS FIS.

Table des matières

- Protection des données dans le cadre du service d'injection de AWS défauts
- Gestion des identités et des accès pour le service d'injection de AWS défauts
- Sécurité de l'infrastructure dans le service d'injection de AWS défauts
- Accédez au AWS FIS à l'aide d'un point de terminaison VPC d'interface ()AWS PrivateLink

Protection des données dans le cadre du service d'injection de AWS défauts

Le modèle de responsabilité AWS partagée Le modèle s'applique à la protection des données dans le service d'injection de AWS défauts. 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 Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

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 AWS Identity and Access Management (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 AWS FIS ou autre à Services AWS l'aide de la console, de l'API ou AWS SDKs. 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.

Chiffrement au repos

AWS FIS chiffre toujours vos données au repos. Les données du AWS FIS sont chiffrées au repos à l'aide d'un chiffrement transparent côté serveur. Cela réduit la lourdeur opérationnelle et la complexité induites par la protection des données sensibles. Le chiffrement au repos vous permet de créer des applications sensibles en matière de sécurité qui sont conformes aux exigences réglementaires et de chiffrement.

Chiffrement en transit

AWS Le FIS chiffre les données en transit entre le service et d'autres services intégrés AWS . Toutes les données qui transitent entre le AWS FIS et les services intégrés sont cryptées à l'aide du protocole TLS (Transport Layer Security). Pour plus d'informations sur les autres AWS services intégrés, consultezSoutenu Services AWS.

Gestion des identités et des accès pour le service d'injection de AWS défauts

AWS Identity and Access Management (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs IAM contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les ressources AWS FIS. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Table des matières

- Public ciblé
- Authentification par des identités
- Gestion des accès à l'aide de politiques
- Comment fonctionne le service d'injection de AWS défauts avec IAM
- AWS Exemples de politiques relatives au service d'injection de défauts
- Utiliser des rôles liés à un service pour le service d'injection de AWS défauts
- AWS politiques gérées pour le service d'injection de AWS défauts

Public ciblé

La façon dont vous utilisez AWS Identity and Access Management (IAM) varie en fonction du travail que vous effectuez dans AWS FIS.

Utilisateur du service : si vous utilisez le service AWS FIS pour effectuer votre travail, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Au fur et à mesure que vous utilisez de plus en plus de fonctionnalités AWS FIS pour effectuer votre travail, vous aurez peut-être besoin d'autorisations supplémentaires. En comprenant bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur.

Administrateur du service — Si vous êtes responsable des ressources du AWS FIS dans votre entreprise, vous avez probablement un accès complet au AWS FIS. C'est à vous de déterminer les fonctionnalités et les ressources du AWS FIS auxquelles les utilisateurs de votre service doivent accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d’IAM.

Administrateur IAM — Si vous êtes administrateur IAM, vous souhaiterez peut-être en savoir plus sur la manière dont vous pouvez rédiger des politiques pour gérer l'accès au AWS FIS.

Authentification par des identités

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié (connecté à AWS) en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en AWS tant qu'identité fédérée en utilisant les informations d'identification fournies par le biais d'une source d'identité. AWS IAM Identity Center Les utilisateurs (IAM Identity Center), l'authentification unique de votre entreprise et vos informations d'identification Google ou Facebook sont des exemples d'identités fédérées. Lorsque vous vous connectez avec une identité fédérée, votre administrateur aura précédemment configuré une fédération d’identités avec des rôles IAM. Lorsque vous accédez à AWS l'aide de la fédération, vous assumez indirectement un rôle.

Selon le type d'utilisateur que vous êtes, vous pouvez vous connecter au portail AWS Management Console ou au portail AWS d'accès. Pour plus d'informations sur la connexion à AWS, consultez Comment vous connecter à votre compte Compte AWS dans le guide de Connexion à AWS l'utilisateur.

Si vous y accédez AWS par programmation, AWS fournit un kit de développement logiciel (SDK) et une interface de ligne de commande (CLI) pour signer cryptographiquement vos demandes à l'aide de vos informations d'identification. Si vous n'utilisez pas d' AWS outils, vous devez signer vous-même les demandes. Pour plus d'informations sur l'utilisation de la méthode recommandée pour signer des demandes vous-même, consultez AWS Signature Version 4 pour les demandes d'API dans le Guide de l'utilisateur IAM.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être fournir des informations de sécurité supplémentaires. Par exemple, il vous AWS recommande d'utiliser l'authentification multifactorielle (MFA) pour renforcer la sécurité de votre compte. Pour plus d'informations, consultez Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center et Authentification multifactorielle AWS dans IAM dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui vous imposent de vous connecter en tant qu'utilisateur racine, consultez Tâches nécessitant des informations d'identification d'utilisateur racine dans le Guide de l'utilisateur IAM.

Identité fédérée

La meilleure pratique consiste à obliger les utilisateurs humains, y compris ceux qui ont besoin d'un accès administrateur, à utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide Services AWS d'informations d'identification temporaires.

Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, d'un fournisseur d'identité Web AWS Directory Service, du répertoire Identity Center ou de tout utilisateur qui y accède à l'aide des informations d'identification fournies Services AWS par le biais d'une source d'identité. Lorsque des identités fédérées y accèdent Comptes AWS, elles assument des rôles, qui fournissent des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Vous pouvez créer des utilisateurs et des groupes dans IAM Identity Center, ou vous pouvez

vous connecter et synchroniser avec un ensemble d'utilisateurs et de groupes dans votre propre source d'identité afin de les utiliser dans toutes vos applications Comptes AWS et applications. Pour obtenir des informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou application. Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme telles que des mots de passe et des clés d'accès. Toutefois, si certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme avec les utilisateurs IAM, nous vous recommandons d'effectuer une rotation des clés d'accès. Pour plus d'informations, consultez Rotation régulière des clés d'accès pour les cas d'utilisation nécessitant des informations d'identification dans le Guide de l'utilisateur IAM.

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez nommer un groupe IAMAdminset lui donner les autorisations nécessaires pour administrer les ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu'un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d'informations d'identification permanentes, mais les rôles fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Cas d'utilisation pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une identité au sein de vous Compte AWS dotée d'autorisations spécifiques. Le concept ressemble à celui d'utilisateur IAM, mais le rôle IAM n'est pas associé à une personne en particulier. Pour assumer temporairement un rôle IAM dans le AWS Management Console, vous pouvez passer d'un rôle d'utilisateur à un rôle IAM (console). Vous pouvez assumer un rôle en appelant une opération d' AWS API AWS CLI ou en utilisant une URL personnalisée. Pour plus d'informations sur les méthodes d'utilisation des rôles, consultez Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré : pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
- Autorisations d'utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d'autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, dans certains Services AWS cas, vous pouvez associer une politique directement à une ressource (au lieu d'utiliser un rôle comme proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, un rôle de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.
  - Rôle de service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM.

Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

- Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés au service apparaissent dans votre Compte AWS fichier et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une EC2 instance et qui envoient des demandes AWS CLI d' AWS API. Cela est préférable au stockage des clés d'accès dans l' EC2 instance. Pour attribuer un AWS rôle à une EC2 instance et le rendre disponible pour toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes exécutés sur l' EC2 instance d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utiliser un rôle IAM pour accorder des autorisations aux applications exécutées sur des EC2 instances Amazon dans le guide de l'utilisateur IAM.

Gestion des accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal (utilisateur, utilisateur root ou session de rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations sur la structure et le contenu des documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une politique qui autorise l'action `iam:GetRole`. Un utilisateur appliquant cette politique peut obtenir des informations sur le rôle à partir de AWS Management Console AWS CLI, de ou de l' AWS API.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre Compte AWS. Les politiques gérées incluent les politiques AWS gérées et les politiques gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

Politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Listes de contrôle d'accès (ACLs)

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3 et AWS WAF Amazon VPC sont des exemples de services compatibles. ACLs Pour en savoir plus ACLs, consultez la présentation de la liste de contrôle d'accès (ACL) dans le guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courants. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations : une limite d'autorisations est une fonctionnalité avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations en résultant représentent la combinaison des politiques basées sur l'identité d'une entité et de ses limites d'autorisation. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCPs) : SCPs politiques JSON qui spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO) dans AWS Organizations. AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les multiples propriétés de votre entreprise. Si vous activez toutes les fonctionnalités d'une organisation, vous pouvez appliquer des politiques de contrôle des services (SCPs) à l'un ou à l'ensemble de vos comptes. Le SCP limite les autorisations pour les entités figurant dans les comptes des membres, y compris chacune Utilisateur racine d'un compte AWS d'entre elles. Pour plus d'informations sur les Organizations SCPs, voir Politiques de contrôle des services dans le Guide de AWS Organizations l'utilisateur.
- Politiques de contrôle des ressources (RCPs) : RCPs politiques JSON que vous pouvez utiliser pour définir le maximum d'autorisations disponibles pour les ressources de vos comptes sans mettre à jour les politiques IAM associées à chaque ressource que vous possédez. Le RCP limite les autorisations pour les ressources des comptes membres et peut avoir un impact sur les

autorisations effectives pour les identités, y compris Utilisateur racine d'un compte AWS, qu'elles appartiennent ou non à votre organisation. Pour plus d'informations sur les Organizations RCPs, y compris une liste de ces Services AWS supports RCPs, consultez la section Resource control policies (RCPs) dans le guide de AWS Organizations l'utilisateur.

- Politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de séance en résultant sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations, consultez Politiques de session dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

Comment fonctionne le service d'injection de AWS défauts avec IAM

Avant d'utiliser IAM pour gérer l'accès au AWS FIS, découvrez quelles fonctionnalités IAM peuvent être utilisées avec le FIS. AWS

Fonctionnalités IAM que vous pouvez utiliser avec le service d'injection de AWS défauts

Fonctionnalité IAM	AWS Assistance FIS
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Non
Actions de politique	Oui
Ressources de politique	Oui
Clés de condition de politique (spécifiques au service)	Oui

Fonctionnalité IAM	AWS Assistance FIS
ACLs	Non
ABAC (étiquettes dans les politiques)	Oui
Informations d'identification temporaires	Oui
Autorisations de principal	Oui
Rôles de service	Oui
Rôles liés à un service	Oui

Pour obtenir une vue d'ensemble de la façon dont le AWS FIS et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez la section AWS Services compatibles avec IAM dans le Guide de l'utilisateur IAM.

Politiques basées sur l'identité pour le FIS AWS

Prend en charge les politiques basées sur l'identité : oui

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Vous ne pouvez pas spécifier le principal dans une politique basée sur une identité, car celle-ci s'applique à l'utilisateur ou au rôle auquel elle est attachée. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Exemples de politiques basées sur l'identité pour le FIS AWS

Pour consulter des exemples de politiques basées sur l'identité AWS FIS, voir. AWS Exemples de politiques relatives au service d'injection de défauts

Politiques basées sur les ressources au sein du FIS AWS

Prend en charge les politiques basées sur les ressources : non

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal intercompte à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource sont différents Comptes AWS, un administrateur IAM du compte sécurisé doit également accorder à l'entité principale (utilisateur ou rôle) l'autorisation d'accéder à la ressource. Pour ce faire, il attache une politique basée sur une identité à l'entité. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Actions politiques pour la AWS FIS

Prend en charge les actions de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Les actions de stratégie portent généralement le même nom que l'opération AWS d'API associée. Il existe quelques exceptions, telles que les actions avec autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour consulter la liste des actions AWS FIS, voir Actions définies par le service d'injection de AWS défauts dans la référence d'autorisation du service.

Les actions politiques dans AWS FIS utilisent le préfixe suivant avant l'action :

```
fis
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
    "fis:action1",
    "fis:action2"
]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `List`, incluez l'action suivante :

```
"Action": "fis:List*"
```

Ressources politiques pour la AWS FIS

Prend en charge les ressources de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Certaines actions de l'API AWS FIS prennent en charge plusieurs ressources. Pour spécifier plusieurs ressources dans une seule instruction, séparez-les ARNs par des virgules.

```
"Resource": [
    "resource1",
    "resource2"
]
```

Pour consulter la liste des types de ressources AWS FIS et leurs caractéristiques ARNs, consultez la section Types de ressources définis par le service d'injection de AWS défauts dans la référence d'autorisation du service. Pour savoir avec quelles actions vous pouvez spécifier l'ARN de chaque ressource, voir Actions définies par le service d'injection de AWS défauts.

Clés de conditions de politique pour le AWS FIS

Prend en charge les clés de condition de politique spécifiques au service : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une `OR` opération logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments d'une politique IAM : variables et identifications dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques au service. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Pour consulter la liste des clés de condition AWS FIS, voir Clés de condition pour le service d'injection de AWS défauts dans la référence d'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, voir Actions définies par le service d'injection de AWS défauts.

Pour consulter des exemples de politiques basées sur l'identité AWS FIS, voir. AWS Exemples de politiques relatives au service d'injection de défauts

ACLs en AWS FIS

Supports ACLs : Non

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

ABAC avec FIS AWS

Prise en charge d'ABAC (balises dans les politiques) : Oui

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux entités IAM (utilisateurs ou rôles) et à de nombreuses AWS ressources. L'étiquetage des entités et des ressources est la première étape d'ABAC. Vous concevez ensuite des politiques ABAC pour autoriser des opérations quand l'identification du principal correspond à celle de la ressource à laquelle il tente d'accéder.

L'ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur ABAC, consultez Définition d'autorisations avec l'autorisation ABAC dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès par attributs (ABAC) dans le Guide de l'utilisateur IAM.

Pour consulter un exemple de politique basée sur l'identité visant à limiter l'accès à une ressource en fonction des balises associées à cette ressource, consultez. Exemple : utilisation de balises pour contrôler l'utilisation des ressources

Utilisation d'informations d'identification temporaires avec AWS FIS

Prend en charge les informations d'identification temporaires : oui

Certains Services AWS ne fonctionnent pas lorsque vous vous connectez à l'aide d'informations d'identification temporaires. Pour plus d'informations, y compris celles qui Services AWS fonctionnent avec des informations d'identification temporaires, consultez Services AWS la section relative à l'utilisation d'IAM dans le guide de l'utilisateur d'IAM.

Vous utilisez des informations d'identification temporaires si vous vous connectez à l' AWS Management Console aide d'une méthode autre qu'un nom d'utilisateur et un mot de passe. Par exemple, lorsque vous accédez à AWS l'aide du lien d'authentification unique (SSO) de votre entreprise, ce processus crée automatiquement des informations d'identification temporaires. Vous créez également automatiquement des informations d'identification temporaires lorsque vous vous connectez à la console en tant qu'utilisateur, puis changez de rôle. Pour plus d'informations sur le changement de rôle, consultez Passage d'un rôle utilisateur à un rôle IAM (console) dans le Guide de l'utilisateur IAM.

Vous pouvez créer manuellement des informations d'identification temporaires à l'aide de l' AWS API AWS CLI or. Vous pouvez ensuite utiliser ces informations d'identification temporaires pour y accéder AWS. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.

Autorisations principales interservices pour FIS AWS

Prend en charge les sessions d'accès direct (FAS) : oui

Lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal

appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.

Rôles de service pour AWS FIS

Prend en charge les rôles de service : oui

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

Rôles liés aux services pour FIS AWS

Prend en charge les rôles liés aux services : Oui

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés au service apparaissent dans votre Compte AWS fichier et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour plus de détails sur la création ou la gestion des rôles liés aux services AWS FIS, consultez. Utiliser des rôles liés à un service pour le service d'injection de AWS défauts

AWS Exemples de politiques relatives au service d'injection de défauts

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier des ressources AWS FIS. Ils ne peuvent pas non plus effectuer de tâches à l'aide de l'API AWS Management Console, AWS Command Line Interface (AWS CLI) ou de AWS l'API. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques IAM (console) dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par AWS FIS, y compris le ARNs format de chaque type de ressource, voir Actions, ressources et clés de condition pour le service d'injection de AWS défauts dans la référence d'autorisation du service.

Table des matières

- Bonnes pratiques en matière de politiques
- Exemple : utilisation de la AWS console FIS
- Exemple : liste des actions AWS FIS disponibles
- Exemple : création d'un modèle d'expérience pour une action spécifique
- Exemple : démarrer une expérience
- Exemple : utilisation de balises pour contrôler l'utilisation des ressources
- Exemple : supprimer un modèle d'expérience avec une balise spécifique
- Exemple : Autoriser les utilisateurs à afficher leurs propres autorisations
- Exemple : utilisez des clés de condition pour ec2:InjectApiError
- Exemple : utilisez des clés de condition pour aws:s3:bucket-pause-replication

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité déterminent si quelqu'un peut créer, accéder ou supprimer des ressources AWS FIS dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accordez les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre

privilège. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.

- Utilisez des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que AWS CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.
- Utilisez l'Analyseur d'accès IAM pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : l'Analyseur d'accès IAM valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politiques avec IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue. Compte AWS Pour exiger la MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Sécurisation de l'accès aux API avec MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Exemple : utilisation de la AWS console FIS

Pour accéder à la console du service d'injection de AWS défauts, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations doivent vous permettre de répertorier et d'afficher les détails relatifs aux ressources AWS FIS de votre Compte AWS. 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.

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 qu'ils tentent d'effectuer.

L'exemple de politique suivant accorde l'autorisation de répertorier et d'afficher toutes les ressources AWS FIS à l'aide de la console AWS FIS, mais pas de les créer, de les mettre à jour ou de les supprimer. Il autorise également l'affichage des ressources disponibles utilisées par toutes les actions AWS FIS que vous pouvez spécifier dans un modèle d'expérience.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FISReadOnlyActions",
            "Effect": "Allow",
            "Action": [
                "fis:List*",
                "fis:Get*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AdditionalReadOnlyActions",
            "Effect": "Allow",
            "Action": [
                "ssm:Describe*",
                "ssm:Get*",
                "ssm:List*",
                "ec2:DescribeInstances",
                "rds:DescribeDBClusters",
                "ecs:DescribeClusters",
                "ecs:ListContainerInstances",
                "eks:DescribeNodegroup",
                "cloudwatch:DescribeAlarms",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsToCreateServiceLinkedRole",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "fis.amazonaws.com"
                }
```

```
            }
        }
    ]
}
```

Exemple : liste des actions AWS FIS disponibles

La politique suivante autorise la liste des actions AWS FIS disponibles.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "fis:ListActions"
            ],
            "Resource": "arn:aws:fis:*:*:action/*"
        }
    ]
}
```

Exemple : création d'un modèle d'expérience pour une action spécifique

La politique suivante autorise la création d'un modèle d'expérience pour l'action`aws:ec2:stop-instances`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyExample",
      "Effect": "Allow",
      "Action": [
          "fis:CreateExperimentTemplate"
      ],
      "Resource": [
        "arn:aws:fis:*:*:action/aws:ec2:stop-instances",
        "arn:aws:fis:*:*:experiment-template/*"
      ]
    },
    {
```

```
      "Sid": "PolicyPassRoleExample",
      "Effect": "Allow",
      "Action": [
          "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::account-id:role/role-name"
      ]
    }
  ]
}
```

Exemple : démarrer une expérience

La politique suivante autorise le lancement d'une expérience en utilisant le rôle IAM et le modèle d'expérience spécifiés. Cela permet également à AWS FIS de créer un rôle lié à un service au nom de l'utilisateur. Pour de plus amples informations, veuillez consulter Utiliser des rôles liés à un service pour le service d'injection de AWS défauts.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyExample",
      "Effect": "Allow",
      "Action": [
          "fis:StartExperiment"
      ],
      "Resource": [
        "arn:aws:fis:*:*:experiment-template/experiment-template-id",
        "arn:aws:fis:*:*:experiment/*"
      ]
    },
    {
        "Sid": "PolicyExampleforServiceLinkedRole",
        "Effect": "Allow",
        "Action": "iam:CreateServiceLinkedRole",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "iam:AWSServiceName": "fis.amazonaws.com"
            }
        }
```

```
    }
  ]
}
```

Exemple : utilisation de balises pour contrôler l'utilisation des ressources

La politique suivante autorise l'exécution d'expériences à partir de modèles d'expériences dotés de la balise`Purpose=Test`. Il n'autorise pas la création ou la modification de modèles d'expériences, ni l'exécution d'expériences à l'aide de modèles ne possédant pas la balise spécifiée.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "fis:StartExperiment",
            "Resource": "arn:aws:fis:*:*:experiment-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Test"
                }
            }
        }
    ]
}
```

Exemple : supprimer un modèle d'expérience avec une balise spécifique

La politique suivante autorise la suppression d'un modèle d'expérience comportant une balise`Purpose=Test`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "fis:DeleteExperimentTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:ResourceTag/Purpose": "Test"
                }
            }
        }
    ]
}
```

Exemple : 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:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

Exemple : utilisez des clés de condition pour **ec2:InjectApiError**

L'exemple de politique suivant utilise la clé de `ec2:FisTargetArns` condition pour définir le périmètre des ressources cibles. Cette politique autorise les actions du AWS FIS `aws:ec2:api-insufficient-instance-capacity-error` et. `aws:ec2:asg-insufficient-instance-capacity-error`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:InjectApiError",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "ec2:FisActionId": [
                        "aws:ec2:api-insufficient-instance-capacity-error",
                    ],
                    "ec2:FisTargetArns": [
                        "arn:aws:iam:*:*:role:role-name"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:InjectApiError",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "ec2:FisActionId": [
                        "aws:ec2:asg-insufficient-instance-capacity-error"
                    ],
                    "ec2:FisTargetArns": [

 "arn:aws:autoscaling:*:*:autoScalingGroup:uuid:autoScalingGroupName/asg-name"
                    ]
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:DescribeAutoScalingGroups",
            "Resource": "*"
        }
    ]
}
```

Exemple : utilisez des clés de condition pour **aws:s3:bucket-pause-replication**

L'exemple de politique suivant utilise la clé de `S3:IsReplicationPauseRequest` condition pour autoriser `PutReplicationConfiguration` et `GetReplicationConfiguration` uniquement lorsqu'elle est utilisée par le AWS FIS dans le contexte de l'action AWS FIS. `aws:s3:bucket-pause-replication`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "S3:PauseReplication"
            ],
            "Resource": "arn:aws:s3:::mybucket",
            "Condition": {
                "StringEquals": {
                    "s3:DestinationRegion": "region"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "S3:PutReplicationConfiguration",
                "S3:GetReplicationConfiguration"
            ],
            "Resource": "arn:aws:s3:::mybucket",
            "Condition": {
                "BoolIfExists": {
                    "s3:IsReplicationPauseRequest": "true"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "S3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
    }
```

Utiliser des rôles liés à un service pour le service d'injection de AWS défauts

AWS Le service d'injection de défauts utilise des AWS Identity and Access Management rôles liés au service (IAM). Un rôle lié à un service est un type unique de rôle IAM directement lié au FIS. AWS Les rôles liés au service sont prédéfinis par le AWS FIS et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service facilite la configuration du AWS FIS, car vous n'avez pas à ajouter manuellement les autorisations nécessaires pour gérer la surveillance et la sélection des ressources pour les expériences. AWS Le FIS définit les autorisations associées à ses rôles liés aux services et, sauf indication contraire, seul le AWS FIS peut assumer ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Outre le rôle lié au service, vous devez également spécifier un rôle IAM qui autorise la modification des ressources que vous spécifiez comme cibles dans un modèle de test. Pour de plus amples informations, veuillez consulter Rôles IAM pour les expériences AWS FIS.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable des ressources connexes. Cela protège vos ressources AWS FIS car vous ne pouvez pas supprimer par inadvertance l'autorisation d'accès aux ressources.

Autorisations de rôle liées au service pour FIS AWS

AWS Le FIS utilise le rôle lié à un service nommé AWSServiceRoleForFIS pour lui permettre de gérer la surveillance et la sélection des ressources pour les expériences.

Le rôle lié au service AWSServiceRoleForFIS fait confiance aux services suivants pour assumer le rôle :

- `fis.amazonaws.com`

Le rôle lié au service AWSServiceRoleForFIS utilise la politique gérée Amazon. FISService RolePolicy Cette politique permet à la AWS FIS de gérer le suivi et la sélection des ressources pour les expériences. Pour plus d'informations, consultez Amazon FISService RolePolicy dans le AWS Managed Policy Reference.

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour que le rôle lié au service AWSServiceRoleForFIS soit correctement créé, l'identité IAM avec laquelle vous utilisez AWS FIS doit disposer des autorisations requises. Pour accorder les autorisations requises, associez la stratégie suivante à l'identité IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "fis.amazonaws.com"
                }
            }
        }
    ]
}
```

Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour FIS AWS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous lancez une expérience AWS FIS dans le AWS Management Console, le ou l' AWS API AWS CLI, le AWS FIS crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous lancez un test AWS FIS, le AWS FIS crée à nouveau le rôle lié au service pour vous.

Modifier un rôle lié à un service pour FIS AWS

AWS FIS ne vous permet pas de modifier le rôle lié au service AWSServiceRoleForFIS. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour plus d'informations, consultez Modification d'un rôle lié à un service dans le IAM Guide de l'utilisateur.

Supprimer un rôle lié à un service pour FIS AWS

Si vous n'avez plus besoin d'utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

> **Note**
>
> Si le service AWS FIS utilise le rôle lorsque vous essayez de nettoyer les ressources, le nettoyage risque d'échouer. Si cela se produit, patientez quelques minutes et réessayez.

Nettoyer les ressources AWS du FIS utilisées par le AWSService RoleFor FIS

Assurez-vous qu'aucune de vos expériences n'est en cours d'exécution. Si nécessaire, arrêtez vos expériences. Pour de plus amples informations, veuillez consulter Arrêt d'une expérience.

Pour supprimer manuellement le rôle lié à un service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au service AWSServiceRoleForFIS. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les AWS rôles liés aux services FIS

AWS La FIS prend en charge l'utilisation de rôles liés au service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez la section Points de terminaison et quotas du service d'injection de AWS défauts.

AWS politiques gérées pour le service d'injection de AWS défauts

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 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.

AWS politique gérée : Amazon FISService RolePolicy

Cette politique est associée au rôle lié au service nommé AWSServiceRoleForFIS pour permettre au FIS de gérer le suivi et la AWS sélection des ressources pour les expériences. Pour de plus amples informations, veuillez consulter Utiliser des rôles liés à un service pour le service d'injection de AWS défauts.

AWS politique gérée : AWSFault InjectionSimulator EC2 Accès

Utilisez cette politique dans le cadre d'un rôle d'expérimentation pour autoriser le AWS FIS à exécuter des tests utilisant les actions AWS FIS pour Amazon. EC2 Pour de plus amples informations, veuillez consulter the section called "Rôle d'expérience".

Pour consulter les autorisations associées à cette politique, consultez la section AWSFaultInjectionSimulatorEC2Accès dans le manuel de référence des politiques AWS gérées.

AWS politique gérée : AWSFault InjectionSimulator ECSAccess

Utilisez cette politique dans le cadre d'un rôle d'expérimentation pour autoriser le AWS FIS à exécuter des tests utilisant les actions AWS FIS pour Amazon ECS. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.

Pour voir les autorisations de cette stratégie, consultez AWSFaultInjectionSimulatorECSAccess dans le AWS Guide de référence des stratégies gérées par.

AWS politique gérée : AWSFault InjectionSimulator EKSAccess

Utilisez cette politique dans le cadre d'un rôle d'expérimentation pour autoriser le AWS FIS à exécuter des tests utilisant les actions AWS FIS pour Amazon EKS. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.

Pour voir les autorisations de cette stratégie, consultez AWSFaultInjectionSimulatorEKSAccess dans le AWS Guide de référence des stratégies gérées par.

AWS politique gérée : AWSFault InjectionSimulatorNetworkAccess

Utilisez cette politique dans le cadre d'un rôle d'expérimentation pour autoriser le AWS FIS à exécuter des expériences utilisant les actions de mise en réseau du AWS FIS. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.

Pour voir les autorisations de cette stratégie, consultez AWSFaultInjectionSimulatorNetworkAccess dans le AWS Guide de référence des stratégies gérées par.

AWS politique gérée : AWSFault InjectionSimulator RDSAccess

Utilisez cette politique dans le cadre d'un rôle d'expérimentation pour autoriser le AWS FIS à exécuter des tests utilisant les actions AWS FIS pour Amazon RDS. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.

Pour voir les autorisations de cette stratégie, consultez AWSFaultInjectionSimulatorRDSAccess dans le AWS Guide de référence des stratégies gérées par.

AWS politique gérée : AWSFault InjectionSimulator SSMAccess

Utilisez cette politique dans le cadre d'un rôle d'expérimentation pour autoriser la AWS FIS à exécuter des expériences utilisant les actions AWS FIS pour Systems Manager. Pour de plus amples informations, veuillez consulter the section called “Rôle d'expérience”.

Pour voir les autorisations de cette stratégie, consultez AWSFaultInjectionSimulatorSSMAccess dans le AWS Guide de référence des stratégies gérées par.

AWS Mises à jour des politiques AWS gérées par le FIS

Consultez les détails des mises à jour apportées aux politiques AWS gérées pour AWS FIS depuis que ce service a commencé à suivre ces modifications.

Modification	Description	Date
AWSFaultInjectionSimulatorECSAccess – Mise à jour d’une stratégie existante	Autorisations ajoutées pour permettre à AWS FIS de résoudre les cibles ECS.	25 janvier 2024
AWSFaultInjectionSimulatorNetworkAccess – Mise à jour d’une politique existante	Des autorisations ont été ajoutées pour permettre au AWS FIS d'exécuter des expériences à l'aide des aws:network:transit-gateway -disrupt-cross-region-connectivity actions aws:network:route-table-disrupt-cross-region-connectivity et.	25 janvier 2024
AWSFaultInjectionSimulatorEC2Accès — Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre au AWS FIS de résoudre les EC2 instances.	13 novembre 2023
AWSFaultInjectionSimulatorEKSAccess – Mise à jour d’une politique existante	Des autorisations ont été ajoutées pour permettre à AWS FIS de résoudre les cibles EKS.	13 novembre 2023
AWSFaultInjectionSimulatorRDSAccess – Mise à jour d’une politique existante	Des autorisations ont été ajoutées pour permettre à AWS FIS de résoudre les cibles RDS.	13 novembre 2023

Modification	Description	Date
AWSFaultInjectionSimulatorEC2Accès — Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre au AWS FIS d'exécuter des documents SSM sur des EC2 instances et de mettre fin EC2 à des instances.	2 juin 2023
AWSFaultInjectionSimulatorSSMAccess – Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre à AWS FIS d'exécuter des documents SSM sur EC2 des instances.	2 juin 2023
AWSFaultInjectionSimulatorECSAccess – Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre au AWS FIS d'exécuter des expériences à l'aide des nouvelles aws:ecs:task actions.	1er juin 2023
AWSFaultInjectionSimulatorEKSAccess – Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre au AWS FIS d'exécuter des expériences à l'aide des nouvelles aws:eks:pod actions.	26 octobre 2022
AWSFaultInjectionSimulatorEC2Accès — Nouvelle politique	Ajout d'une politique permettant à AWS FIS de mener une expérience utilisant des actions AWS FIS pour Amazon. EC2	26 octobre 2022
AWSFaultInjectionSimulatorECSAccess : nouvelle politique	Ajout d'une politique permettant à AWS FIS d'exécuter une expérience utilisant des actions AWS FIS pour Amazon ECS.	26 octobre 2022
AWSFaultInjectionSimulatorEKSAccess : nouvelle politique	Ajout d'une politique permettant à AWS FIS d'exécuter une expérience utilisant des actions AWS FIS pour Amazon EKS.	26 octobre 2022

Modification	Description	Date
AWSFaultInjectionSimulatorNetworkAccess : nouvelle politique	Ajout d'une politique permettant au AWS FIS d'exécuter une expérience utilisant des actions réseau du AWS FIS.	26 octobre 2022
AWSFaultInjectionSimulatorRDSAccess : nouvelle politique	Ajout d'une politique permettant à AWS FIS d'exécuter une expérience utilisant des actions AWS FIS pour Amazon RDS.	26 octobre 2022
AWSFaultInjectionSimulatorSSMAccess : nouvelle politique	Ajout d'une politique permettant à AWS FIS d'exécuter une expérience utilisant des actions AWS FIS pour Systems Manager.	26 octobre 2022
Amazon FISService RolePolicy — Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre à AWS FIS de décrire les sous-réseaux.	26 octobre 2022
Amazon FISService RolePolicy — Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre à AWS FIS de décrire les clusters EKS.	7 juillet 2022
Amazon FISService RolePolicy — Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre à AWS FIS de répertorier et de décrire les tâches de vos clusters.	7 février 2022
Amazon FISService RolePolicy — Mise à jour d'une politique existante	Suppression de la `events:ManagedBy` condition de l'`events:DescribeRule` action.	6 janvier 2022
Amazon FISService RolePolicy — Mise à jour d'une politique existante	Des autorisations ont été ajoutées pour permettre au AWS FIS de récupérer l'historique des CloudWatch alarmes utilisées en cas d'arrêt.	30 Juin 2021

Modification	Description	Date
AWS Le FIS a commencé à suivre les modifications	AWS Le FIS a commencé à suivre les modifications apportées à ses politiques AWS gérées	1er mars 2021

Sécurité de l'infrastructure dans le service d'injection de AWS défauts

En tant que service géré, le service d'injection de AWS défauts 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 au AWS FIS via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Accédez au AWS FIS à l'aide d'un point de terminaison VPC d'interface ()AWS PrivateLink

Vous pouvez établir une connexion privée entre votre VPC et le service d'injection de AWS défauts en créant un point de terminaison VPC d'interface. Les points de terminaison VPC sont alimentés par AWS PrivateLinkune technologie qui vous permet d'accéder au AWS FIS en privé APIs sans

passerelle Internet, appareil NAT, connexion VPN ou connexion Direct AWS Connect. Les instances de votre VPC n'ont pas besoin d'adresses IP publiques pour communiquer avec AWS FIS. APIs

Chaque point de terminaison d'interface est représenté par une ou plusieurs interfaces réseau Elastic dans vos sous-réseaux.

Pour plus d'informations, consultez la section Accès Services AWS par AWS PrivateLink le biais du AWS PrivateLink guide.

Considérations relatives aux points AWS de terminaison VPC FIS

Avant de configurer un point de terminaison VPC d'interface pour AWS FIS, consultez la section Accès et Service AWS utilisation d'un point de terminaison VPC d'interface dans le Guide.AWS PrivateLink

AWS FIS permet d'appeler toutes ses actions d'API depuis votre VPC.

Création d'un point de terminaison VPC d'interface pour FIS AWS

Vous pouvez créer un point de terminaison VPC pour le service AWS FIS à l'aide de la console Amazon VPC ou du (). AWS Command Line Interface AWS CLI Pour plus d'informations, consultez Créer un point de terminaison d'un VPC dans le Guide AWS PrivateLink .

Créez un point de terminaison VPC pour AWS FIS en utilisant le nom de service suivant :. `com.amazonaws.region.fis`

Si vous activez le DNS privé pour le point de terminaison, vous pouvez envoyer des demandes d'API à AWS FIS en utilisant son nom DNS par défaut pour la région, par exemple,`fis.us-east-1.amazonaws.com`.

Création d'une politique de point de terminaison VPC pour FIS AWS

Vous pouvez associer une politique de point de terminaison à votre point de terminaison VPC qui contrôle l'accès au AWS FIS. La politique spécifie les informations suivantes :

- Le principal qui peut exécuter des actions.
- Les actions qui peuvent être effectuées.
- Les ressources sur lesquelles les actions peuvent être exécutées.

Pour plus d'informations, consultez la section Contrôler l'accès aux points de terminaison VPC à l'aide des politiques relatives aux points de terminaison dans le Guide.AWS PrivateLink

Exemple : politique de point de terminaison VPC pour des actions FIS spécifiques AWS

La politique de point de terminaison VPC suivante accorde à tous les principaux l'accès aux actions AWS FIS répertoriées sur toutes les ressources.

```
{
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "fis:ListExperimentTemplates",
            "fis:StartExperiment",
            "fis:StopExperiment",
            "fis:GetExperiment"
         ],
         "Resource":"*",
         "Principal":"*"
      }
   ]
}
```

Exemple : politique de point de terminaison VPC qui refuse l'accès depuis un point spécifique Compte AWS

La politique de point de terminaison VPC suivante refuse l' Compte AWS accès spécifié à toutes les actions et ressources, mais accorde tous les autres Comptes AWS accès à toutes les actions et ressources.

```
{
   "Statement":[
      {
         "Effect": "Allow",
         "Action": "*",
         "Resource": "*",
         "Principal": "*"
      },
      {
         "Effect":"Deny",
         "Action": "*",
```

```
          "Resource": "*",
          "Principal": {
            "AWS": [ "123456789012" ]
          }
       }
   ]
}
```

Marquage de vos ressources AWS FIS

Une balise est une étiquette de métadonnées que vous attribuez ou que vous AWS attribuez à une AWS ressource. Chaque balise se compose d'une clé et d'une valeur. Vous définissez la clé et la valeur des balises que vous affectez. Par exemple, vous pouvez définir la clé comme `purpose` et la valeur comme `test` pour une ressource.

Les balises vous permettent d'effectuer les actions suivantes :

- Identifiez et organisez vos AWS ressources. De nombreux AWS services prennent en charge le balisage. Vous pouvez donc attribuer le même tag aux ressources de différents services pour indiquer que les ressources sont liées.
- Contrôlez l'accès à vos AWS ressources. Pour plus d'informations, consultez Contrôle de l'accès à l'aide de balises dans le Guide de l'utilisateur IAM.

Restrictions de balisage

Les restrictions de base suivantes s'appliquent aux balises figurant sur les ressources AWS FIS :

- Nombre maximum de balises que vous pouvez attribuer à une ressource : 50
- Longueur de clé maximale : 128 caractères Unicode
- Longueur de valeur maximale : 256 caractères Unicode
- Caractères valides pour les clés et les valeurs : a-z, A-Z, 0-9, espace et les caractères suivants : _. :/= + - et @
- Les clés et les valeurs sont sensibles à la casse.
- Vous ne pouvez pas l'utiliser `aws:` comme préfixe pour les clés, car il est réservé à l'usage AWS

Travailler avec des tags

Les ressources du service d'injection de AWS défauts (AWS FIS) suivantes prennent en charge le balisage :

- Actions
- Expériences
- Modèles d'expériences

Vous pouvez utiliser la console pour utiliser des balises pour des expériences et des modèles d'expériences. Pour plus d’informations, consultez les ressources suivantes :

- Marquer une expérience
- Modèles d'expériences de tags

Vous pouvez utiliser les AWS CLI commandes suivantes pour utiliser des balises pour les actions, les expériences et les modèles d'expériences :

- tag-resource — Ajoute des balises à une ressource.
- untag-resource — Supprime les balises d'une ressource.
- list-tags-for-resource— Répertorie les balises d'une ressource spécifique.

Quotas et limites pour le service d'injection de AWS défauts

Vous Compte AWS disposez de quotas par défaut, anciennement appelés limites, pour chaque AWS service. Sauf indication contraire, chaque quota est spécifique à une région. Vous pouvez demander des augmentations pour les quotas marqués comme ajustables dans le tableau ci-dessous.

Pour consulter les quotas de AWS FIS dans votre compte, ouvrez la console Service Quotas. Dans le volet de navigation, choisissez AWS services, puis sélectionnez AWS Fault Injection Service. Les valeurs allant jusqu'aux quotas approuvés automatiquement sont appliquées instantanément. Les quotas approuvés automatiquement sont présentés dans la colonne de description du tableau ci-dessous. Si vous avez besoin de quotas dépassant les limites approuvées automatiquement, veuillez en faire la demande. Les valeurs supérieures aux limites approuvées automatiquement sont examinées par le service client et approuvées dans la mesure du possible.

Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Vous Compte AWS disposez des quotas suivants liés au AWS FIS.

Nom	Par défaut	Ajusta	Description
Durée de l'action en heures	Chaque région prise en charge : 12	Non	Le nombre maximum d'heures autorisées pour exécuter une action sur ce compte dans la région actuelle.
Actions par modèle d'expérience	Chaque Région prise en charge : 20	Non	Nombre maximum d'actions que vous pouvez créer dans un modèle d'expérience dans ce compte dans la région actuelle.
Expériences actives	Chaque région prise en charge : 5	Non	Le nombre maximum d'expériences actives que vous pouvez exécuter simultanément sur ce

Nom	Par défaut	Ajusta	Description
			compte dans la région actuelle.
Conservation des données des expériences terminées en jours	Chaque région prise en charge : 120	Non	Le nombre maximum de jours autorisés à la AWS FIS pour conserver les données relatives aux expériences achevées sur ce compte dans la région actuelle.
Durée de l'expérience en heures	Chaque région prise en charge : 12	Non	Le nombre maximum d'heures autorisées pour exécuter une expérienc e sur ce compte dans la région actuelle.
Modèles d'expériences	Chaque région prise en charge : 500	Non	Le nombre maximum de modèles d'expériences que vous pouvez créer dans ce compte dans la région actuelle.
Nombre maximum de listes de préfixes gérées dans aws:network : -region-c onnectivity route-table-disrupt-cross	Chaque région prise en charge : 15	Non	Le nombre maximum de listes de préfixes gérées autorisées par aws:netwo rk : route-table-disrupt-cross -region-connectivi ty, par action.
Nombre maximum de tables de routage dans aws:network : -region-connectivity route-table-disrupt-cross	Chaque Région prise en charge : 10	Non	Le nombre maximum de tables de routage autorisées par aws:netwo rk : route-table-disrupt-cross -region-connectivi ty, par action.

Nom	Par défaut	Ajusta	Description
Nombre maximum de routes dans aws:network : -region-connectivity route-table-disrupt-cross	Chaque région prise en charge : 200	Non	Le nombre maximum de routes autorisées par aws:network : route-table-disrupt-cross -region-connectivity, par action.
Actions parallèles par expérience	Chaque Région prise en charge : 10	Non	Le nombre maximum d'actions que vous pouvez exécuter en parallèle dans une expérience sur ce compte dans la région actuelle.
Conditions d'arrêt par modèle d'expérience	Chaque région prise en charge : 5	Non	Le nombre maximum de conditions d'arrêt que vous pouvez ajouter à un modèle d'expérience dans ce compte dans la région actuelle.
Groupes Auto Scaling cibles pour aws:ec2 : -error asg-insufficient-instance-capacity	Chaque Région prise en charge : 5	Oui	Le nombre maximum de groupes Auto Scaling que aws:ec2 : asg-insufficient-instance-capacity -error peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 500.

Nom	Par défaut	Ajusta	Description
Buckets cibles pour aws:s3 : bucket-pause-replication	Chaque région prise en charge : 20	Oui	Le nombre maximum de compartiments S3 que aws:s3 : bucket-pause-replication peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 25.
Clusters cibles pour aws:ecs : drain-container-instances	Chaque Région prise en charge : 5	Oui	Le nombre maximum de clusters que aws:ecs : drain-container-instances peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 100.

Nom	Par défaut	Ajusta	Description
Clusters cibles pour aws:rds : failover-db-cluster	Chaque Région prise en charge : 5	Oui	Le nombre maximum de clusters que aws:rds : failover-db-cluster peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augment ation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 160.
Cible DBInstances pour aws:rds : reboot-db-instances	Chaque Région prise en charge : 5	Oui	Le nombre maximum de cibles DBInstances que aws:rds : reboot-db -instances peut cibler lorsque vous identifie z des cibles à l'aide de balises, par expérience. Les demandes d'augment ation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 130.

Nom	Par défaut	Ajusta	Description
Instances cibles pour aws:ec2:reboot-instances	Chaque Région prise en charge : 5	Oui	Le nombre maximum d'instances que aws:ec2:reboot-instances peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 600.
Instances cibles pour aws:ec2:stop-instances	Chaque Région prise en charge : 5	Oui	Le nombre maximum d'instances que aws:ec2:stop-instances peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 400.

Nom	Par défaut	Ajusta	Description
Instances cibles pour aws:ec2:terminate-instances	Chaque Région prise en charge : 5	Oui	Le nombre maximum d'instances que aws:ec2:terminate-instances peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 300.
Instances cibles pour aws:ssm:send-command	Chaque Région prise en charge : 5	Oui	Le nombre maximum d'instances que aws:ssm:send-command peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.

Nom	Par défaut	Ajusta	Description
Groupes de nœuds cibles pour aws:eks : terminate-nodegroup-instances	Chaque Région prise en charge : 5	Oui	Le nombre maximum de groupes de nœuds que aws:eks : terminate-nodegroup-instances peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 100.
Pods cibles pour aws:eks : pod-cpu-stress	Chaque Région prise en charge : 50	Oui	Le nombre maximum de pods que aws:eks : pod-cpu-stress peut cibler lorsque vous identifiez des cibles à l'aide de paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour des valeurs allant jusqu'à 1 000.

Nom	Par défaut	Ajusta	Description
Pods cibles pour aws:eks:pod-delete	Chaque Région prise en charge : 50	Oui	Le nombre maximum de pods que aws:eks:pod-delete peut cibler lorsque vous identifiez des cibles à l'aide de paramètre s, par expérience. Les demandes d'augment ation de quota seront automatiquement approuvées pour des valeurs allant jusqu'à 1 000.
Pods cibles pour aws:eks : pod-io-stress	Chaque Région prise en charge : 50	Oui	Le nombre maximum de pods que aws:eks : pod-io-stress peut cibler lorsque vous identifiez des cibles à l'aide de paramètres, par expérience. Les demandes d'augment ation de quota seront automatiquement approuvées pour des valeurs allant jusqu'à 1 000.

Nom	Par défaut	Ajusta	Description
Pods cibles pour aws:eks : pod-memory-stress	Chaque Région prise en charge : 50	Oui	Le nombre maximum de pods que aws:eks : pod-memory-stress peut cibler lorsque vous identifiez des cibles à l'aide de paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour des valeurs allant jusqu'à 1 000.
Pods cibles pour aws:eks : pod-network-blackhole-port	Chaque Région prise en charge : 50	Oui	Le nombre maximum de pods que aws:eks : pod-network-blackhole-port peut cibler lorsque vous identifiez des cibles à l'aide de paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour des valeurs allant jusqu'à 1 000.

Nom	Par défaut	Ajusta	Description
Pods cibles pour aws:eks : pod-network-latency	Chaque Région prise en charge : 50	Oui	Le nombre maximum de pods que aws:eks : pod-network-latency peut cibler lorsque vous identifiez des cibles à l'aide de paramètres, par expérience. Les demandes d'augment ation de quota seront automatiquement approuvées pour des valeurs allant jusqu'à 1 000.
Pods cibles pour aws:eks : pod-network-packet-loss	Chaque Région prise en charge : 50	Oui	Le nombre maximum de pods que aws:eks : pod-network-packet-loss peut cibler lorsque vous identifiez des cibles à l'aide de paramètre s, par expérience. Les demandes d'augment ation de quota seront automatiquement approuvées pour des valeurs allant jusqu'à 1 000.

Nom	Par défaut	Ajusta	Description
Cible ReplicationGroups pour aws:elasticache : - Obsolète planifiée interrupt-cluster-az-power	Chaque Région prise en charge : 5	Oui	Le nombre maximum de cibles ReplicationGroups que aws:elasticache : interrupt-cluster-az-power peut cibler lorsque vous identifiez des cibles à l'aide de balises/paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 5.
Cible ReplicationGroups pour aws:elasticache : replicationgroup-interrupt-az-power	Chaque Région prise en charge : 5	Oui	Le nombre maximum de cibles ReplicationGroups que aws:elasticache : replicationgroup-interrupt-az-power peut cibler par expérience. Une limite quotidienne s'applique au ciblage ReplicationGroups. Pour plus d'informations, rendez-vous sur : https://docs.aws.amazon.com/fis/ latest/userguide/fis -quotas.html. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 20.

Nom	Par défaut	Ajusta	Description
Cible SpotInstances pour aws:ec2 : send-spot-instance-interruptions	Chaque Région prise en charge : 5	Oui	Le nombre maximum de cibles SpotInstances que aws:ec2 : send-spot-instance-interruptions peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 100.
Sous-réseaux cibles pour aws:network:disrupt-connectivity	Chaque Région prise en charge : 5	Oui	Le nombre maximum de sous-réseaux que aws:network:disrupt-connectivity peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les quotas supérieurs à 5 s'appliquent uniquement au paramètre scope:all. Si vous avez besoin d'un quota plus élevé pour un autre type de scope, contactez le service client. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 100.

Nom	Par défaut	Ajusta	Description
Sous-réseaux cibles pour aws:network : -region-connectivity route-table-disrupt-cross	Chaque région prise en charge : 6	Oui	Le nombre maximum de sous-réseaux que aws:network : route-table-disrupt-cross -region-connectivity peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.
Tâches cibles pour aws:ecs:stop-task	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches que aws:ecs:stop-task peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 500.

Nom	Par défaut	Ajusta	Description
Tâches cibles pour aws:ecs : task-cpu-stress	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches que aws:ecs : task-cpu-stress peut cibler lorsque vous identifiez des cibles à l'aide de balises/paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.
Tâches cibles pour aws:ecs : task-io-stress	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches que aws:ecs : task-io-stress peut cibler lorsque vous identifiez des cibles à l'aide de balises/paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.

Nom	Par défaut	Ajusta	Description
Tâches cibles pour aws:ecs : task-kill-process	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches que aws:ecs : task-kill-process peut cibler lorsque vous identifiez des cibles à l'aide de balises/paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.
Tâches cibles pour aws:ecs : task-network-blackhole-port	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches que aws:ecs : task-network-blackhole-port peut cibler lorsque vous identifiez des cibles à l'aide de balises/paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.

Nom	Par défaut	Ajusta	Description
Tâches cibles pour aws:ecs : task-network-latency	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches que aws:ecs : task-network-latency peut cibler lorsque vous identifiez des cibles à l'aide de balises/paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.
Tâches cibles pour aws:ecs : task-network-packet-loss	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches que aws:ecs : task-network-packet-loss peut cibler lorsque vous identifiez des cibles à l'aide de balises/paramètres, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.

Nom	Par défaut	Ajusta	Description
Cible TransitGateways pour aws:network : -region-connectivity transit-gateway-disrupt-cross	Chaque Région prise en charge : 5	Oui	Le nombre maximum de passerelles de transit que aws:network : transit-gateway-disrupt-cross -region-connectivity peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 50.
Volumes cibles pour aws:ebs : pause-volume-io	ca-central-1 : 160 eu-central-1 : 160 Chacune des autres régions prises en charge : 5	Oui	Le nombre maximum de volumes que aws:ebs : pause-volume-io peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 160.

Nom	Par défaut	Ajusta	Description
Configurations de compte cible par modèle d'expérience	Par région prise en charge : 10	Oui	Nombre maximal de configurations de compte cible que vous pouvez créer pour un modèle d'expérience dans ce compte dans la région actuelle. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 40.
Fonctions cibles pour aws:lambda : action. invocation-add-delay	Chaque Région prise en charge : 5	Oui	Le nombre maximum de fonctions Lambda que aws:lambda : invocation-add-delay peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 140.

Nom	Par défaut	Ajusta	Description
Fonctions cibles pour l'action aws:lambda:invocation-error.	Chaque Région prise en charge : 5	Oui	Le nombre maximum de fonctions Lambda que aws:lambda:invocation-error peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 140.
Fonctions cibles pour aws:lambda : action. invocation-http-integration-response	Chaque Région prise en charge : 5	Oui	Le nombre maximum de fonctions Lambda que aws:lambda : invocation-http-integration-response peut cibler lorsque vous identifiez des cibles à l'aide de balises, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 140.

Nom	Par défaut	Ajusta	Description
Tables cibles pour aws:dynamodb : action global-table-pause-replication	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tables globales que aws:dynamodb : global-table-pause-replication peut cibler, par expérience. Les demandes d'augmentation de quota seront automatiquement approuvées pour les valeurs allant jusqu'à 60.

Votre utilisation du AWS FIS est soumise aux restrictions supplémentaires suivantes :

Nom	Limitation
Limite quotidienne de cibles ReplicationGroups pour aws:elasticache : replicationgroup-interrupt-az-power	La limite est de 20 ReplicationGroups cibles par compte, par région et par jour. Vous pouvez demander une augmentation en créant un dossier de support dans la console du centre de support AWS.

Historique du document

Le tableau suivant décrit les mises à jour importantes de la documentation du guide de l'utilisateur du service d'injection de AWS défauts.

Modification	Description	Date
Support de l'ARC dans la AWS FIS	Vous pouvez utiliser le AWS FIS pour tester la façon dont ARC Zonal AutoShift rétablit automatiquement votre application lors d'une coupure de courant AZ.	26 mars 2025
Nouvelle configuration du rapport d'expérimentation	Vous pouvez désormais activer le AWS FIS pour générer des rapports pour les expériences qui résument les actions et les réponses des expériences à partir de CloudWatch tableaux de bord.	12 novembre 2024
Nouvelles actions Lambda	Vous pouvez désormais utiliser les actions aws:lambda:function pour injecter des erreurs dans les invocations de vos fonctions Lambda.	31 octobre 2024
Nouvelle fonction de levier de sécurité	AWS Le FIS prend désormais en charge des leviers de sécurité qui vous permettent d'arrêter rapidement toutes les expériences en cours et d'empêcher le démarrage de nouvelles expériences.	3 septembre 2024

Nouveau chapitre sur le dépannage	AWS La FIS a ajouté un guide de dépannage qui inclut les codes d'erreur et le contexte des expériences ayant échoué.	13 août 2024
Nouvelle action	Vous pouvez désormais utiliser cette `aws:dynamodb:global-table-pause-replication` action pour suspendre la réplication des données entre la table globale cible et ses tables de réplication. L'`aws:dynamodb:encrypted-global-table-pause-replication` action ne sera plus prise en charge.	24 avril 2024
Nouvelle option d'expérimentation en mode actions	Vous pouvez définir le mode actions `skip-all` pour générer un aperçu de la cible avant d'exécuter une expérience.	13 mars 2024
AWS mises à jour des politiques gérées	AWS Le FIS a mis à jour les politiques gérées existantes.	25 janvier 2024
Nouveaux scénarios et actions	Vous pouvez désormais utiliser les scénarios AWS FIS Cross-Region:Connectivity et AZ Availability : Power Interruption.	30 novembre 2023

Nouvelle action	Vous pouvez désormais utiliser l'aws:ec2:asg-insufficient-instance-capacity-erroraction.	30 novembre 2023
Nouvelle action	Vous pouvez désormais utiliser l'aws:ec2:api-insufficient-instance-capacity-erroraction.	30 novembre 2023
Nouvelle action	Vous pouvez désormais utiliser l'aws:network:route-table-disrupt-cross-region-connectivityaction.	30 novembre 2023
Nouvelle action	Vous pouvez désormais utiliser l'aws:network:transit-gateway-disrupt-cross-region-connectivityaction.	30 novembre 2023
Nouvelle action	Vous pouvez désormais utiliser l'aws:dynamodb:encrypted-global-table-pause-replicationaction.	30 novembre 2023
Nouvelle action	Vous pouvez désormais utiliser l'aws:s3:bucket-pause-replicationaction.	30 novembre 2023
Nouvelle action	Vous pouvez désormais utiliser l'aws:elasticache:interrupt-cluster-az-poweraction.	30 novembre 2023
Nouvelles options d'expérimentation	Vous pouvez désormais utiliser les options d'expérimentation AWS FIS pour le ciblage des comptes et la résolution des cibles vides.	27 novembre 2023

Changement de nom de la AWS FIS	Nom de service mis à jour en service d'injection de AWS défauts.	15 novembre 2023
AWS mises à jour des politiques gérées	AWS Le FIS a mis à jour les politiques gérées existantes.	13 novembre 2023
Nouvelle bibliothèque de scénarios	Vous pouvez désormais utiliser la fonctionnalité de bibliothèque de scénarios AWS FIS.	7 novembre 2023
Nouveau planificateur d'expériences	Vous pouvez désormais utiliser le AWS planificateur d'expériences FIS.	7 novembre 2023
AWS mises à jour des politiques gérées	AWS Le FIS a mis à jour les politiques gérées existantes.	2 juin 2023
Nouvelles actions	Vous pouvez utiliser aws:ecs:task les nouvelles aws:eks:pod actions.	1er juin 2023
AWS mises à jour des politiques gérées	AWS Le FIS a mis à jour les politiques gérées existantes.	1er juin 2023
Nouveau document SSM préconfiguré	Vous pouvez utiliser le document SSM préconfiguré suivant : AWSFIS-Run -Disk-Fill.	28 avril 2023
Nouvelle action	Vous pouvez utiliser cette aws:ebs:pause-volume-io action pour suspendre les E/S entre les volumes cibles et les instances auxquelles ils sont attachés.	27 janvier 2023

Nouvelle action	Vous pouvez utiliser cette aws:network:disrupt-connectivity action pour refuser certains types de trafic vers les sous-réseaux cibles.	26 octobre 2022
Nouvelle action	Vous pouvez utiliser cette aws:eks:inject-kubernetes-custom-resource action pour exécuter une expérience ChaosMesh ou une expérience Litmus sur un seul cluster cible.	7 juillet 2022
Enregistrement des expériences	Vous pouvez configurer vos modèles de test pour envoyer les journaux d'activité des tests vers CloudWatch Logs ou vers un compartiment S3.	28 février 2022
Nouvelles notifications	Lorsque l'état d'une expérience change, le AWS FIS émet une notification. Ces notifications sont mises à disposition sous forme d'événements via Amazon EventBridge.	24 février 2022
Nouvelle action	Vous pouvez utiliser cette aws:ecs:stop-task action pour arrêter la tâche spécifiée.	9 février 2022
Nouvelle action	Vous pouvez utiliser cette aws:cloudwatch:assert-alarm-state action pour vérifier que les alarmes spécifiées sont dans l'un des états d'alarme spécifiés.	5 novembre 2021

Nouveaux documents SSM préconfigurés	Vous pouvez utiliser les documents SSM préconfig urés suivants : AWSFIS-Ru n -IO-Stress, AWSFIS-Run -Network-Blackhold-Port, - Network-Latency-Sources, -Network-Packet-Loss et - Network-Packet-Loss-Source s. AWSFIS-Run AWSFIS-Run AWSFIS-Run	4 novembre 2021
Nouvelle action	Vous pouvez utiliser cette aws:ec2:send-spot-instance-interruptions action pour envoyer un avis d'interru ption d'une instance Spot aux instances Spot cibles, puis interrompre les instances Spot cibles.	20 octobre 2021
Nouvelle action	Vous pouvez utiliser cette aws:ssm:start-automation-execution action pour lancer l'exécution d'un runbook d'automatisation.	17 septembre 2021
Première version	Version initiale du guide de l'utilisateur du service d'injecti on de AWS défauts.	15 mars 2021

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.