



사용자 가이드

AWS 장애 주입 서비스



AWS 장애 주입 서비스: 사용자 가이드

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

Amazon의 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께 사용하여 고객에게 혼란을 초래하거나 Amazon을 폄하 또는 브랜드 이미지에 악영향을 끼치는 목적으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계와 관계없이 해당 소유자의 자산입니다.

Table of Contents

AWS FIS란 무엇인가요?	1
개념	1
작업	2
대상	2
중지 조건	2
지원되는 AWS 서비스	2
AWS FIS에 액세스	3
요금	4
실험 계획 세우기	5
기본 원칙 및 지침	5
실험 계획 지침	6
튜토리얼	8
테스트 인스턴스 중지 및 시작	8
사전 조건	8
1단계: 실험 템플릿 만들기	8
2단계: 실험 시작	11
3단계: 실험 진행 상황 추적하기	12
4단계: 실험 결과 확인	12
5단계: 정리	12
인스턴스에서 CPU 스트레스 실행	13
사전 조건	13
1단계: 중지 조건에 대한 경보 생성 CloudWatch	14
2단계: 실험 템플릿 만들기	15
3단계: 실험 시작	17
4단계: 실험 진행 상황 추적하기	17
5단계: 실험 결과 확인	17
6단계: 정리	12
스팟 인스턴스 중단 테스트	19
사전 조건	20
1단계: 실험 템플릿 만들기	21
2단계: 실험 시작	23
3단계: 실험 진행 상황 추적하기	24
4단계: 실험 결과 확인	24
5단계: 정리	25

자습서: 연결 이벤트 시뮬레이션	25
사전 조건	26
1단계: AWS FIS 실험 템플릿 만들기	27
2단계: Amazon S3 엔드포인트에 대해 핑 전송	28
3단계: AWS FIS 실험 시작	29
4단계: AWS FIS 실험 진행 상황 추적하기	29
5단계: Amazon S3 네트워크 중단 확인	30
5단계: 정리	30
반복 실험 예약	31
사전 조건	31
1단계: IAM 역할 및 정책 생성	31
2단계: Amazon EventBridge 스케줄러 생성	33
3단계: 실험 확인	34
4단계: 정리	34
작업	35
작업 식별자	35
작업 파라미터	35
작업 대상	36
작업 참조	37
오류 주입 작업	38
대기 작업	40
아마존 CloudWatch 액션	40
Amazon DynamoDB 작업	41
Amazon EBS 작업	43
Amazon EC2 작업	43
Amazon ECS 작업	48
Amazon EKS 작업	55
아마존 ElastiCache 액션	63
네트워크 작업	64
Amazon RDS 작업	68
Amazon S3 작업	69
Systems Manager 작업	70
SSM 문서 사용	72
aws:ssm:send-command 작업을 사용	73
사전 구성된 AWS FIS SSM 문서	74
예제	82

문제 해결	82
ECS 작업 동작 사용	82
작업	83
제한 사항	83
요구 사항	83
스크립트의 참조 버전	86
실험 템플릿 예시	89
EKS 포드 작업 사용	90
작업	90
제한 사항	90
요구 사항	91
Kubernetes 서비스 계정에 대한 서비스 역할 생성	91
Kubernetes 서비스 계정 구성	91
실험 역할을 Kubernetes 사용자에게 매핑	93
포드 컨테이너 이미지	93
실험 템플릿 예시	95
작업 나열	96
실험 템플릿	99
템플릿 구성 요소	99
템플릿 구문	99
시작	100
작업 세트	100
작업 구문	100
작업 기간	101
작업 예	102
대상	104
대상 구문	104
리소스 유형	106
대상 리소스 식별	107
선택 모드	110
예시 대상	110
필터 예제	112
중지 조건	115
중지 조건 구문	116
자세히 알아보기	116
실험 역할	117

사전 조건	117
옵션 1: 실험 역할 생성 및 AWS 관리형 정책 연결	118
옵션 2: 실험 역할 생성 및 인라인 정책 문서 추가	119
실험 옵션	121
계정 타겟팅	122
빈 대상 확인 모드	123
액션 모드	123
실험 템플릿 작업하기	124
실험 템플릿 만들기	124
실험 템플릿 보기	126
실험 템플릿에서 목표 미리 보기를 생성합니다.	127
템플릿에서 실험 시작	128
실험 템플릿 업데이트	129
실험 템플릿에 태그 지정	129
실험 템플릿 만들기	130
템플릿 예제	131
필터를 기반으로 EC2 인스턴스 중지	131
지정된 수의 EC2 인스턴스 중지	132
사전 구성된 AWS FIS SSM 문서 실행	133
사전 정의된 자동화 런북 실행	134
대상 IAM 역할을 사용하여 EC2 인스턴스에서의 API 작업을 제한하세요.	135
Kubernetes 클러스터 내 포드의 CPU 스트레스 테스트	136
다중 계정 실험	139
개념	139
오케스트레이터 계정	139
대상 계정	140
대상 계정 구성	140
필수 조건	140
권한	140
중지 조건(선택 사항)	143
다중 계정 실험 작업	143
모범 사례	143
다중 계정 실험 템플릿 생성	144
대상 계정 구성 업데이트	145
대상 계정 구성 삭제	146
시나리오 라이브러리	147

시나리오 작업	147
시나리오 보기	147
시나리오 사용	148
시나리오 내보내기	149
시나리오 참조	149
AZ Availability: Power Interruption	151
작업	152
제한 사항	154
요구 사항	155
권한	155
시나리오 콘텐츠	159
Cross-Region: Connectivity	164
작업	165
제한 사항	166
요구 사항	166
권한	166
시나리오 콘텐츠	174
실험	177
실험 시작	177
실험 보기	178
실험 상태	178
작업 상태	179
실험에 태그 지정	179
실험 중지	180
확인된 대상 나열	180
실험 스케줄러	181
시작하기	181
FIS 실험 예약	185
콘솔을 사용하여 일정을 업데이트하려면	185
실험 일정 업데이트	186
콘솔을 사용하여 실험 실행을 비활성화하거나 삭제	186
모니터링	188
CloudWatch를 사용한 모니터링	189
AWS FIS 실험 모니터링	189
AWS FIS 사용량 지표	190
다음을 사용하여 모니터링하십시오. EventBridge	191

실험 로깅	192
권한	192
로그 스키마	193
로그 대상	194
로그 레코드 예	195
실험 로깅 활성화	199
실험 로깅 비활성화	200
AWS CloudTrail을 사용하여 API 호출 로그	200
사용 CloudTrail	201
AWS FIS 로그 파일 항목 이해	202
보안	206
데이터 보호	206
저장된 데이터 암호화	207
전송 중 암호화	207
자격 증명 및 액세스 관리	208
고객	208
자격 증명을 통한 인증	208
정책을 사용한 액세스 관리	212
AWS 장애 인젝션 서비스가 IAM과 함께 작동하는 방식	214
정책 예제	220
서비스 연결 역할 사용	232
AWS 관리형 정책	234
인프라 보안	238
AWS PrivateLink	239
고려 사항	239
인터페이스 VPC 엔드포인트 생성	239
VPC 엔드포인트 정책 생성	240
리소스에 태그 지정	242
태그 지정 제한	242
태그 작업	242
제한 사항 및 할당량	244
문서 기록	253
.....	cclvii

AWS Fault Injection Service란 무엇인가요?

AWS Fault Injection Service(AWS FIS)는 AWS 워크로드에 대한 오류 주입 실험을 수행하는 데 사용할 수 있는 관리형 서비스입니다. 오류 주입은 카오스 엔지니어링의 원칙을 기반으로 합니다. 이러한 실험은 애플리케이션이 어떻게 반응하는지 관찰할 수 있도록 방해 이벤트를 생성하여 애플리케이션에 스트레스를 줍니다. 그런 다음 이 정보를 사용하여 애플리케이션이 예상대로 작동하도록 애플리케이션의 성능과 복원력을 개선할 수 있습니다.

AWS FIS를 사용하려면 다른 방법으로는 찾기 어려운 애플리케이션 문제를 발견하는 데 필요한 실제 조건을 만드는 실험을 설계하고 실행해야 합니다. AWS FIS는 중단을 유발하는 템플릿과 특정 조건이 충족될 경우 실험을 자동으로 롤백하거나 중지하는 등, 프로덕션 환경에서 실험을 실행하는 데 필요한 제어 및 가드레일을 제공합니다.

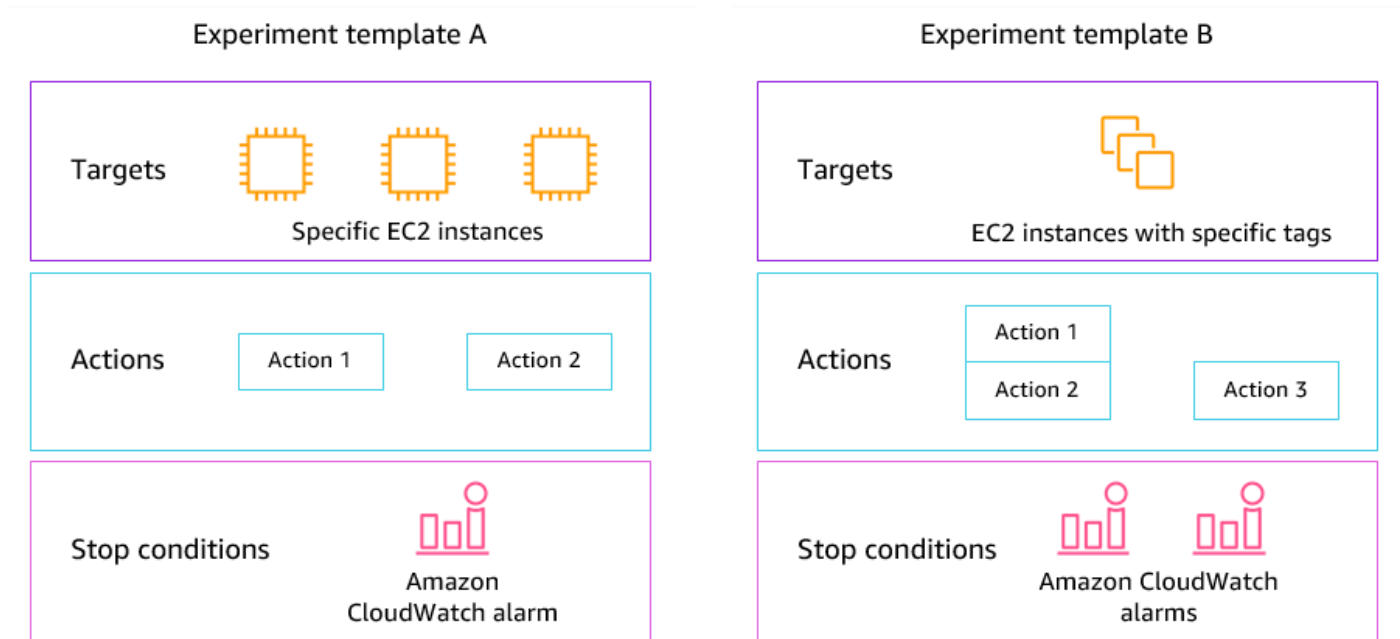
Important

AWS FIS는 시스템의 실제 AWS 리소스에 대해 실제 조치를 취합니다. 따라서 AWS FIS를 사용하여 프로덕션에서 실험을 실행하기 전에 먼저 계획 단계를 완료하고 사전 프로덕션 환경에서 실험을 실행하는 것이 좋습니다.

실험 계획에 대한 자세한 내용은 [테스트 신뢰성 및 AWS FIS 실험 계획 세우기](#)를 참조하세요. AWS FIS에 대한 자세한 내용은 [AWS Fault Injection Service](#)를 참조하세요.

AWS FIS 개념

AWS FIS를 사용하려면 AWS 리소스에 대한 실험을 실행하여 오류 조건에서 애플리케이션 또는 시스템이 어떻게 작동하는지에 대한 이론을 테스트합니다. 실험을 실행하려면 먼저 실험 템플릿을 만들어야 합니다. 실험 템플릿은 실험의 청사진입니다. 여기에는 실험에서 수행할 작업, 목표, 중지 조건이 포함됩니다. 실험 템플릿을 만든 후 이를 사용하여 실험을 실행할 수 있습니다. 실험이 실행되는 동안 진행 상황을 추적하고 상태를 볼 수 있습니다. 실험의 모든 작업이 실행되면 실험이 완료된 것입니다.



작업

작업은 AWS FIS가 실험 중에 AWS 리소스에 대해 수행하는 활동입니다. AWS FIS는 AWS 리소스 유형에 따라 사전 구성된 일련의 작업을 제공합니다. 각 작업은 실험 중 지정된 기간 동안 또는 실험을 중지할 때까지 실행됩니다. 작업은 순차적으로 또는 동시에 (병렬) 실행할 수 있습니다.

대상

대상은 AWS FIS가 실험 중에 작업을 수행하는 하나 이상의 AWS 리소스입니다. 특정 리소스를 선택하거나 태그 또는 상태와 같은 특정 기준에 따라 리소스 그룹을 선택할 수 있습니다.

중지 조건

AWS FIS는 AWS 워크로드에서 안전하게 실험을 실행하는 데 필요한 제어 및 가드레일을 제공합니다. 중지 조건은 Amazon CloudWatch 경보로 정의한 임계값에 도달할 경우 실험을 중단하는 메커니즘입니다. 실험이 실행되는 동안 중지 조건이 트리거되면 AWS FIS에서 실험을 중지합니다.

지원되는 AWS 서비스

AWS FIS는 AWS 서비스 전반에 걸쳐 특정 유형의 대상에게 사전 구성된 작업을 제공합니다. AWS FIS는 다음 AWS 서비스의 대상 자원에 대한 작업을 지원합니다.

- 아마존 CloudWatch

- Amazon EBS
- Amazon EC2
- Amazon ECS
- Amazon EKS
- 아마존 ElastiCache
- Amazon RDS
- AWS Systems Manager
- Amazon VPC

단일 계정 실험의 경우, 대상 리소스는 실험과 동일한 AWS 계정 계정에 있어야 합니다. AWS FIS 다중 계정 실험을 사용하여 다른 AWS 계정 계정의 리소스를 대상으로 하는 AWS FIS 실험을 실행할 수 있습니다.

자세한 설명은 [에 대한 조치 AWS FIS](#) 섹션을 참조하세요.

AWS FIS에 액세스

다음 방법 중 하나를 사용하여 AWS FIS에서 작업할 수 있습니다.

- AWS Management Console - AWS FIS에 액세스할 때 사용할 수 있는 웹 인터페이스를 제공합니다. 자세한 내용은 [AWS Management Console 작업](#) 단원을 참조하세요.
- AWS Command Line Interface (AWS CLI) - AWS FIS를 포함한 다양한 AWS 서비스에서 사용되는 명령을 제공하며 Windows, macOS 및 Linux에서 지원됩니다. 자세한 설명은 [AWS Command Line Interface](#) 섹션을 참조하세요. AWS FIS 명령에 대한 자세한 내용은 AWS CLI 명령 참조의 [fis](#)를 참조하세요.
- AWS CloudFormation - AWS 리소스를 설명하는 템플릿을 생성합니다. 템플릿을 사용하여 이러한 리소스를 하나의 단위로 프로비저닝하고 관리할 수 있습니다. 자세한 내용은 [AWS Fault Injection Service 리소스 유형 참조](#)를 참조하세요.
- AWS SDK - 언어별 API를 제공하고, 서명 계산, 요청 재시도 처리 및 오류 처리와 같은 많은 연결 세부 정보를 관리합니다. 자세한 내용은 [AWS SDK](#) 섹션을 참조하세요.
- HTTPS API - HTTPS 요청을 사용하여 직접 호출할 수 있는 하위 수준의 API 작업을 제공합니다. 자세한 내용은 [AWS Fault Injection Service API 참조](#)를 참조하세요.

AWS FIS 요금

실험의 대상 계정 수에 따라 작업이 시작부터 끝까지 실행되는 분당 요금이 부과됩니다. 자세한 내용은 [AWS FIS 요금](#)을 참조하세요.

AWS FIS 실험 계획 세우기

오류 주입은 서버 중단이나 API 제한과 같은 운영 중단 이벤트를 발생시켜 테스트 또는 프로덕션 환경에서 애플리케이션에 스트레스를 주는 프로세스입니다. 시스템이 어떻게 반응하는지 관찰한 다음 개선을 구현할 수 있습니다. 시스템에서 실험을 실행하면 시스템에 의존하는 고객에게 영향을 미치지 전에 통제된 방식으로 시스템상의 약점을 식별하는 데 도움이 될 수 있습니다. 그러면 문제를 사전에 해결하여 예측할 수 없는 결과를 예방할 수 있습니다.

AWS FIS를 사용하여 오류 주입 실험을 실행하기 전에 다음 원칙과 지침을 숙지하는 것이 좋습니다.

Important

AWS FIS는 시스템의 실제 AWS 리소스에 대해 실제 조치를 취합니다. 따라서 AWS FIS를 사용하여 실험을 시작하기 전에 먼저 계획 단계를 완료하고 사전 프로덕션 또는 테스트 환경에서 테스트하는 것이 좋습니다.

내용

- [기본 원칙 및 지침](#)
- [실험 계획 지침](#)

기본 원칙 및 지침

AWS FIS 실험을 시작하기 전에 다음 단계를 수행합니다.

1. 실험 대상 배포 식별 - 먼저 대상 배포를 식별하세요. 실험이 처음인 경우 사전 프로덕션 또는 테스트 환경에서 시작하는 것이 좋습니다.
2. 애플리케이션 아키텍처 검토 - 각 구성 요소의 애플리케이션 구성 요소, 종속성 및 복구 절차를 모두 식별했는지 확인해야 합니다. 먼저 애플리케이션 아키텍처를 검토하세요. 애플리케이션에 따라 [AWS Well-Architected Framework](#)를 참조하세요.
3. 정상 상태 동작 정의 - 지연 시간, CPU 부하, 분당 로그인 실패, 재시도 횟수, 페이지 로드 속도 등 중요한 기술 및 비즈니스 지표를 기준으로 시스템의 정상 상태 동작을 정의합니다.
4. 가설 세우기 - 실험 중에 시스템 동작이 어떻게 변할 것으로 예상하는지에 대한 가설을 세우세요. 가설 정의는 다음과 같은 형식을 따릅니다.

##이 수행되는 경우 ##### ## ## ### ## ##이 #을 초과해서는 안 됩니다.

예를 들어 인증 서비스에 대한 가설은 다음과 같을 수 있습니다. “네트워크 지연 시간이 10% 증가할 경우 로그인 실패 증가는 1% 미만입니다.” 실험이 완료된 후 애플리케이션 복원력이 비즈니스 및 기술 기대치에 부합하는지 평가합니다.

또한 AWS FIS를 사용할 때는 다음 지침을 따르는 것이 좋습니다.

- 항상 테스트 환경에서 AWS FIS 실험을 시작하세요. 프로덕션 환경에서는 절대 시작하지 마세요. 오류 주입 실험이 진행됨에 따라 테스트 환경 이외의 다른 통제된 환경에서도 실험할 수 있습니다.
- 하나의 대상에서 `aws:ec2:stop-instances` 작업을 실행하는 것과 같이 작고 간단한 실험부터 시작하여 애플리케이션 복원력에 대한 팀의 신뢰를 구축하세요.
- 오류 주입으로 인해 실제 문제가 발생할 수 있습니다. 주의를 기울여 진행하고 고객이 영향을 받지 않도록 테스트 인스턴스에 첫 번째 오류 주입이 이루어지도록 하세요.
- 테스트를 거듭하고 몇 가지 더 테스트해 보세요. 오류 주입은 잘 계획된 실험이 포함된 통제된 환경에서 구현되어야 합니다. 이를 통해 난류 조건을 견딜 수 있는 애플리케이션 및 도구의 성능에 대한 확신을 구축할 수 있습니다.
- 시작하기 전에 우수한 모니터링 및 경고 프로그램을 마련해 두는 것이 좋습니다. 이 기능이 없으면 실험의 영향을 이해하거나 측정할 수 없는데, 이는 지속 가능한 오류 주입 관행에 매우 중요한 요소입니다.

실험 계획 지침

AWS FIS를 사용하면 AWS 리소스에 대한 실험을 실행하여 오류 조건에서 애플리케이션 또는 시스템이 어떻게 작동하는지에 대한 이론을 테스트할 수 있습니다.

다음은 AWS FIS 실험 계획을 위한 권장 지침입니다.

- 정전 기록 검토 - 시스템의 과거 정전 및 이벤트를 검토하세요. 이를 통해 시스템의 전반적인 상태와 복원력을 파악할 수 있습니다. 시스템에서 실험을 시작하기 전에 시스템의 알려진 문제와 약점을 해결해야 합니다.
- 가장 큰 영향을 미치는 서비스 식별 - 서비스를 검토하고 서비스가 중단되거나 제대로 작동하지 않을 경우 최종 사용자나 고객에게 가장 큰 영향을 미치는 서비스를 식별하세요.
- 대상 시스템 식별 - 대상 시스템은 실험을 실행할 시스템입니다. AWS FIS를 처음 사용하거나 오류 주입 실험을 한 번도 해본 적이 없는 경우에는 사전 프로덕션 또는 테스트 시스템에서 실험을 실행하는 것으로 시작하는 것이 좋습니다.

- 팀원들과 논의 - 어떤 걱정을 하고 있는지 물어보세요. 고객의 우려를 입증하거나 반증하기 위한 가설을 세울 수 있습니다. 팀원들이 걱정하지 않는 것은 무엇인지 물어볼 수도 있습니다. 이 질문을 통해 두 가지 일반적인 오류, 즉 매몰 비용 오류와 확증 편향 오류가 드러날 수 있습니다. 팀원들의 답변을 기반으로 가설을 세우면 시스템 상태의 현실에 대한 자세한 정보를 제공하는 데 도움이 될 수 있습니다.
- 애플리케이션 아키텍처 검토 - 시스템 또는 애플리케이션을 검토하고 애플리케이션의 모든 구성 요소, 종속성 및 각 구성 요소의 복구 절차를 모두 식별했는지 확인하세요.

AWS Well-Architected Framework를 검토하는 것이 좋습니다. 이 프레임워크는 애플리케이션과 워크로드를 위한 안전하고 성능 및 복원력이 뛰어나며 효율적인 인프라를 구축할 수 있도록 지원합니다. 자세한 내용은 [AWS Well-Architected](#)를 참조하세요.

- 적용 가능한 지표 식별 — Amazon CloudWatch 지표를 사용하여 실험이 AWS 리소스에 미치는 영향을 모니터링할 수 있습니다. 이러한 지표를 사용하여 애플리케이션이 최적의 성능을 발휘할 때 기준선 또는 “안정 상태”를 판단할 수 있습니다. 그런 다음 실험 중 또는 실험 후에 이러한 지표를 모니터링하여 영향을 확인할 수 있습니다. 자세한 설명은 [Amazon CloudWatch를 사용한 AWS FIS 사용량 지표 모니터링](#) 섹션을 참조하세요.
- 시스템에 적합한 성능 임계값 정의 - 시스템에 허용 가능한 안정 상태를 나타내는 지표를 식별하세요. 이 지표를 사용하여 실험 중지 조건을 나타내는 하나 이상의 CloudWatch 경보를 생성할 수 있습니다. 경보가 트리거되면 실험이 자동으로 중지됩니다. 자세한 내용은 [AWS FIS의 중지 조건을\(를\)](#) 참조하세요.

오류 인젝션 서비스 튜토리얼 AWS

다음 자습서에서는 AWS 오류 주입 서비스 (AWS FIS) 를 사용하여 실험을 만들고 실행하는 방법을 보여줍니다.

튜토리얼

- [자습서: 테스트 인스턴스 중지 및 AWS FIS 사용 시작](#)
- [자습서: AWS FIS를 사용하여 인스턴스에서 CPU 스트레스 실행](#)
- [자습서: AWS FIS를 사용한 스팟 인스턴스 중단 테스트](#)
- [자습서: 연결 이벤트 시뮬레이션](#)
- [자습서: 반복 실험 예약](#)

자습서: 테스트 인스턴스 중지 및 AWS FIS 사용 시작

AWS Fault Injection Service(AWS FIS)를 사용하여 애플리케이션이 인스턴스 중지 및 시작을 처리하는 방법을 테스트할 수 있습니다. 이 자습서에 설명된 AWS FIS `aws:ec2:stop-instances` 작업을 사용하여 인스턴스 중 하나를 중지한 후 두 번째 인스턴스를 중지할 수 있습니다.

사전 조건

이 자습서를 완료하려면 다음을 수행합니다.

- 계정에서 두 개의 테스트 EC2 인스턴스를 시작합니다. 인스턴스를 시작한 후 두 인스턴스의 ID를 기록합니다.
- AWS FIS 서비스가 사용자를 대신하여 `aws:ec2:stop-instances` 작업을 수행할 수 있도록 하는 IAM 역할을 생성하세요. 자세한 설명은 [AWS FIS 실험을 위한 IAM 역할](#) 섹션을 참조하세요.
- AWS FIS 액세스 권한이 있어야 합니다. 자세한 내용을 알아보려면 [AWS FIS 정책 예제](#)를 참조하세요.

1단계: 실험 템플릿 만들기

AWS FIS 콘솔을 사용하여 실험 템플릿을 생성합니다. 템플릿에서 각각 3분 동안 순차적으로 실행되는 두 개의 작업을 지정합니다. 첫 번째 작업은 AWS FIS가 임의로 선택한 테스트 인스턴스 중 하나를 중지합니다. 두 번째 작업은 두 테스트 인스턴스를 모두 중지합니다.

실험 템플릿 만들기

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿 생성을 선택합니다.
4. 이름 및 설명에 템플릿에 대한 이름과 설명을 입력합니다.
5. 작업에서 다음을 수행합니다.
 - a. 작업 추가를 선택합니다.
 - b. 작업의 이름을 입력합니다. 예를 들면 **stopOneInstance**를 입력합니다.
 - c. 작업 유형으로는 `aws:ec2:stop-instances`를 선택합니다.
 - d. 대상의 경우 AWS FIS가 생성한 대상을 그대로 유지하세요.
 - e. 작업 파라미터의 경우 기간 이후 인스턴스 시작은 3분(PT3M)으로 지정합니다.
 - f. 저장을 선택합니다.
6. 대상에서 다음을 수행합니다.
 - a. AWS FIS가 이전 단계에서 자동으로 생성한 대상에 대해 편집을 선택합니다.
 - b. 기본 이름을 좀 더 이해하기 쉬운 이름으로 바꾸세요. 예를 들면 **oneRandomInstance**를 입력합니다.
 - c. 리소스 유형이 `aws:ec2:instance`인지 확인하세요.
 - d. 대상 메서드의 경우 리소스 ID를 선택한 다음, 두 테스트 인스턴스의 ID를 선택합니다.
 - e. 선택 모드에서는 개수를 선택합니다. 리소스 수에 **1**를 입력합니다.
 - f. 저장을 선택합니다.
7. 대상 추가를 선택하고 다음과 같이 합니다.
 - a. 대상의 이름을 입력합니다. 예를 들면 **bothInstances**를 입력합니다.
 - b. 리소스 유형에서 `aws:ec2:instance`를 선택합니다.
 - c. 대상 메서드의 경우 리소스 ID를 선택한 다음, 두 테스트 인스턴스의 ID를 선택합니다.
 - d. 선택 모드에서는 모두를 선택합니다.
 - e. 저장을 선택합니다.
8. 작업 섹션에서 작업 추가를 선택합니다. 다음을 따릅니다.
 - a. 이름에서 작업의 이름을 입력합니다. 예를 들면 **stopBothInstances**를 입력합니다.
 - b. 작업 유형으로는 `aws:ec2:stop-instances`를 선택합니다.

- c. 시작하기 전에서 추가한 첫 번째 작업(**stopOneInstance**)을 선택합니다.
 - d. 대상에서, 추가한 두 번째 대상(**bothInstances**)을 선택합니다.
 - e. 작업 파라미터의 경우 기간 이후 인스턴스 시작은 3분(PT3M)으로 지정합니다.
 - f. 저장을 선택합니다.
9. 서비스 액세스의 경우 기존 IAM 역할 사용을 선택한 다음 이 자습서의 사전 조건에 설명된 대로 생성한 IAM 역할을 선택합니다. 역할이 표시되지 않는 경우 해당 역할에 필요한 신뢰 관계가 있는지 확인하세요. 자세한 설명은 [the section called “실험 역할”](#) 섹션을 참조하세요.
 10. (선택 사항) 태그의 경우 새 태그 추가를 선택하고 태그 키와 태그 값을 지정합니다. 추가한 태그는 템플릿을 사용하여 실행되는 실험이 아니라 실험 템플릿에 적용됩니다.
 11. 실험 템플릿 생성을 선택합니다. 확인 메시지가 나타나면 **create**을 입력하고 실험 템플릿 생성을 선택합니다.

(선택사항) 실험 템플릿 JSON을 보려면

내보내기 탭을 선택합니다. 다음은 이전 콘솔 절차에서 생성한 JSON 예제입니다.

```
{
  "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": {}
}

```

2단계: 실험 시작

실험 템플릿 생성을 완료하면 이를 사용하여 실험을 시작할 수 있습니다.

실험을 시작하려면

1. 방금 만든 실험 템플릿의 세부정보 페이지로 이동해야 합니다. 그렇지 않으면 실험 템플릿을 선택한 다음 실험 템플릿의 ID를 선택하여 세부 정보 페이지를 엽니다.
2. 실험 시작을 선택합니다.
3. (선택 사항) 실험에 태그를 추가하려면 새 태그 추가를 선택하고 태그 키와 태그 값을 입력합니다.
4. 실험 시작을 선택합니다. 확인 메시지가 나타나면 **start**을 입력하고 실험 시작을 선택합니다.

3단계: 실험 진행 상황 추적하기

실험이 완료, 중지 또는 실패할 때까지 진행 중인 실험의 진행 상황을 추적할 수 있습니다.

실험 진행 상황 추적하기

1. 방금 시작한 실험의 세부정보 페이지로 이동해야 합니다. 그렇지 않으면 실험을 선택한 다음 실험의 ID를 선택하여 세부 정보 페이지를 엽니다.
2. 실험 상태를 보려면 세부 정보 창에서 상태를 확인하세요. 자세한 내용은 [실험 상태](#)를 참조하세요.
3. 실험 상태가 실행 중이면 다음 단계로 이동합니다.

4단계: 실험 결과 확인

예상대로 실험에 의해 인스턴스가 중지되고 시작되었는지 확인할 수 있습니다.

실험 결과를 확인하려면

1. 새 브라우저 탭 또는 창에 <https://console.aws.amazon.com/ec2/>에서 Amazon EC2 콘솔을 엽니다. 이렇게 하면 Amazon EC2 콘솔에서 실험 결과를 보면서 AWS FIS 콘솔에서 실험 진행 상황을 계속 추적할 수 있습니다.
2. 탐색 창에서 인스턴스를 선택합니다.
3. 첫 번째 작업의 상태가 보류 중에서 실행 중(AWS FIS 콘솔)으로 변경되면 대상 인스턴스 중 하나의 상태가 실행 중에서 중지됨으로 변경됩니다(Amazon EC2 콘솔).
4. 3분 후 첫 번째 작업의 상태는 완료됨으로 변경되고, 두 번째 작업의 상태는 실행 중으로 변경되며, 다른 대상 인스턴스의 상태는 중지됨으로 변경됩니다.
5. 3분 후 두 번째 작업의 상태는 완료됨으로 변경되고, 대상 인스턴스의 상태는 실행 중으로 변경되며, 실험의 상태는 완료됨으로 변경됩니다.

5단계: 정리

이 실험용으로 생성한 테스트 EC2 인스턴스가 더 이상 필요하지 않으면 종료할 수 있습니다.

인스턴스를 종료하려면

1. <https://console.aws.amazon.com/ec2/>에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택합니다.

3. 테스트 인스턴스를 모두 선택하고 Instance state(인스턴스 상태), Terminate instance(인스턴스 종료)를 선택합니다.
4. 확인 메시지가 나타나면 종료를 선택합니다.

실험 템플릿이 더 이상 필요하지 않으면 삭제할 수 있습니다.

AWS FIS 콘솔을 사용하여 실험 템플릿을 삭제하려면

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 삭제를 선택합니다.
4. 확인 메시지가 나타나면 **delete**를 입력한 다음 실험 템플릿 삭제를 선택합니다.

자습서: AWS FIS를 사용하여 인스턴스에서 CPU 스트레스 실행

AWS Fault Injection Service(AWS FIS)를 사용하여 애플리케이션이 CPU 스트레스를 처리하는 방법을 테스트할 수 있습니다. 이 자습서를 사용하여 인스턴스에서 CPU 스트레스를 실행하는 사전 구성된 SSM 문서를 실행하기 위해 AWS FIS를 사용하는 실험 템플릿을 만들 수 있습니다. 이 자습서에서는 인스턴스의 CPU 사용률이 구성된 임계값을 초과할 경우 중지 조건을 사용하여 실험을 중단합니다.

자세한 설명은 [the section called “사전 구성된 AWS FIS SSM 문서”](#) 섹션을 참조하세요.

사전 조건

AWS FIS를 사용하여 CPU 스트레스를 실행하려면 먼저 다음 사전 조건을 완료하세요.

IAM 역할 생성

역할을 생성하고 AWS FIS가 사용자 대신 `aws:ssm:send-command` 작업을 사용할 수 있도록 하는 정책을 첨부하세요. 자세한 설명은 [AWS FIS 실험을 위한 IAM 역할](#) 섹션을 참조하세요.

AWS FIS에 대한 액세스 권한 확인

AWS FIS 액세스 권한이 있어야 합니다. 자세한 내용을 알아보려면 [AWS FIS 정책 예제](#)를 참조하세요.

테스트 EC2 인스턴스 준비

- 사전 구성된 SSM 문서의 요구에 따라 Amazon Linux 2 또는 Ubuntu를 사용하여 EC2 인스턴스를 시작합니다.

- 인스턴스는 SSM으로 관리해야 합니다. 인스턴스가 SSM으로 관리되는지 확인하려면 [플릿 관리자 콘솔](#)을 여세요. 인스턴스가 SSM으로 관리되지 않는 경우 SSM 에이전트가 설치되어 있고 인스턴스에 AmazonSSM 정책에 따라 연결된 IAM 역할이 있는지 확인하십시오. ManagedInstanceCore 설치된 SSM 에이전트를 확인하려면 인스턴스에 연결하고 다음 명령을 실행합니다.

Amazon Linux 2

```
yum info amazon-ssm-agent
```

Ubuntu

```
apt list amazon-ssm-agent
```

- 인스턴스에 대한 세부 모니터링을 활성화합니다. 추가 비용을 지불하면 데이터를 1분 기간으로 사용할 수 있습니다. 인스턴스를 선택하고 작업, 모니터링 및 문제 해결, 세부 모니터링 관리를 선택합니다.

1단계: 중지 조건에 대한 경보 생성 CloudWatch

CPU 사용률이 지정한 임계값을 초과하는 경우 실험을 중단할 수 있도록 CloudWatch 경보를 구성하십시오. 다음 절차는 대상 인스턴스의 CPU 사용률 50%로 임계값을 설정합니다. 자세한 설명은 [중지 조건](#) 섹션을 참조하세요.

CPU 사용률이 임계값을 초과할 때 알려주는 경보를 만들려면

1. <https://console.aws.amazon.com/ec2/>에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택합니다.
3. 대상 인스턴스를 선택하고 작업, 모니터링 및 문제 해결, CloudWatch 경보 관리를 선택합니다.
4. 경보 알림에서 토글을 사용하여 Amazon SNS 알림을 끕니다.
5. 경보 임계값의 경우 다음 설정을 사용하세요.
 - 샘플 그룹화 기준: 최대값
 - 샘플링할 데이터 유형: CPU 사용률
 - 퍼센트: **50**
 - 기간: **1 Minute**
6. 경보 구성이 완료되면 생성을 선택합니다.

2단계: 실험 템플릿 만들기

AWS FIS 콘솔을 사용하여 실험 템플릿을 생성합니다. 템플릿에서 다음과 같은 작업을 실행하도록 지정합니다. [AWSFISaws:ssm:send-command/ -run-cpu-stress](#).

실험 템플릿 만들기

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿 생성을 선택합니다.
4. 이름 및 설명에 템플릿에 대한 이름과 설명을 입력합니다.
5. 작업에서 다음을 수행합니다.
 - a. 작업 추가를 선택합니다.
 - b. 작업의 이름을 입력합니다. 예를 들면 **runCpuStress**를 입력합니다.
 - c. 작업 유형으로는 `aws:ssm:send-command/ -run-cpu-stress`를 선택합니다. AWSFIS 그러면 SSM 문서의 ARN이 문서 ARN에 자동으로 추가됩니다.
 - d. 대상의 경우 AWS FIS가 생성한 대상을 그대로 유지하세요.
 - e. 작업 파라미터, 문서 파라미터에 다음을 입력합니다.

```
{"DurationSeconds":"120"}
```

- f. 작업 파라미터 지속 시간에 5분(PT5M)을 지정합니다.
- g. 저장을 선택합니다.
6. 대상에서 다음을 수행합니다.
 - a. AWS FIS가 이전 단계에서 자동으로 생성한 대상에 대해 편집을 선택합니다.
 - b. 기본 이름을 좀 더 이해하기 쉬운 이름으로 바꾸세요. 예를 들면 **testInstance**를 입력합니다.
 - c. 리소스 유형이 `aws:ec2:instance`인지 확인하세요.
 - d. 대상 메서드의 경우 리소스 ID를 선택한 다음 테스트 인스턴스의 ID를 선택합니다.
 - e. 선택 모드에서는 모두를 선택합니다.
 - f. 저장을 선택합니다.

7. 서비스 액세스의 경우 기존 IAM 역할 사용을 선택한 다음 이 자습서의 사전 조건에 설명된 대로 생성한 IAM 역할을 선택합니다. 역할이 표시되지 않는 경우 해당 역할에 필요한 신뢰 관계가 있는지 확인하세요. 자세한 설명은 [the section called “실험 역할”](#) 섹션을 참조하세요.
8. 중지 조건의 경우 1단계에서 생성한 CloudWatch 경보를 선택합니다.
9. (선택 사항) 태그의 경우 새 태그 추가를 선택하고 태그 키와 태그 값을 지정합니다. 추가한 태그는 템플릿을 사용하여 실행되는 실험이 아니라 실험 템플릿에 적용됩니다.
10. 실험 템플릿 생성을 선택합니다.

(선택사항) 실험 템플릿 JSON을 보려면

내보내기 탭을 선택합니다. 다음은 이전 콘솔 절차에서 생성한 JSON 예제입니다.

```
{
  "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": {}  
}
```

3단계: 실험 시작

실험 템플릿 생성을 완료하면 이를 사용하여 실험을 시작할 수 있습니다.

실험을 시작하려면

1. 방금 만든 실험 템플릿의 세부정보 페이지로 이동해야 합니다. 그렇지 않으면 실험 템플릿을 선택한 다음 실험 템플릿의 ID를 선택하여 세부 정보 페이지를 엽니다.
2. 실험 시작을 선택합니다.
3. (선택 사항) 실험에 태그를 추가하려면 새 태그 추가를 선택하고 태그 키와 태그 값을 입력합니다.
4. 실험 시작을 선택합니다. 확인 메시지가 표시되면 **start**를 입력합니다. 실험 시작을 선택합니다.

4단계: 실험 진행 상황 추적하기

실험이 완료, 중지 또는 실패할 때까지 진행 중인 실험의 진행 상황을 추적할 수 있습니다.

실험 진행 상황 추적하기

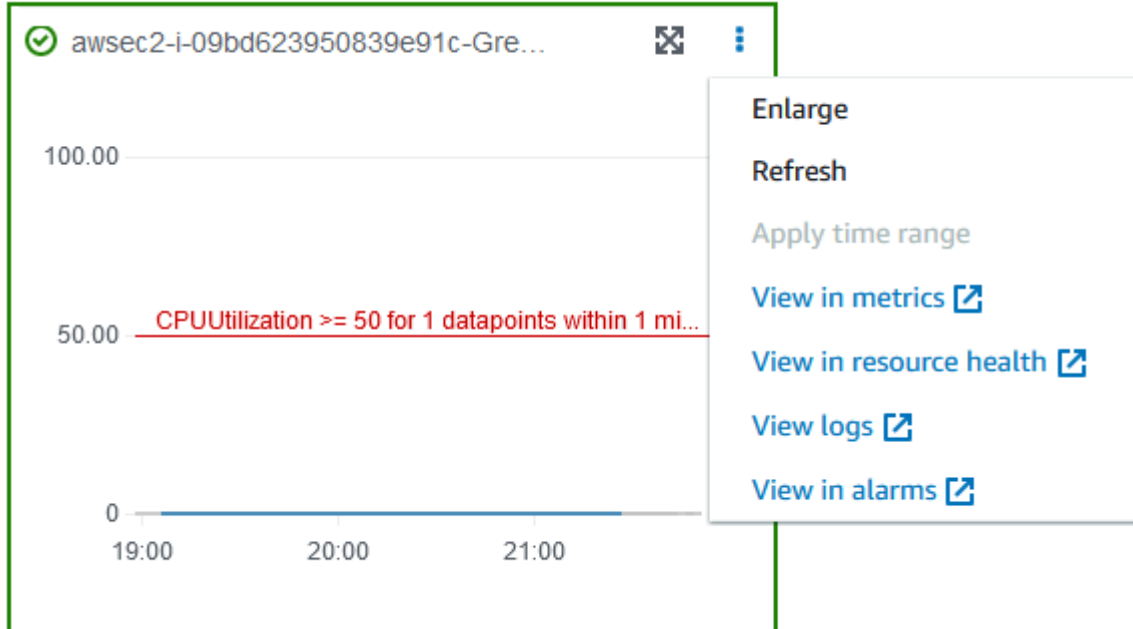
1. 방금 시작한 실험의 세부정보 페이지로 이동해야 합니다. 그렇지 않으면 실험을 선택한 다음 실험의 ID를 선택하여 실험의 세부 정보 페이지를 엽니다.
2. 실험 상태를 보려면 세부 정보 창에서 상태를 확인하세요. 자세한 내용은 [실험 상태](#)를 참조하세요.
3. 실험 상태가 실행 중으로 표시되면 다음 단계로 이동합니다.

5단계: 실험 결과 확인

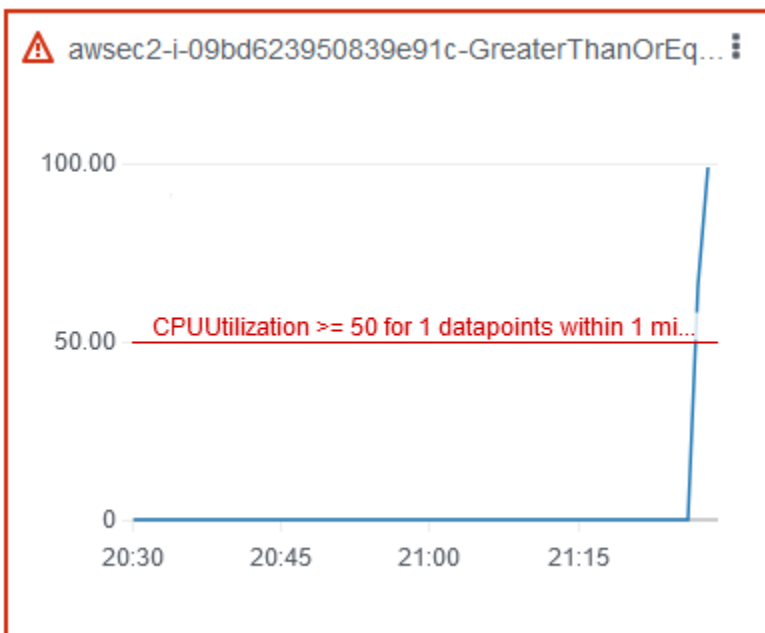
실험이 실행되는 동안 인스턴스의 CPU 사용률을 모니터링할 수 있습니다. CPU 사용률이 임계값에 도달하면 경보가 트리거되고 중지 조건에 따라 실험이 중단됩니다.

실험 결과를 확인하려면

1. 중지 조건 탭을 선택합니다. 녹색 테두리와 녹색 체크 표시 아이콘은 경보의 초기 상태가 OK임을 나타냅니다. 빨간색 선은 경보 임계값을 나타냅니다. 더 자세한 그래프를 원하면 위젯 메뉴에서 확대를 선택합니다.



2. CPU 사용률이 임계값을 초과하면 중지 조건 탭의 빨간색 테두리와 빨간색 느낌표 아이콘이 경보 상태가 ALARM로 변경되었음을 나타냅니다. 세부 정보 창에서는 실험 상태가 중지됨입니다. 상태를 선택하면 “중지 조건에 따라 실험이 중단됨”이라는 메시지가 표시됩니다.



3. CPU 사용률이 임계값 아래로 감소하면 녹색 테두리와 녹색 체크 표시 아이콘은 경보 상태가 OK로 변경되었음을 나타냅니다.
4. (선택 사항) 위젯 메뉴에서 경보에서 보기를 선택합니다. 그러면 CloudWatch 콘솔에 경보 세부 정보 페이지가 열리고 경보에 대한 자세한 내용을 확인하거나 경보 설정을 편집할 수 있습니다.

6단계: 정리

이 실험용으로 생성한 테스트 EC2 인스턴스가 더 이상 필요하지 않으면 종료할 수 있습니다.

인스턴스를 해지하려면

1. <https://console.aws.amazon.com/ec2/>에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택합니다.
3. 테스트 인스턴스를 선택하고 인스턴스 상태, 인스턴스 종료를 선택합니다.
4. 확인 메시지가 나타나면 종료를 선택합니다.

실험 템플릿이 더 이상 필요하지 않으면 삭제할 수 있습니다.

AWS FIS 콘솔을 사용하여 실험 템플릿을 삭제하려면

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 삭제를 선택합니다.
4. 확인 메시지가 나타나면 **delete**를 입력한 다음 실험 템플릿 삭제를 선택합니다.

자습서: AWS FIS를 사용한 스팟 인스턴스 중단 테스트

스팟 인스턴스는 온디맨드 요금과 비교하여 최대 90% 할인된 가격으로 제공되는 예비 EC2 용량을 사용합니다. 그러나 Amazon EC2는 용량이 다시 필요할 때 스팟 인스턴스를 중단할 수 있습니다. 스팟 인스턴스를 사용할 때는 중단 가능성에 대비해야 합니다. 자세한 내용은 Amazon EC2 사용 설명서의 [스팟 인스턴스 중단](#)을 참조하세요.

AWS Fault Injection Service(AWS FIS)를 사용하여 애플리케이션이 스팟 인스턴스 중단을 처리하는 방법을 테스트할 수 있습니다. 이 자습서에 설명된 AWS FIS `aws:ec2:send-spot-instance-interruptions` 작업을 사용하여 스팟 인스턴스 중 하나를 중단하는 실험 템플릿을 만들 수 있습니다.

또는 Amazon EC2 콘솔을 사용하여 실험을 시작하려면 Amazon EC2 사용 설명서의 [스팟 인스턴스 중단 시작](#)을 참조하세요.

사전 조건

AWS FIS를 사용하여 스팟 인스턴스를 중단하려면 먼저 다음 사전 조건을 완료하세요.

1. IAM 역할 생성

역할을 생성하고 AWS FIS가 사용자 대신 `aws:ec2:send-spot-instance-interruptions` 작업을 수행할 수 있도록 하는 정책을 첨부하세요. 자세한 설명은 [AWS FIS 실험을 위한 IAM 역할](#) 섹션을 참조하세요.

2. AWS FIS에 대한 액세스 권한 확인

AWS FIS 액세스 권한이 있어야 합니다. 자세한 내용을 알아보려면 [AWS FIS 정책 예제](#)를 참조하세요.

3. (선택 사항) 스팟 인스턴스 요청 생성

이 실험에 새 스팟 인스턴스를 사용하려면 [run-instances](#) 명령을 사용하여 스팟 인스턴스를 요청하세요. 기본값은 중단된 스팟 인스턴스를 종료하는 것입니다. 중단 동작을 `stop`로 설정하는 경우 유형도 `persistent`로 설정해야 합니다. 이 자습서에서는 최대 절전 모드 프로세스가 즉시 시작되므로 중단 동작을 `hibernate`로 설정하지 마세요.

```
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
```

다음은 `spot-options.json` 파일의 예입니다.

```
{
  "MarketType": "spot",
  "SpotOptions": {
    "SpotInstanceType": "persistent",
    "InstanceInterruptionBehavior": "stop"
  }
}
```

```
}
}
```

예제 명령의 `--query` 옵션을 사용하면 명령이 스팟 인스턴스의 인스턴스 ID만 반환하도록 할 수 있습니다. 출력의 예제는 다음과 같습니다.

```
[
  "i-0abcdef1234567890"
]
```

4. AWS FIS가 대상 스팟 인스턴스를 식별할 수 있도록 태그 추가

대상 스팟 인스턴스에 `Name=interruptMe` 태그를 추가하려면 [create-tags](#) 명령을 사용합니다.

```
aws ec2 create-tags \
  --resources i-0abcdef1234567890 \
  --tags Key=Name,Value=interruptMe
```

1단계: 실험 템플릿 만들기

AWS FIS 콘솔을 사용하여 실험 템플릿을 생성합니다. 템플릿에서 실행할 작업을 지정합니다. 작업은 지정된 태그가 있는 스팟 인스턴스를 중단합니다. 태그가 있는 스팟 인스턴스가 두 개 이상 있는 경우 AWS FIS는 그 중 하나를 임의로 선택합니다.

실험 템플릿 만들기

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿 생성을 선택합니다.
4. 이름 및 설명에 템플릿에 대한 이름과 설명을 입력합니다.
5. 작업에서 다음을 수행합니다.
 - a. 작업 추가를 선택합니다.
 - b. 작업의 이름을 입력합니다. 예를 들면 **interruptSpotInstance**를 입력합니다.
 - c. 작업 유형으로는 `aws:ec2:`를 선택합니다. `send-spot-instance-interruptions`
 - d. 대상의 경우 AWS FIS가 생성한 대상을 그대로 유지하세요.
 - e. 작업 파라미터의 경우 중단 전 기간을 2분(PT2M)으로 지정합니다.

- f. 저장을 선택합니다.
6. 대상에서 다음을 수행합니다.
 - a. AWS FIS가 이전 단계에서 자동으로 생성한 대상에 대해 편집을 선택합니다.
 - b. 기본 이름을 좀 더 이해하기 쉬운 이름으로 바꾸세요. 예를 들면 **oneSpotInstance**를 입력합니다.
 - c. 리소스 유형이 `aws:ec2:spot-instance`인지 확인하세요.
 - d. 대상 메서드의 경우 리소스 태그, 필터, 파라미터를 선택합니다.
 - e. 리소스 태그에 대해 새 태그 추가를 선택하고 태그 키와 태그 값을 입력합니다. 이 자습서의 사전 요구 사항에 설명된 대로 스팟 인스턴스에 추가한 태그를 사용하여 중단하세요.
 - f. 리소스 필터의 경우 새 필터 추가를 선택하고 경로로 **State.Name**, 값으로 **running**를 입력합니다.
 - g. 선택 모드에서는 개수를 선택합니다. 리소스 수에 **1**를 입력합니다.
 - h. 저장을 선택합니다.
7. 서비스 액세스의 경우 기존 IAM 역할 사용을 선택한 다음 이 자습서의 사전 조건에 설명된 대로 생성한 IAM 역할을 선택합니다. 역할이 표시되지 않는 경우 해당 역할에 필요한 신뢰 관계가 있는지 확인하세요. 자세한 설명은 [the section called “실험 역할”](#) 섹션을 참조하세요.
8. (선택 사항) 태그의 경우 새 태그 추가를 선택하고 태그 키와 태그 값을 지정합니다. 추가한 태그는 템플릿을 사용하여 실행되는 실험이 아니라 실험 템플릿에 적용됩니다.
9. 실험 템플릿 생성을 선택합니다. 확인 메시지가 나타나면 **create**을 입력하고 실험 템플릿 생성을 선택합니다.

(선택사항) 실험 템플릿 JSON을 보려면

내보내기 탭을 선택합니다. 다음은 이전 콘솔 절차에서 생성한 JSON 예제입니다.

```
{
  "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"
}
}
```

2단계: 실험 시작

실험 템플릿 생성을 완료하면 이를 사용하여 실험을 시작할 수 있습니다.

실험을 시작하려면

1. 방금 만든 실험 템플릿의 세부정보 페이지로 이동해야 합니다. 그렇지 않으면 실험 템플릿을 선택한 다음 실험 템플릿의 ID를 선택하여 세부 정보 페이지를 엽니다.
2. 실험 시작을 선택합니다.
3. (선택 사항) 실험에 태그를 추가하려면 새 태그 추가를 선택하고 태그 키와 태그 값을 입력합니다.
4. 실험 시작을 선택합니다. 확인 메시지가 나타나면 **start**을 입력하고 실험 시작을 선택합니다.

3단계: 실험 진행 상황 추적하기

실험이 완료, 중지 또는 실패할 때까지 진행 중인 실험의 진행 상황을 추적할 수 있습니다.

실험 진행 상황 추적하기

1. 방금 시작한 실험의 세부정보 페이지로 이동해야 합니다. 그렇지 않으면 실험을 선택한 다음 실험의 ID를 선택하여 세부 정보 페이지를 엽니다.
2. 실험 상태를 보려면 세부 정보 창에서 상태를 확인하세요. 자세한 내용은 [실험 상태](#)를 참조하세요.
3. 실험 상태가 실행 중이면 다음 단계로 이동합니다.

4단계: 실험 결과 확인

이 실험에 대한 작업이 완료되면 다음과 같이 진행됩니다.

- 대상 스팟 인스턴스가 [인스턴스 리밸런싱 권고](#)를 수신합니다.
- [스팟 인스턴스 중단 공지](#)는 Amazon EC2가 인스턴스를 종료 또는 중지하기 2분 전에 생성됩니다.
- 2분 후 스팟 인스턴스가 종료되거나 중지됩니다.
- AWS FIS에 의해 중지된 스팟 인스턴스는 다시 시작할 때까지 중지된 상태로 유지됩니다.

인스턴스가 실험에 의해 중단되었는지 확인

1. <https://console.aws.amazon.com/ec2/>에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 별도의 브라우저 탭 또는 창으로 Spot Requests(스팟 요청)와 Instances(인스턴스)를 엽니다.
3. Spot Requests(스팟 요청)에서 스팟 인스턴스 요청을 선택합니다. 초기 상태는 fulfilled입니다. 실험이 완료되면 상태가 다음과 같이 변경됩니다.
 - terminate - 상태가 instance-terminated-by-experiment로 변경됩니다.
 - stop - 상태가 marked-for-stop-by-experiment으로 변경되었다가 instance-stopped-by-experiment로 변경됩니다.
4. Instances(인스턴스)에서 스팟 인스턴스를 선택합니다. 초기 상태는 Running입니다. 스팟 인스턴스 중단 알림을 받고 2분 후 상태가 다음과 같이 변경됩니다.
 - stop - 상태가 Stopping으로 변경되었다가 Stopped로 변경됩니다.
 - terminate - 상태가 Shutting-down으로 변경되었다가 Terminated로 변경됩니다.

5단계: 정리

중단 동작이 stop인 이 실험에 대한 테스트 스팟 인스턴스를 생성했는데 더 이상 필요하지 않은 경우 스팟 인스턴스 요청을 취소하고 스팟 인스턴스를 종료할 수 있습니다.

요청을 취소하고 AWS CLI을 사용하여 인스턴스를 종료하려면

1. [cancel-spot-instance-requests](#) 명령을 사용하여 스팟 인스턴스 요청을 취소합니다.

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-ksie869j
```

2. 인스턴스를 종료하려면 [terminate-instances](#) 명령을 사용하세요.

```
aws ec2 terminate-instances --instance-ids i-0abcdef1234567890
```

실험 템플릿이 더 이상 필요하지 않으면 삭제할 수 있습니다.

AWS FIS 콘솔을 사용하여 실험 템플릿을 삭제하려면

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 삭제를 선택합니다.
4. 확인 메시지가 나타나면 **delete**를 입력한 다음 실험 템플릿 삭제를 선택합니다.

자습서: 연결 이벤트 시뮬레이션

AWS Fault Injection Service(AWS FIS)를 사용하여 다양한 연결 이벤트를 시뮬레이션할 수 있습니다. AWS FIS는 다음 방법 중 하나를 사용하여 네트워크 연결을 차단하여 연결 이벤트를 시뮬레이션합니다.

- **all** - 서브넷으로 들어오고 나가는 모든 트래픽을 거부합니다. 이 옵션은 서브넷의 네트워크 인터페이스로 들어오고 나가는 트래픽을 포함하여 서브넷 내 트래픽을 허용한다는 점에 유의하세요.
- **availability-zone** - 다른 가용 영역에 있는 서브넷으로 들어오고 나가는 VPC 내부 트래픽을 거부합니다.
- **dynamodb** - 현재 리전의 DynamoDB 리전 엔드포인트로 들어오고 나가는 트래픽을 거부합니다.
- **prefix-list** - 지정된 접두사 목록으로 들어오고 나가는 트래픽을 거부합니다.

- s3 - 현재 리전의 Amazon S3 리전 엔드포인트로 들어오고 나가는 트래픽을 거부합니다.
- vpc - VPC로 들어오고 나가는 트래픽을 거부합니다.

이 자습서를 사용하여 대상 서브넷에서 Amazon S3와의 연결 손실을 유발하는 AWS FIS `aws:network:disrupt-connectivity` 작업을 사용하는 실험 템플릿을 생성할 수 있습니다.

주제

- [사전 조건](#)
- [1단계: AWS FIS 실험 템플릿 만들기](#)
- [2단계: Amazon S3 엔드포인트에 대해 핑 전송](#)
- [3단계: AWS FIS 실험 시작](#)
- [4단계: AWS FIS 실험 진행 상황 추적하기](#)
- [5단계: Amazon S3 네트워크 중단 확인](#)
- [5단계: 정리](#)

사전 조건

이 자습서를 시작하기 전에 AWS 계정에서 적절한 권한을 가진 역할과 테스트 Amazon EC2 인스턴스가 필요합니다.

AWS 계정에서 권한을 가진 역할

역할을 생성하고 AWS FIS가 사용자 대신 `aws:network:disrupt-connectivity` 작업을 수행할 수 있도록 하는 정책을 첨부하세요.

IAM 역할에는 다음의 정책이 필요합니다.

- [AWSFaultInjectionSimulatorNetworkAccess](#)— Amazon EC2 네트워킹 및 기타 필수 서비스에서 네트워크 인프라와 관련된 AWS FIS AWS 작업을 수행하는 데 필요한 FIS 서비스 권한을 부여합니다.

Note

이 자습서에서는 단순화를 위해 AWS 관리형 정책을 사용합니다. 프로덕션 사용 용도의 경우 사용 사례에 필요한 최소한의 권한만 부여하는 것이 좋습니다.

IAM 역할을 생성하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 [AWS FIS 실험용 IAM 역할\(AWS CLI\)](#) 또는 [IAM 역할 생성\(콘솔\)](#)을 참조하세요.

테스트 Amazon EC2 인스턴스

테스트 Amazon EC2 인스턴스를 시작하고 연결합니다. 다음 자습서를 사용하여 Amazon EC2 인스턴스를 시작하고 연결할 수 있습니다. Amazon EC2 - Linux 인스턴스용 사용 설명서의 [자습서: Amazon EC2 Linux 인스턴스 시작](#).

1단계: AWS FIS 실험 템플릿 만들기

AWS FIS AWS Management Console를 사용하여 실험 템플릿을 생성합니다. AWS FIS 템플릿은 작업, 대상, 중지 조건 및 실험 역할로 구성됩니다. 템플릿 작동 방식에 대한 자세한 내용은 [AWS FIS용 실험 템플릿](#)을 참조하세요.

시작하기 전에 다음 항목이 준비되었는지 확인합니다.

- 올바른 권한을 가진 IAM 역할.
- Amazon EC2 인스턴스.
- Amazon EC2 인스턴스의 서브넷 ID.

실험 템플릿 만들기

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿 생성을 선택합니다.
4. 템플릿에 대한 설명(예: Amazon S3 Network Disrupt Connectivity)을 입력합니다.
5. 작업에서 작업 추가를 선택합니다.
 - a. 이름에 `disruptConnectivity`를 입력합니다.
 - b. 작업 유형에서 `aws:network:disrupt-connectivity`를 선택합니다.
 - c. 작업 파라미터에서 기간을 2 minutes로 설정합니다.
 - d. 범위에서 `s3`를 선택합니다.
 - e. 상단에서 저장을 선택합니다.
6. 대상 아래에서 자동으로 생성된 대상을 확인할 수 있습니다. 편집을 선택합니다.

- a. 리소스 유형이 `aws:ec2:subnet`인지 확인하세요.
 - b. 대상 메서드에서 리소스 ID를 선택한 다음, [사전 조건](#) 단계에서 Amazon EC2 인스턴스를 생성할 때 사용한 서브넷을 선택합니다.
 - c. 선택 모드가 모두인지 확인하세요.
 - d. 저장을 선택합니다.
7. 서비스 액세스에서 이 자습서의 [사전 조건](#)에 설명된 대로 생성한 IAM 역할을 선택합니다. 역할이 표시되지 않는 경우 해당 역할에 필요한 신뢰 관계가 있는지 확인하세요. 자세한 설명은 [the section called “실험 역할”](#) 섹션을 참조하세요.
 8. (선택 사항) 중지 조건에서는 상황이 발생할 경우 실험을 중단할 CloudWatch 경보를 선택할 수 있습니다. 자세한 내용은 [AWS FIS의 중지 조건](#)을 참조하세요.
 9. (선택 사항) 로그에서 Amazon S3 버킷을 선택하거나 CloudWatch 실험에 사용할 로그를 보낼 수 있습니다.
 10. 실험 템플릿 생성을 선택하고 확인 메시지가 표시되면 `create`를 입력합니다. 그런 다음 실험 템플릿 생성을 선택합니다.

2단계: Amazon S3 엔드포인트에 대해 핑 전송

Amazon EC2 인스턴스가 Amazon S3 엔드포인트에 도달할 수 있는지 확인하세요.

1. [사전 조건](#) 단계에 만든 Amazon EC2 인스턴스에 연결합니다.

문제 해결을 위해 Linux 인스턴스용 Amazon EC2 사용 설명서의 [인스턴스 연결 문제 해결](#)을 참조하세요.

2. 인스턴스가 있는 AWS 리전 위치를 확인하세요. 이렇게 하려면 Amazon EC2 콘솔에 있거나 다음 명령을 실행합니다.

```
hostname
```

예를 들어 `us-west-2`에서 Amazon EC2 인스턴스를 시작한 경우 다음과 같은 출력이 표시됩니다.

```
[ec2-user@ip-172.16.0.0 ~]$ hostname
ip-172.16.0.0.us-west-2.compute.internal
```

3. 사용자 AWS 리전에서 Amazon S3 엔드포인트에 대해 핑을 보내세요. **AWS ##**을 해당 리전으로 바꿉니다.

```
ping -c 1 s3.AWS ##.amazonaws.com
```

출력의 경우 다음 예제와 같이 핑에 성공하여 패킷 손실이 0% 인 것을 볼 수 있습니다.

```
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
```

3단계: AWS FIS 실험 시작

방금 만든 실험 템플릿으로 실험을 시작합니다.

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 실험 템플릿을 선택합니다.
3. 만든 실험 템플릿의 ID를 선택하여 세부 정보 페이지를 엽니다.
4. 실험 시작을 선택합니다.
5. (선택 사항) 확인 페이지에서 실험에 사용할 태그를 추가합니다.
6. 확인 페이지에서 실험 시작을 선택합니다.

4단계: AWS FIS 실험 진행 상황 추적하기

실험이 완료, 중지 또는 실패할 때까지 진행 중인 실험의 진행 상황을 추적할 수 있습니다.

1. 방금 시작한 실험의 세부정보 페이지로 이동해야 합니다. 그렇지 않은 경우 실험을 선택한 다음 실험의 ID를 선택하여 해당 세부정보 페이지를 여세요.
2. 실험 상태를 보려면 세부 정보 창에서 상태를 확인하세요. 자세한 내용은 [실험 상태](#)를 참조하세요.
3. 실험 상태가 실행 중이면 다음 단계로 이동합니다.

5단계: Amazon S3 네트워크 중단 확인

Amazon S3 엔드포인트에 핑을 보내 실험 진행 상황을 확인할 수 있습니다.

- Amazon EC2 인스턴스에서 사용자 AWS 리전의 Amazon S3 엔드포인트에 핑을 보냅니다. **AWS ##**을 해당 리전으로 바꿉니다.

```
ping -c 1 s3.AWS ##.amazonaws.com
```

출력의 경우 다음 예제와 같이 핑에 실패하여 패킷 손실이 100% 인 것을 볼 수 있습니다.

```
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
```

5단계: 정리

이 실험을 위해 만든 Amazon EC2 인스턴스나 AWS FIS 템플릿이 더 이상 필요하지 않은 경우 이 인스턴스를 제거할 수 있습니다.

Amazon EC2 인스턴스를 제거하려면

- <https://console.aws.amazon.com/ec2/>에서 Amazon EC2 콘솔을 엽니다.
- 탐색 창에서 인스턴스를 선택합니다.
- 테스트 인스턴스를 선택하고 인스턴스 상태, 인스턴스 종료를 차례로 선택합니다.
- 확인 메시지가 나타나면 종료를 선택합니다.

AWS FIS 콘솔을 사용하여 실험 템플릿을 삭제하려면

- <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
- 탐색 창에서 실험 템플릿을 선택합니다.
- 실험 템플릿을 선택한 다음 작업, 실험 템플릿 삭제를 선택합니다.
- 확인 메시지가 나타나면 delete를 입력한 다음 실험 템플릿 삭제를 선택합니다.

자습서: 반복 실험 예약

AWS Fault Injection Service(AWS FIS)를 통해 AWS 워크로드에 대한 오류 주입 실험을 수행할 수 있습니다. 이러한 실험은 지정된 대상에서 실행할 하나 이상의 작업이 포함된 템플릿에서 실행됩니다. Amazon EventBridge도 사용하는 경우 실험을 일회성 작업 또는 반복 작업으로 예약할 수 있습니다.

이 자습서를 사용하여 5분마다 AWS FIS 실험 템플릿을 실행하는 EventBridge 일정을 만들 수 있습니다.

Tasks

- [사전 조건](#)
- [1단계: IAM 역할 및 정책 생성](#)
- [2단계: Amazon EventBridge 스케줄러 생성](#)
- [3단계: 실험 확인](#)
- [4단계: 정리](#)

사전 조건

이 자습서를 시작하기 전에 일정에 따라 실행할 AWS FIS 실험 템플릿이 있어야 합니다. 이미 제대로 작동하는 실험 템플릿이 있다면 템플릿 ID 및 AWS 리전을 메모해 두세요. 그렇지 않으면 [the section called “테스트 인스턴스 중지 및 시작”](#)의 지침에 따라 템플릿을 만든 다음 이 자습서로 돌아갈 수 있습니다.

1단계: IAM 역할 및 정책 생성

IAM 역할 및 정책을 생성하려면

1. <https://console.aws.amazon.com/iam/>에서 IAM 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 사용자 지정 신뢰 정책을 선택한 후, 다음 스니펫을 삽입하여 Amazon EventBridge 스케줄러가 사용자를 대신하여 역할을 맡을 수 있도록 합니다.

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

```

    "Principal": {
      "Service": "scheduler.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
  }
]
}

```

다음을 선택합니다.

4. 권한 추가, 정책 생성을 선택합니다.
5. JSON을 선택하고 다음 정책을 삽입합니다. *your-experiment-template-id* 값을 사전 요구 사항 단계에 있는 실험의 템플릿 ID로 바꾸십시오.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "fis:StartExperiment",
      "Resource": [
        "arn:aws:fis:*:*:experiment-template/your-experiment-template-id",
        "arn:aws:fis:*:*:experiment/*"
      ]
    }
  ]
}

```

특정 태그 값이 있는 AWS FIS 실험만 실행하도록 스케줄러를 제한할 수 있습니다. 예를 들어 다음 정책은 모든 AWS FIS 실험 템플릿에 대한 StartExperiment 권한을 부여하지만 스케줄러는 Purpose=Schedule 태그가 지정된 실험만 실행하도록 제한합니다.

```

{
  "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"
      }
    }
  }
]
}

```

다음: 태그를 선택합니다.

6. 다음: 검토를 선택합니다.
7. 정책 검토에서 정책 이름을 FIS_RecurringExperiment로 지정한 다음 정책 생성을 선택합니다.
8. 권한 추가에서 새 FIS_RecurringExperiment 정책을 역할에 추가하고 다음을 선택합니다.
9. 이름 지정, 검토 및 생성 아래에서 역할 이름에 FIS_RecurringExperiment_role을 입력하고 역할 생성을 선택합니다.

2단계: Amazon EventBridge 스케줄러 생성

Amazon EventBridge 스케줄러 생성하기

1. <https://console.aws.amazon.com/events/> 에서 아마존 EventBridge 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 일정을 선택합니다.
3. AWS FIS 실험 템플릿과 동일한 AWS 리전에 있는지 확인하세요.
4. 일정 생성을 선택하고 다음 필드를 입력합니다.
 - 일정 이름 아래에 다음을 입력합니다. FIS_recurring_experiment_tutorial
 - 일정 패턴에서 반복 일정을 선택합니다.
 - 일정 유형에서 요금 기반 일정을 선택합니다.
 - rate 표현식에서 5분을 선택합니다.
 - 유연한 기간 아래에서 끄기를 선택합니다.
 - (선택 사항) 기간 아래에서 시간대를 선택합니다.
 - 다음을 선택합니다.

5. 대상 선택에서 모든 API를 선택한 다음 AWS FIS를 검색합니다.
6. AWSFIS를 선택한 다음 선택합니다 StartExperiment.
7. 입력에 다음 JSON 페이로드를 삽입합니다. *your-experiment-template-id*값을 실험의 템플릿 ID로 바꾸십시오. ClientToken는 스케줄러의 고유 식별자입니다. 이 자습서에서는 Amazon EventBridge 스케줄러에서 허용하는 컨텍스트 키워드를 사용합니다. 자세한 내용은 Amazon EventBridge 사용 설명서의 [컨텍스트 속성 추가](#)를 참조하십시오.

```
{
  "ClientToken": "<aws.scheduler.execution-id>",
  "ExperimentTemplateId": "your-experiment-template-id"
}
```

다음을 선택합니다.

8. (선택 사항) 설정에서 재시도 정책, DLQ(Dead Letter Queue) 및 암호화 설정을 지정할 수 있습니다. 또는 기본값을 그대로 유지합니다.
9. 권한 아래에서 기존 역할 사용을 선택한 다음 FIS_RecurringExperiment_role을 검색합니다.
10. 다음을 선택합니다.
11. 일정 검토 및 생성에서 스케줄러 세부 정보를 검토한 다음 일정 생성을 선택합니다.

3단계: 실험 확인

AWS FIS 실험이 일정대로 진행되었는지 확인하려면

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 실험을 선택합니다.
3. 일정을 만든 후 5분이 지나면 실험이 진행 중인 것을 확인할 수 있습니다.

4단계: 정리

Amazon EventBridge 스케줄러를 비활성화하려면

1. <https://console.aws.amazon.com/events/>에서 아마존 EventBridge 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 일정을 선택합니다.
3. 새로 만든 스케줄러를 선택한 다음 비활성화를 선택합니다.

에 대한 조치 AWS FIS

액션은 AWS Fault Injection Service (AWS FIS) 를 사용하여 대상에 대해 실행하는 오류 주입 활동입니다. AWS FIS AWS 서비스 전반의 특정 유형의 대상에 대해 사전 구성된 작업을 제공합니다. 실험 템플릿에 작업을 추가한 다음, 실험을 실행하는 데 사용합니다.

내용

- [작업 식별자](#)
- [작업 파라미터](#)
- [작업 대상](#)
- [AWS FIS 액션 레퍼런스](#)
- [FIS와 함께 AWS Systems Manager SSM 문서 사용](#)
- [AWS FIS aws:ecs:태스크 액션을 사용하세요.](#)
- [FIS aws:eks:pod 액션을 사용하십시오. AWS](#)
- [다음을 사용하여 AWS FIS 작업을 나열하십시오. AWS CLI](#)

작업 식별자

각 AWS FIS 작업에는 다음 형식의 식별자가 있습니다.

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

예를 들어 다음 작업은 대상 Amazon EC2 인스턴스를 중지합니다.

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

전체 작업 목록은 [AWS FIS 액션 레퍼런스](#) 섹션을 참조하세요. 를 사용하여 목록을 AWS CLI가져오려면 을 참조하십시오 [작업 나열](#).

작업 파라미터

일부 AWS FIS 액션에는 해당 액션에만 적용되는 추가 매개변수가 있습니다. 이러한 매개 변수는 작업이 AWS FIS 실행될 때 정보를 전달하는 데 사용됩니다.

AWS FIS SSM 에이전트와 SSM 명령 문서를 사용하여 대상 인스턴스에 장애 조건을 생성하는 `aws:ssm:send-command` 작업을 사용하여 사용자 지정 장애 유형을 지원합니다. `aws:ssm:send-command` 작업에는 SSM 문서의 Amazon 리소스 이름(ARN)을 값으로 취하는 `documentArn` 파라미터가 포함됩니다. 실험 템플릿에 작업을 추가할 때 파라미터 값을 지정합니다.

`aws:ssm:send-command` 작업 파라미터 지정에 대한 자세한 내용은 [aws:ssm:send-command 작업을 사용](#)을 참조하세요.

가능한 경우 작업 파라미터 내에 롤백 구성(사후 조치라고도 함)을 입력할 수 있습니다. 사후 작업은 대상을 작업이 실행되기 전의 상태로 되돌립니다. 사후 작업은 작업 기간에 지정된 시간 이후에 실행됩니다. 모든 작업이 사후 작업을 지원할 수 있는 것은 아닙니다. 예를 들어, 작업으로 인해 Amazon EC2 인스턴스가 종료되는 경우, 종료된 후에는 인스턴스를 복구할 수 없습니다.

작업 대상

작업은 지정한 대상 리소스에서 실행됩니다. 대상을 정의한 후 작업을 정의할 때 이름을 지정할 수 있습니다.

```
"targets": {
  "resource_type": "resource_name"
}
```

AWS FIS 액션은 액션 타겟에 대해 다음과 같은 리소스 유형을 지원합니다.

- Auto Scaling 그룹 – Amazon EC2 Auto Scaling 그룹
- 버킷 - Amazon S3 버킷
- 클러스터 - Amazon EKS 클러스터
- 클러스터 - Amazon ECS 클러스터 또는 Amazon Aurora DB 클러스터
- DB 인스턴스 – Amazon RDS DB 인스턴스
- 암호화된 글로벌 테이블 - 고객 관리형 키로 암호화된 글로벌 테이블인 Amazon DynamoDB
- 인스턴스 – Amazon EC2 인스턴스
- 노드 그룹 – Amazon EKS 노드 그룹
- 포드 – Amazon EKS의 Kubernetes 포드
- ReplicationGroups— ElastiCache Redis 복제 그룹
- 역할 - IAM 역할

- SpotInstances— 아마존 EC2 스팟 인스턴스
- 서브넷 - VPC 서브넷
- 작업 - Amazon ECS 작업
- TransitGateways— 트랜짓 게이트웨이
- 볼륨 - Amazon EBS 볼륨

예를 보려면 [the section called “작업 예”](#) 섹션을 참조하십시오.

AWS FIS 액션 레퍼런스

이 참조는 작업 파라미터 및 필수 IAM 권한에 대한 정보를 포함하여 의 일반적인 작업을 설명합니다. AWS FIS AWS FIS 콘솔 또는 () 의 [list-actions](#) 명령을 사용하여 지원되는 AWS FIS 작업을 나열할 수도 있습니다. AWS Command Line Interface AWS CLI

자세한 내용은 [예 대한 조치 AWS FIS](#) 및 [AWS 장애 인젝션 서비스가 IAM과 함께 작동하는 방식](#) 섹션을 참조하세요.

작업

- [오류 주입 작업](#)
- [대기 작업](#)
- [아마존 CloudWatch 액션](#)
- [Amazon DynamoDB 작업](#)
- [Amazon EBS 작업](#)
- [Amazon EC2 작업](#)
- [Amazon ECS 작업](#)
- [Amazon EKS 작업](#)
- [아마존 ElastiCache 액션](#)
- [네트워크 작업](#)
- [Amazon RDS 작업](#)
- [Amazon S3 작업](#)
- [Systems Manager 작업](#)

오류 주입 작업

AWS FIS 다음과 같은 오류 삽입 작업을 지원합니다.

작업

- [aws:fis:inject-api-internal-error](#)
- [aws:fis:inject-api-throttle-error](#)
- [aws:fis:inject-api-unavailable-error](#)

aws:fis:inject-api-internal-error

대상 IAM 역할이 만든 요청에 내부 오류를 삽입합니다.

리소스 유형

- aws:iam:role

파라미터

- duration - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.
- service— 대상 AWS API 네임스페이스. 지원되는 값은 ec2입니다.
- percentage - 오류를 주입하기 위한 호출의 백분율(1~100).
- operations - 오류를 주입하는 작업으로, 심포로 구분합니다. ec2 네임스페이스에 대한 API 작업 목록은 Amazon EC2 API 참조의 [작업](#)을 참조하세요.

권한

- fis:InjectApiInternalError

aws:fis:inject-api-throttle-error

대상 IAM 역할이 만든 요청에 제한 오류를 삽입합니다.

리소스 유형

- aws:iam:role

파라미터

- `duration` - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.
- `service`— 대상 AWS API 네임스페이스. 지원되는 값은 `ec2`입니다.
- `percentage` - 오류를 주입하기 위한 호출의 백분율(1~100).
- `operations` - 오류를 주입하는 작업으로, 심표로 구분합니다. `ec2` 네임스페이스에 대한 API 작업 목록은 Amazon EC2 API 참조의 [작업](#)을 참조하세요.

권한

- `fis:InjectApiThrottleError`

aws:fis:inject-api-unavailable-error

대상 IAM 역할이 만든 요청에 사용할 수 없음 오류를 삽입합니다.

리소스 유형

- `aws:iam:role`

파라미터

- `duration` - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.
- `service`— 대상 AWS API 네임스페이스. 지원되는 값은 `ec2`입니다.
- `percentage` - 오류를 주입하기 위한 호출의 백분율(1~100).
- `operations` - 오류를 주입하는 작업으로, 심표로 구분합니다. `ec2` 네임스페이스에 대한 API 작업 목록은 Amazon EC2 API 참조의 [작업](#)을 참조하세요.

권한

- `fis:InjectApiUnavailableError`

대기 작업

AWS FIS 다음 대기 작업을 지원합니다.

aws:fis:wait

AWS FIS 대기 동작을 실행합니다.

파라미터

- **duration** - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

- None

아마존 CloudWatch 액션

AWS FIS 다음 Amazon CloudWatch 작업을 지원합니다.

aws:cloudwatch:assert-alarm-state

지정된 경보가 지정된 경보 상태 중 하나에 해당하는지 확인합니다.

리소스 유형

- None

파라미터

- **alarmArns** - 경보의 ARN으로, 쉼표로 구분됩니다. 최대 5개의 경보를 지정할 수 있습니다.
- **alarmStates** - 경보 상태로, 쉼표로 구분됩니다. 가능한 경보 상태는 OK, ALARM 및 INSUFFICIENT_DATA입니다.

권한

- **cloudwatch:DescribeAlarms**

Amazon DynamoDB 작업

AWS FIS 는 다음과 같은 Amazon DynamoDB 작업을 지원합니다.

aws:dynamodb:encrypted-global-table-pause-replication

고객 관리 키 (CMK) 로 AWS Key Management Service 암호화된 테이블에 대한 Amazon DynamoDB 글로벌 테이블 복제를 일시 중지합니다. 이 작업은 대상 DynamoDB 글로벌 테이블의 데이터를 보호하는 데 사용되는 키에 액세스할 AWS KMS 수 있는 DynamoDB 복제 서비스 연결 역할에 대한 권한을 제거합니다. 작업이 시작된 후 최대 5분 동안 테이블이 계속 복제될 수 있습니다.

대상 DynamoDB 글로벌 테이블의 데이터를 보호하는 데 사용되는 AWS KMS 키의 정책에 다음 명령문이 동적으로 추가됩니다.

```
{
  "Sid": "DO_NOT_MODIFY_FIS_DDB_PAUSE_REPLICATION-EXP123456789012345",
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::123456789012:role/aws-service-role/
replication.dynamodb.amazonaws.com/AWSServiceRoleForDynamoDBReplication"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:aws:dynamodb:tableName": [
        "transactions-global-table",
        "inventory-global-table"
      ]
    }
  }
}
```

위의 정책 문은 DynamoDB 서비스 연결 역할이

kms:EncryptionContext:aws:dynamodb:tableName 컨텍스트 키에 나열된 테이블에 데이터

를 복제할 수 있는 권한을 제거합니다. 위 예제에서는 `transaction-global-table`, `inventory-global-table`과 같은 이름을 가진 DynamoDB 글로벌 테이블에 대한 복제가 중지됩니다.

리소스 유형

- `aws:dynamodb:encrypted-global-table`

파라미터

- `duration`— AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, `PT1M`은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

- `kms:DescribeKey`
- `kms:GetKeyPolicy`
- `kms:PutKeyPolicy`
- `dynamodb:DescribeTable`
- `dynamodb:DescribeGlobalTable`
- `tag:GetResources`

정책 예제는 [예: `aws:dynamodb:encrypted-global-table-pause-replication` 실행 권한이 있는 실험 역할](#)을 참조하세요.

Note

AWS FIS `kms:PutKeyPolicy`를 사용하여 DynamoDB, 즉 고객 AWS KMS 관리 키에 대한 액세스를 거부하여 복제를 중지합니다. 이 작업으로 실험을 적극적으로 실행하는 경우에만 역할을 사용하는 것이 좋으며, 그렇지 않은 경우에는 삭제하는 것이 좋습니다. 역할을 삭제하면 `kms:PutKeyPolicy`에 대한 FIS 권한이 제거됩니다. 실험이 완료된 후 실험 템플릿 세부 정보에서 역할을 찾습니다. IAM 콘솔에서 IAM 역할에 대한 링크를 선택하고 삭제를 선택합니다. 역할을 삭제한 후 AWS KMS 콘솔로 이동하여 대상 DynamoDB 테이블에서 데이터를 보호하는 데 사용되는 AWS KMS 키를 찾습니다. AWS KMS 키 정책이 예상과 일치하는지 확인하십시오. 더 이상 AWS FIS 진술 (예: `FIS_DDB_PAUSE_REPLICATION-EXP123456789012345_D0_NOT_MODIFY`) 이 표시되지 않을 것입니다.

Amazon EBS 작업

AWS FIS 는 다음과 같은 Amazon EBS 작업을 지원합니다.

aws:ebs:pause-volume-io

대상 EBS 볼륨의 I/O 작업을 일시 중지합니다. 대상 볼륨은 동일한 가용 영역에 있어야 하며 Nitro System 기반 인스턴스에 연결되어 있어야 합니다. Outpost의 인스턴스에는 볼륨을 연결할 수 없습니다.

Amazon EC2 콘솔을 사용하여 실험을 시작하려면 Amazon EC2 사용 설명서의 [Amazon EBS에서의 오류 테스트](#)를 참조하세요.

리소스 유형

- aws:ec2:ebs-volume

파라미터

- duration - 소요 시간은 1초에서 12시간입니다. AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어 PT1M은 1분, PT5S는 5초, PT6H는 6시간을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다. 지속 시간이 짧으면(예: PT5S) 지정된 기간 동안 I/O가 일시 중지되지만 실험을 초기화하는 데 걸리는 시간 때문에 실험이 완료되는 데 시간이 더 오래 걸릴 수 있습니다.

권한

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

Amazon EC2 작업

AWS FIS 는 다음과 같은 Amazon EC2 작업을 지원합니다.

작업

- [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 AWS Systems Manager SSM 에이전트를 통한 오류 주입 작업도 지원합니다. Systems Manager는 EC2 인스턴스에서 실행하는 작업을 정의하는 SSM 문서를 사용합니다. 자체 문서를 사용하여 사용자 지정 오류를 주입하거나 사전 구성된 SSM 문서를 사용할 수 있습니다. 자세한 정보는 [the section called "SSM 문서 사용"](#)을 참조하세요.

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

대상 IAM 역할의 요청에 대해 InsufficientInstanceCapacity 오류 응답을 삽입합니다. 지원되는 작업은 RunInstances,, CreateCapacityReservation StartInstances, CreateFleet 호출입니다. 여러 가용성 영역에서 용량 요청을 포함하는 요청은 지원되지 않습니다. 이 작업은 리소스 태그, 필터 또는 파라미터를 사용하여 대상을 정의하는 것을 지원하지 않습니다.

리소스 유형

- aws:iam:role

파라미터

- duration— AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.
- availabilityzoneidentifiers - 쉼표로 구분된 사용 가능 영역 목록. 영역 ID(예: "use1-az1, use1-az2") 및 영역 이름(예: "us-east-1a")을 지원합니다.
- percentage - 오류를 주입하기 위한 호출의 백분율(1~100).

권한

- 조건 키 ec2:FisActionId 값이 aws:ec2:api-insufficient-instance-capacity-error로 설정된 ec2:InjectApiError 조건 키와 타겟 IAM 역할로 설정된 ec2:FisTargetArns 조건 키가 있습니다.

정책 예제는 [예: ec2:InjectApiError의 조건 키 사용](#)을 참조하세요.

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

대상 Auto Scaling 그룹의 요청에 대해 `InsufficientInstanceCapacity` 오류 응답을 삽입합니다. 이 작업은 시작 템플릿을 사용하는 Auto Scaling 그룹만 지원합니다. 인스턴스 용량 부족 오류에 대해 자세히 알아보려면 [Amazon EC2 사용 설명서](#)를 참조하세요.

리소스 유형

- `aws:ec2:autoscaling-group`

파라미터

- `duration`— AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.
- `availabilityzoneidentifiers` - 쉼표로 구분된 사용 가능 영역 목록. 영역 ID(예: "use1-az1, use1-az2") 및 영역 이름(예: "us-east-1a")을 지원합니다.
- `percentage` - 선택 사항입니다. 대상 Auto Scaling 그룹의 시작 요청 중 오류를 주입할 비율(1~100)입니다. 기본값은 100입니다.

권한

- `ec2:InjectApiError` 조건 키 `ec2:FisActionId` 값이 `aws:ec2:asg-insufficient-instance-capacity-error` 설정되고 `ec2:FisTargetArns` 조건 키가 Auto Scaling 그룹을 대상으로 설정된 경우
- `autoscaling:DescribeAutoScalingGroups`

정책 예제는 [예: `ec2:InjectApiError`의 조건 키 사용](#)을 참조하세요.

aws:ec2:reboot-instances

대상 EC2 [RebootInstances](#) 인스턴스에서 Amazon EC2 API 작업을 실행합니다.

리소스 유형

- `aws:ec2:instance`

파라미터

- None

권한

- ec2:RebootInstances
- ec2:DescribeInstances

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEC2Access](#)

aws:ec2:send-spot-instance-interruptions

대상 스팟 인스턴스를 중단합니다. 대상 스팟 인스턴스를 중단하기 2분 전에 [스팟 인스턴스 중단 공지](#)를 대상 스팟 인스턴스에 보냅니다. 중단 시간은 지정된 `durationBeforeInterruption` 매개변수에 따라 결정됩니다. 중단 시점으로부터 2분 후에 스팟 인스턴스는 중단 동작에 따라 종료되거나 중지됩니다. AWS FIS 에 의해 중지된 스팟 인스턴스는 다시 시작할 때까지 중지된 상태로 유지됩니다.

작업이 시작된 직후 대상 인스턴스는 [EC2 인스턴스 재조정 권장 사항](#)을 수신합니다. 지정한 `durationBeforeInterruption` 경우 재조정 권장 사항과 중단 알림 사이에 지연이 있을 수 있습니다.

자세한 정보는 [the section called “스팟 인스턴스 중단 테스트”](#)을 참조하세요. 또는 Amazon EC2 콘솔을 사용하여 실험을 시작하려면 Amazon EC2 사용 설명서의 [스팟 인스턴스 중단 시작](#)을 참조하세요.

리소스 유형

- aws:ec2:spot-instance

파라미터

- `durationBeforeInterruption` - 인스턴스 중단까지 걸리는 대기 시간(2~15분). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT2M은 2분을 나타냅니다. AWS FIS 콘솔에서 시간(분)을 입력합니다.

권한

- `ec2:SendSpotInstanceInterruptions`
- `ec2:DescribeInstances`

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEC2Access](#)

aws:ec2:stop-instances

대상 EC2 [StopInstances](#) 인스턴스에서 Amazon EC2 API 작업을 실행합니다.

리소스 유형

- `aws:ec2:instance`

파라미터

- `startInstancesAfterDuration` – 선택 사항입니다. 인스턴스를 시작하기 전에 대기하는 데 걸리는 시간은 1분에서 12시간입니다. AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간 단위 숫자를 입력합니다. 인스턴스에 암호화된 EBS 볼륨이 있는 경우 볼륨을 암호화하는 데 사용된 KMS 키에 AWS FIS 권한을 부여하거나 KMS 키 정책에 실험 역할을 추가해야 합니다.
- `completeIfInstancesTerminated` – 선택 사항입니다. true인 경우 그리고 `startInstancesAfterDuration`도 true인 경우, 대상 EC2 인스턴스가 FIS 외부의 별도 요청에 의해 종료되어 다시 시작할 수 없는 경우에도 이 작업은 실패하지 않습니다. 예를 들어, Auto Scaling 그룹은 이 작업이 완료되기 전에 해당 그룹이 제어하는 중지된 EC2 인스턴스를 종료할 수 있습니다. 기본값은 false입니다.

권한

- `ec2:StopInstances`
- `ec2:StartInstances`
- `ec2:DescribeInstances` – 선택 사항입니다. 작업 `completeIfInstances` 종료 시 인스턴스 상태를 확인하려면 `Terminated`와 함께 필요합니다.

- kms:CreateGrant – 선택 사항입니다. startInstancesAfterDuration과 함께 암호화된 볼륨으로 인스턴스를 다시 시작하는 데 필요합니다.

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEC2Access](#)

aws:ec2:terminate-instances

대상 EC2 [TerminateInstances](#) 인스턴스에서 Amazon EC2 API 작업을 실행합니다.

리소스 유형

- aws:ec2:instance

파라미터

- None

권한

- ec2:TerminateInstances
- ec2:DescribeInstances

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEC2Access](#)

Amazon ECS 작업

AWS FIS 는 다음과 같은 Amazon ECS 작업을 지원합니다.

작업

- [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

Amazon ECS API 작업을 [UpdateContainerInstancesState](#) 실행하여 대상 클러스터에 있는 기본 Amazon EC2 인스턴스의 지정된 비율을 비웁니다.

리소스 유형

- aws:ecs:cluster

파라미터

- drainagePercentage - 백분율(1~100).
- duration - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorECSAccess](#)

aws:ecs:stop-task

Amazon ECS API 작업을 [StopTask](#) 실행하여 대상 작업을 중지합니다.

리소스 유형

- `aws:ecs:task`

파라미터

- None

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorECSAccess](#)

`aws:ecs:task-cpu-stress`

대상 작업에 CPU 스트레스를 실행합니다. [AWSFIS-RUN-CPU-스트레스](#) SSM 문서를 사용합니다. 작업은 에서 관리해야 합니다. AWS Systems Manager 자세한 정보는 [ECS 작업 동작 사용](#)을 참조하세요.

리소스 유형

- `aws:ecs:task`

파라미터

- `duration` - ISO 8601 형식의 스트레스 테스트 기간입니다.
- `percent` - 선택 사항입니다. 0(무부하)에서 100(완전 부하)까지의 목표 부하 백분율. 기본값은 100입니다.
- `workers` - 선택 사항입니다. 사용할 스트레스 요인의 수입니다. 기본값은 0이며, 모든 스트레스 요인을 사용합니다.

- `installDependencies` – 선택 사항입니다. 이 값이 `True`이면 Systems Manager가 SSM 에이전트의 사이드카 컨테이너에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 `True`입니다. 종속성은 `stress-ng`입니다.

권한

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

aws:ecs:task-io-stress

대상 작업에 I/O 스트레스를 실행합니다. [AWSFIS-run-IO-stress SSM](#) 문서를 사용합니다. 작업은 에서 관리해야 합니다. AWS Systems Manager 자세한 정보는 [ECS 작업 동작 사용](#)을 참조하세요.

리소스 유형

- `aws:ecs:task`

파라미터

- `duration` - ISO 8601 형식의 스트레스 테스트 기간입니다.
- `percent` – 선택 사항입니다. 스트레스 테스트 중에 사용할 파일 시스템의 여유 공간 비율입니다. 기본값은 80%입니다.
- `workers` – 선택 사항입니다. 작업자 수. 작업자는 순차, 임의 및 메모리 매핑된 읽기/쓰기 작업, 강제 동기화 및 캐시 삭제를 혼합하여 수행합니다. 여러 하위 프로세스가 동일한 파일에서 다양한 I/O 작업을 수행합니다. 기본값은 1입니다.
- `installDependencies` – 선택 사항입니다. 이 값이 `True`이면 Systems Manager가 SSM 에이전트의 사이드카 컨테이너에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 `True`입니다. 종속성은 `stress-ng`입니다.

권한

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

aws:ecs:task-kill-process

killall 명령을 사용하여 작업에서 지정된 프로세스를 중지합니다. [AWSFIS-Run-Kill-Process SSM 문서를](#) 사용하십시오. 작업 정의의 pidMode가 task로 설정되어야 합니다. 작업은 에서 관리해야 합니다. AWS Systems Manager자세한 정보는 [ECS 작업 동작 사용](#)을 참조하세요.

리소스 유형

- aws:ecs:task

파라미터

- processName - 중지할 프로세스의 이름.
- signal - 선택 사항입니다. 명령과 함께 전송할 신호입니다. 가능한 값은 SIGTERM(수신자가 무시하도록 선택할 수 있는 값)과 SIGKILL(무시할 수 없는 값)입니다. 기본값은 SIGTERM입니다.
- installDependencies - 선택 사항입니다. 이 값이 True이면 Systems Manager가 SSM 에이전트의 사이드카 컨테이너에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 killall입니다.

권한

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

aws:ecs:task-network-blackhole-port

지정된 프로토콜 및 포트에 대한 인바운드 또는 아웃바운드 트래픽을 삭제합니다. [AWSFIS-Run-Network-Blackhole-Port SSM 문서를](#) 사용하십시오. 작업 정의의 pidMode가 task로 설정되어야 합니다. 작업은 에서 관리해야 합니다. AWS Systems Manager작업 정의에서 networkMode를 bridge로 설정할 수 없습니다. 자세한 정보는 [ECS 작업 동작 사용](#)을 참조하세요.

리소스 유형

- aws:ecs:task

파라미터

- duration - ISO 8601 형식의 테스트 기간입니다.
- port - 포트 번호입니다.
- trafficType - 트래픽 유형입니다. 가능한 값은 ingress와 egress입니다.
- protocol - 선택 사항입니다. 프로토콜. 가능한 값은 tcp와 udp입니다. 기본값은 tcp입니다.
- installDependencies - 선택 사항입니다. 이 값이 True이면 Systems Manager가 SSM 에이전트의 사이드카 컨테이너에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 atd, dig 및 iptables입니다.

권한

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

aws:ecs:task-network-latency

특정 소스에서 오가는 트래픽을 처리하는 tc 도구를 사용하여 네트워크 인터페이스에 지연 시간과 지터를 추가합니다. [AWSFIS-Run-Network-Latency-Sources SSM 문서를](#) 사용하십시오. 작업 정의의 pidMode가 task로 설정되어야 합니다. 작업은 에서 관리해야 합니다. AWS Systems Manager 작업 정의에서 networkMode를 bridge로 설정할 수 없습니다. 자세한 정보는 [ECS 작업 동작 사용](#)을 참조하십시오.

리소스 유형

- aws:ecs:task

파라미터

- duration - ISO 8601 형식의 테스트 기간입니다.
- interface - 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 eth0입니다.
- delayMilliseconds - 선택 사항입니다. 밀리 초 단위의 지연 시간입니다. 기본값은 200입니다.
- jitterMilliseconds - 선택 사항입니다. 밀리 초 단위의 지터입니다. 기본값은 10.

- `sources` – 선택 사항입니다. 쉘표로 구분된 소스입니다. 가능한 값은 IPv4 주소, IPv4 CIDR 블록, 도메인 이름, DYNAMODB 및 S3입니다. DYNAMODB 또는 S3를 지정하는 경우 이는 현재 리전의 리전 엔드포인트에만 적용됩니다. 기본값은 모든 IPv4 트래픽과 일치하는 0.0.0.0/0입니다.
- `installDependencies` – 선택 사항입니다. 이 값이 True이면 Systems Manager가 SSM 에이전트의 사이드카 컨테이너에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 `atd`, `dig`, `jq` 및 `tc`입니다.

권한

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

aws:ecs:task-network-packet-loss

`tc` 도구를 사용하여 네트워크 인터페이스에 패킷 손실을 추가합니다. [AWSFIS-Run-Network-Packet-Loss-Sources SSM 문서](#)를 사용합니다. 작업 정의의 `pidMode`가 `task`로 설정되어야 합니다. 작업은 에서 관리해야 합니다. AWS Systems Manager 작업 정의에서 `networkMode`를 `bridge`로 설정할 수 없습니다. 자세한 정보는 [ECS 작업 동작 사용](#)을 참조하세요.

리소스 유형

- `aws:ecs:task`

파라미터

- `duration` - ISO 8601 형식의 테스트 기간입니다.
- `interface` – 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 `eth0`입니다.
- `lossPercent` – 선택 사항입니다. 패킷 손실의 백분율. 기본값은 7%입니다.
- `sources` – 선택 사항입니다. 쉘표로 구분된 소스입니다. 가능한 값은 IPv4 주소, IPv4 CIDR 블록, 도메인 이름, DYNAMODB 및 S3입니다. DYNAMODB 또는 S3를 지정하는 경우 이는 현재 리전의 리전 엔드포인트에만 적용됩니다. 기본값은 모든 IPv4 트래픽과 일치하는 0.0.0.0/0입니다.
- `installDependencies` – 선택 사항입니다. 이 값이 True이면 Systems Manager가 SSM 에이전트의 사이드카 컨테이너에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 `atd`, `dig`, `jq` 및 `tc`입니다.

권한

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

Amazon EKS 작업

AWS FIS 는 다음과 같은 Amazon EKS 작업을 지원합니다.

작업

- [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

단일 대상 클러스터에서 ChaosMesh 또는 Litmus 실험을 실행합니다. 대상 클러스터에 ChaosMesh 또는 Litmus를 설치해야 합니다.

실험 템플릿을 생성하고 `aws:eks:cluster` 대상 유형을 정의할 때는 이 작업을 단일 Amazon 리소스 이름(ARN)으로 타겟팅해야 합니다. 이 작업은 리소스 태그, 필터 또는 파라미터를 사용하여 대상을 정의하는 것을 지원하지 않습니다.

설치할 ChaosMesh 때 적절한 컨테이너 런타임을 지정해야 합니다. Amazon EKS 버전 1.23부터 기본 런타임이 Docker에서 containerd로 변경되었습니다. 버전 1.24부터 Docker가 제거되었습니다.

리소스 유형

- `aws:eks:cluster`

파라미터

- `kubernetesApiVersion` – [Kubernetes 사용자 지정 리소스](#)의 API 버전. 가능한 값: `chaos-mesh.org/v1alpha1` | `litmuschaos.io/v1alpha1`.
- `kubernetesKind` - Kubernetes 사용자 지정 리소스 종류. 값은 API 버전에 따라 달라집니다.
 - `chaos-mesh.org/v1alpha1` – 가능한 값: `AWSChaos` | `DNSChaos` | `GCPChaos` | `HTTPChaos` | `IIOChaos` | `JVMChaos` | `KernelChaos` | `NetworkChaos` | `PhysicalMachineChaos` | `PodChaos` | `PodHttpChaos` | `PodIOChaos` | `PodNetworkChaos` | `Schedule` | `StressChaos` | `TimeChaos` |
 - `litmuschaos.io/v1alpha1` – 가능한 값: `ChaosEngine`.
- `kubernetesNamespace` – [Kubernetes 네임스페이스](#).
- `kubernetesSpec` – JSON 형식으로 된 Kubernetes 사용자 지정 리소스의 `spec` 섹션.
- `maxDuration` - 자동화 실행이 완료되는 데 허용되는 최대 시간은 1분에서 12시간입니다. AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, `PT1M`은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

이 AWS 작업에는 ID 및 Access Management (IAM) 권한이 필요하지 않습니다. 이 작업을 사용하는 데 필요한 권한은 RBAC 인증을 사용하여 Kubernetes에서 제어합니다. 자세한 내용은 공식 Kubernetes 문서의 [RBAC 승인 사용](#)을 참조하세요. Chaos Mesh에 대한 자세한 내용은 [공식 Chaos Mesh 문서](#)를 참조하세요. Litmus에 대한 자세한 내용은 [공식 Litmus 문서](#)를 참조하세요.

aws:eks:pod-cpu-stress

대상 포드에 CPU 스트레스를 실행합니다. 자세한 정보는 [EKS 포드 작업 사용](#)을 참조하세요.

리소스 유형

- `aws:eks:pod`

파라미터

- `duration` - ISO 8601 형식의 스트레스 테스트 기간입니다.
- `percent` – 선택 사항입니다. 0(무부하)에서 100(완전 부하)까지의 목표 부하 백분율. 기본값은 100입니다.

- `workers` – 선택 사항입니다. 사용할 스트레스 요인의 수입니다. 기본값은 0이며, 모든 스트레스 요인을 사용합니다.
- `kubernetesServiceAccount` - Kubernetes 서비스 계정. 필요한 권한에 대한 자세한 내용은 [the section called “Kubernetes 서비스 계정 구성”](#) 단원을 참조하세요.
- `fisPodContainerImage` – 선택 사항입니다. 오류 인젝터 포드를 만드는 데 사용된 컨테이너 이미지. 기본값은 에서 제공한 AWS FIS 이미지를 사용하는 것입니다. 자세한 정보는 [the section called “포드 컨테이너 이미지”](#)을 참조하세요.
- `maxErrorsPercent` – 선택 사항입니다. 오류 주입이 실패하기 전에 실패할 수 있는 대상의 비율입니다. 기본값은 0입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

`aws:eks:pod-delete`

대상 포드를 삭제합니다. 자세한 정보는 [EKS 포드 작업 사용](#)을 참조하세요.

리소스 유형

- `aws:eks:pod`

파라미터

- `gracePeriodSeconds` – 선택 사항입니다. 포드가 정상적으로 종료될 때까지 기다리는 시간(초)입니다. 값이 0이면 작업을 즉시 수행합니다. 값이 nil인 경우 포드의 기본 유예 기간을 사용합니다.
- `kubernetesServiceAccount` - Kubernetes 서비스 계정. 필요한 권한에 대한 자세한 내용은 [the section called “Kubernetes 서비스 계정 구성”](#) 단원을 참조하세요.

- `fisPodContainerImage` – 선택 사항입니다. 오류 인젝터 포드를 만드는 데 사용된 컨테이너 이미지. 기본값은 에서 제공한 이미지를 사용하는 AWS FIS것입니다. 자세한 정보는 [the section called “포드 컨테이너 이미지”](#)을 참조하세요.
- `maxErrorsPercent` – 선택 사항입니다. 오류 주입이 실패하기 전에 실패할 수 있는 대상의 비율입니다. 기본값은 0입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

aws:eks:pod-io-stress

대상 포드에서 I/O 스트레스를 실행합니다. 자세한 정보는 [EKS 포드 작업 사용](#)을 참조하세요.

리소스 유형

- `aws:eks:pod`

파라미터

- `duration` - ISO 8601 형식의 스트레스 테스트 기간입니다.
- `workers` – 선택 사항입니다. 작업자 수. 작업자는 순차, 임의 및 메모리 매핑된 읽기/쓰기 작업, 강제 동기화 및 캐시 삭제를 혼합하여 수행합니다. 여러 하위 프로세스가 동일한 파일에서 다양한 I/O 작업을 수행합니다. 기본 값은 1입니다.
- `percent` – 선택 사항입니다. 스트레스 테스트 중에 사용할 파일 시스템의 여유 공간 비율입니다. 기본 값은 80%입니다.
- `kubernetesServiceAccount` - Kubernetes 서비스 계정. 필요한 권한에 대한 자세한 내용은 [the section called “Kubernetes 서비스 계정 구성”](#) 단원을 참조하세요.

- `fisPodContainerImage` – 선택 사항입니다. 오류 인젝터 포드를 만드는 데 사용된 컨테이너 이미지. 기본값은 에서 제공한 이미지를 사용하는 AWS FIS것입니다. 자세한 정보는 [the section called “포드 컨테이너 이미지”](#)을 참조하세요.
- `maxErrorsPercent` – 선택 사항입니다. 오류 주입이 실패하기 전에 실패할 수 있는 대상의 비율입니다. 기본값은 0입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

aws:eks:pod-memory-stress

대상 포드에 메모리 스트레스를 실행합니다. 자세한 정보는 [EKS 포드 작업 사용](#)을 참조하세요.

리소스 유형

- `aws:eks:pod`

파라미터

- `duration` - ISO 8601 형식의 스트레스 테스트 기간입니다.
- `workers` – 선택 사항입니다. 사용할 스트레스 요인의 수입니다. 기본 값은 1입니다.
- `percent` – 선택 사항입니다. 스트레스 테스트 중에 사용할 가상 메모리의 비율입니다. 기본값은 80%입니다.
- `kubernetesServiceAccount` - Kubernetes 서비스 계정. 필요한 권한에 대한 자세한 내용은 [the section called “Kubernetes 서비스 계정 구성”](#) 단원을 참조하세요.
- `fisPodContainerImage` – 선택 사항입니다. 오류 인젝터 포드를 만드는 데 사용된 컨테이너 이미지. 기본값은 에서 제공한 이미지를 사용하는 AWS FIS것입니다. 자세한 정보는 [the section called “포드 컨테이너 이미지”](#)을 참조하세요.

- `maxErrorsPercent` – 선택 사항입니다. 오류 주입이 실패하기 전에 실패할 수 있는 대상의 비율입니다. 기본값은 0입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

`aws:eks:pod-network-blackhole-port`

지정된 프로토콜 및 포트에 대한 인바운드 또는 아웃바운드 트래픽을 삭제합니다. 자세한 정보는 [EKS 포트 작업 사용](#)을 참조하세요.

리소스 유형

- `aws:eks:pod`

파라미터

- `duration` - ISO 8601 형식의 테스트 기간입니다.
- `protocol` – 선택 사항입니다. 프로토콜. 가능한 값은 `tcp`와 `udp`입니다. 기본값은 `tcp`입니다.
- `trafficType` - 트래픽 유형입니다. 가능한 값은 `ingress`와 `egress`입니다.
- `port` - 포트 번호입니다.
- `kubernetesServiceAccount` - Kubernetes 서비스 계정. 필요한 권한에 대한 자세한 내용은 [the section called “Kubernetes 서비스 계정 구성”](#) 단원을 참조하세요.
- `fisPodContainerImage` – 선택 사항입니다. 오류 인젝터 포드를 만드는 데 사용된 컨테이너 이미지. 기본값은 에서 제공한 이미지를 사용하는 AWS FIS것입니다. 자세한 정보는 [the section called “포트 컨테이너 이미지”](#)을 참조하세요.
- `maxErrorsPercent` – 선택 사항입니다. 오류 주입이 실패하기 전에 실패할 수 있는 대상의 비율입니다. 기본값은 0입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

aws:eks:pod-network-latency

특정 소스에서 오가는 트래픽을 처리하는 tc 도구를 사용하여 네트워크 인터페이스에 지연 시간과 지터를 추가합니다. 자세한 정보는 [EKS 포드 작업 사용](#)을 참조하세요.

리소스 유형

- `aws:eks:pod`

파라미터

- `duration` - ISO 8601 형식의 테스트 기간입니다.
- `interface` - 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 `eth0`입니다.
- `delayMilliseconds` - 선택 사항입니다. 밀리 초 단위의 지연 시간입니다. 기본값은 200입니다.
- `jitterMilliseconds` - 선택 사항입니다. 밀리 초 단위의 지터입니다. 기본값은 10.
- `sources` - 선택 사항입니다. 쉘표로 구분된 소스입니다. 가능한 값은 IPv4 주소, IPv4 CIDR 블록, 도메인 이름, DYNAMODB 및 S3입니다. DYNAMODB 또는 S3를 지정하는 경우 이는 현재 리전의 리전 엔드포인트에만 적용됩니다. 기본값은 모든 IPv4 트래픽과 일치하는 `0.0.0.0/0`입니다.
- `kubernetesServiceAccount` - Kubernetes 서비스 계정. 필요한 권한에 대한 자세한 내용은 [the section called “Kubernetes 서비스 계정 구성”](#) 단원을 참조하세요.
- `fisPodContainerImage` - 선택 사항입니다. 오류 인젝터 포드를 만드는 데 사용된 컨테이너 이미지. 기본값은 에서 제공한 이미지를 사용하는 AWS FIS것입니다. 자세한 정보는 [the section called “포드 컨테이너 이미지”](#)을 참조하세요.
- `maxErrorsPercent` - 선택 사항입니다. 오류 주입이 실패하기 전에 실패할 수 있는 대상의 비율입니다. 기본값은 0입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

`aws:eks:pod-network-packet-loss`

tc 도구를 사용하여 네트워크 인터페이스에 패킷 손실을 추가합니다. 자세한 정보는 [EKS 포트 작업 사용](#)을 참조하세요.

리소스 유형

- `aws:eks:pod`

파라미터

- `duration` - ISO 8601 형식의 테스트 기간입니다.
- `interface` - 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 `eth0`입니다.
- `lossPercent` - 선택 사항입니다. 패킷 손실의 백분율. 기본값은 7%입니다.
- `sources` - 선택 사항입니다. 쉘표로 구분된 소스입니다. 가능한 값은 IPv4 주소, IPv4 CIDR 블록, 도메인 이름, DYNAMODB 및 S3입니다. DYNAMODB 또는 S3를 지정하는 경우 이는 현재 리전의 리전 엔드포인트에만 적용됩니다. 기본값은 모든 IPv4 트래픽과 일치하는 `0.0.0.0/0`입니다.
- `kubernetesServiceAccount` - Kubernetes 서비스 계정. 필요한 권한에 대한 자세한 내용은 [the section called "Kubernetes 서비스 계정 구성"](#) 단원을 참조하세요.
- `fisPodContainerImage` - 선택 사항입니다. 오류 인젝터 포드를 만드는 데 사용된 컨테이너 이미지. 기본값은 에서 제공한 이미지를 사용하는 AWS FIS것입니다. 자세한 정보는 [the section called "포트 컨테이너 이미지"](#)을 참조하세요.
- `maxErrorsPercent` - 선택 사항입니다. 오류 주입이 실패하기 전에 실패할 수 있는 대상의 비율입니다. 기본값은 0입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

`aws:eks:terminate-nodegroup-instances`

대상 노드 그룹에서 Amazon EC2 API [TerminateInstances](#) 작업을 실행합니다.

리소스 유형

- `aws:eks:nodegroup`

파라미터

- `instanceTerminationPercentage` - 종료할 인스턴스의 백분율(1~100).

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEKSAccess](#)

아마존 ElastiCache 액션

AWS FIS 다음 ElastiCache 작업을 지원합니다.

aws:elasticache:interrupt-cluster-az-power

대상 Redis 복제 그룹에 대해 지정된 가용 영역에 있는 노드의 전원을 중단합니다. 기본 노드가 타깃이 되면 복제 지연이 가장 적은 해당 읽기 복제본이 프라이머리 노드로 승격됩니다. 지정된 가용성 영역의 읽기 복제본 교체는 이 작업이 진행되는 동안 차단되므로 대상 복제 그룹이 줄어든 용량으로 작동합니다.

리소스 유형

- `aws:elasticache:redis-replicationgroup`

파라미터

- `duration` - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

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

네트워크 작업

AWS FIS 다음 네트워크 작업을 지원합니다.

작업

- [aws:network:disrupt-connectivity](#)
- [aws:network:route-table-disrupt-cross-region-connectivity](#)
- [aws:network:transit-gateway-disrupt-cross-region-connectivity](#)

aws:network:disrupt-connectivity

대상 서브넷으로의 지정된 트래픽을 거부합니다.

리소스 유형

- `aws:ec2:subnet`

파라미터

- `scope` - 거부할 트래픽의 유형입니다. 가능한 값은 다음과 같습니다.
 - `all` - 서브넷으로 들어오고 나가는 모든 트래픽을 거부합니다. 이 옵션은 서브넷의 네트워크 인터페이스로 들어오고 나가는 트래픽을 포함하여 서브넷 내 트래픽을 허용한다는 점에 유의하세요.
 - `availability-zone` - 다른 가용 영역에 있는 서브넷으로 들어오고 나가는 VPC 내부 트래픽을 거부합니다.
 - `dynamodb` - 현재 리전의 DynamoDB 리전 엔드포인트로 들어오고 나가는 트래픽을 거부합니다.
 - `prefix-list` - 지정된 접두사 목록으로 들어오고 나가는 트래픽을 거부합니다.
 - `s3` - 현재 리전의 Amazon S3 리전 엔드포인트로 들어오고 나가는 트래픽을 거부합니다.
 - `vpc` - VPC로 들어오고 나가는 트래픽을 거부합니다.
- `duration` - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.
- `prefixListIdentifier` - 범위가 `prefix-list`인 경우 고객이 관리하는 접두사 목록의 식별자입니다. 이름, ID 또는 ARN을 지정할 수 있습니다. 접두사 목록에는 최대 10개의 항목이 있을 수 있습니다.

권한

- `ec2:CreateNetworkAcl` - `managedByFIS=true` 태그를 사용하여 네트워크 ACL을 생성합니다.
- `ec2:CreateNetworkAclEntry` - 네트워크 ACL에 `managedByFIS=true` 태그가 있어야 합니다.
- `ec2:CreateTags`
- `ec2>DeleteNetworkAcl` - 네트워크 ACL에 `managedByFIS=true` 태그가 있어야 합니다.
- `ec2:DescribeManagedPrefixLists`
- `ec2:DescribeNetworkAcls`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:GetManagedPrefixListEntries`
- `ec2:ReplaceNetworkAclAssociation`

AWS 관리형 정책

- [AWSFaultInjectionSimulatorNetworkAccess](#)

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

대상 서브넷에서 시작하여 지정된 리전으로 향하는 트래픽을 차단합니다.

리소스 유형

- aws:ec2:subnet

파라미터

- region - 격리할 리전의 코드(예: eu-west-1)입니다.
- duration - 작업이 지속되는 시간입니다. AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

- 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

† managedByFIS=true 태그를 사용하여 범위를 지정합니다.

AWS 관리형 정책

- [AWSFaultInjectionSimulatorNetworkAccess](#)

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

지정된 리전으로 향하는 대상 전송 게이트웨이 피어링 첨부 파일의 트래픽을 차단합니다.

리소스 유형

- aws:ec2:transit-gateway

파라미터

- region - 격리할 리전의 코드(예: eu-west-1)입니다.
- duration - 작업이 지속되는 시간입니다. AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorNetworkAccess](#)

Amazon RDS 작업

AWS FIS 는 다음과 같은 Amazon RDS 작업을 지원합니다.

작업

- [aws:rds:failover-db-cluster](#)
- [aws:rds:reboot-db-instances](#)

aws:rds:failover-db-cluster

대상 Aurora DB 클러스터에서 Amazon RDS API 작업 [FailoverDBCluster](#)를 실행합니다.

리소스 유형

- aws:rds:cluster

파라미터

- None

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorRDSAccess](#)

aws:rds:reboot-db-instances

대상 DB 인스턴스에서 Amazon RDS API 작업 [RebootDBInstance](#)를 실행합니다.

리소스 유형

- aws:rds:db

파라미터

- forceFailover – 선택 사항입니다. 값이 true이고 인스턴스가 다중 AZ인 경우, 한 가용 영역에서 다른 가용 영역으로 강제 장애 조치합니다. 기본값은 false입니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorRDSAccess](#)

Amazon S3 작업

AWS FIS 다음 Amazon S3 작업을 지원합니다.

작업

- [aws:s3:bucket-pause-replication](#)

aws:s3:bucket-pause-replication

대상 소스 버킷에서 대상 버킷으로의 복제를 일시 중지합니다. 대상 버킷은 다른 AWS 리전에 있을 수도 있고 원본 버킷과 동일한 리전 내에 있을 수도 있습니다. 기존 객체는 작업이 시작된 후 최대 1시간 동안 계속 복제될 수 있습니다. 이 작업은 태그별 타겟팅만 지원합니다. Amazon S3 복제에 대해 자세히 알아보려면 [Amazon S3 사용 설명서](#)를 참조하세요.

리소스 유형

- aws:s3:bucket

파라미터

- `duration` - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.
- `region` - 대상 버킷이 위치한 AWS 리전입니다.
- `destinationBuckets` - 선택 사항입니다. 쉼표로 구분된 대상 S3 버킷 목록입니다.
- `prefixes` - 선택 사항입니다. 복제 규칙 필터에서 쉼표로 구분된 S3 객체 키 접두사 목록입니다. 접두사를 기반으로 하는 필터가 있는 대상 버킷의 복제 규칙이 일시 중지됩니다.

권한

- 조건 키 `S3:IsReplicationPauseRequest`가 `True`로 설정된 `S3:PutReplicationConfiguration`
- 조건 키 `S3:IsReplicationPauseRequest`가 `True`로 설정된 `S3:GetReplicationConfiguration`
- `S3:PauseReplication`
- `S3:ListAllMyBuckets`
- `tag:GetResources`

정책 예제는 [예: `aws:s3:bucket-pause-replication`의 조건 키 사용](#)을 참조하세요.

Systems Manager 작업

AWS FIS 는 다음과 같은 Systems Manager 작업을 지원합니다.

작업

- [aws:ssm:send-command](#)
- [aws:ssm:start-automation-execution](#)

aws:ssm:send-command

대상 EC2 인스턴스에서 Systems Manager API 작업을 [SendCommand](#) 실행합니다. Systems Manager 문서(SSM 문서)는 Systems Manager가 인스턴스에서 수행하는 작업을 정의합니다. 자세한 정보는 [aws:ssm:send-command 작업을 사용](#)을 참조하세요.

리소스 유형

- `aws:ec2:instance`

파라미터

- `documentArn` - 문서의 Amazon 리소스 이름(ARN)입니다. 콘솔에서 [사전 구성된 AWS FIS SSM 문서](#) 중 하나에 해당하는 Action type 값을 선택하면 이 파라미터가 자동으로 완성됩니다.
- `documentVersion` - 선택 사항입니다. 문서의 버전입니다. 비어 있으면 기본 버전이 실행됩니다.
- `documentParameters` - 조건부. 문서에 허용되는 필수 및 선택 파라미터. 형식은 키가 문자열이고 값이 문자열 또는 문자열 배열인 JSON 객체입니다.
- `duration` - 소요 시간(1분에서 12시간). AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

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

AWS 관리형 정책

- [AWSFaultInjectionSimulatorEC2Access](#)

`aws:ssm:start-automation-execution`

Systems Manager API 작업을 실행합니다 [StartAutomationExecution](#).

리소스 유형

- None

파라미터

- `documentArn` - 자동화 문서의 Amazon 리소스 이름(ARN)입니다.
- `documentVersion` - 선택 사항입니다. 문서의 버전입니다. 비어 있으면 기본 버전이 실행됩니다.

- `documentParameters` - 조건부. 문서에 허용되는 필수 및 선택 파라미터. 형식은 키가 문자열이고 값이 문자열 또는 문자열 배열인 JSON 객체입니다.
- `maxDuration` - 자동화 실행이 완료되는 데 허용되는 최대 시간은 1분에서 12시간입니다. AWS FIS API에서 값은 ISO 8601 형식의 문자열입니다. 예를 들어, PT1M은 1분을 나타냅니다. AWS FIS 콘솔에서 초, 분 또는 시간을 입력합니다.

권한

- `ssm:GetAutomationExecution`
- `ssm:StartAutomationExecution`
- `ssm:StopAutomationExecution`
- `iam:PassRole` - 선택 사항입니다. 자동화 문서가 역할을 맡는 경우 필수입니다.

AWS 관리형 정책

- [AWSFaultInjectionSimulatorSSMAccess](#)

FIS와 함께 AWS Systems Manager SSM 문서 사용

AWS FIS는 AWS Systems Manager SSM 에이전트와 FIS 작업을 통해 사용자 지정 장애 유형을 지원합니다. AWS [aws:ssm:send-command](#) 일반적인 오류 삽입 조치를 생성하는 데 사용할 수 있는 사전 구성된 Systems Manager SSM 문서 (SSM AWS 문서)는 - 접두사로 시작하는 공개 문서로 제공됩니다. AWSFIS

SSM 에이전트는 Amazon EC2 인스턴스, 온프레미스 서버 또는 가상 머신(VM)에 설치 및 구성할 수 있는 Amazon 소프트웨어입니다. 이를 통해 Systems Manager가 이러한 리소스를 관리할 수 있습니다. 에이전트는 Systems Manager의 요청을 처리하고 요청에 지정된 대로 실행합니다. 자체 SSM 문서를 포함하여 사용자 지정 오류를 삽입하거나 Amazon 소유의 공개 문서 중 하나를 참조할 수 있습니다.

요구 사항

SSM 에이전트가 대상에서 작업을 실행해야 하는 작업의 경우 다음을 확인해야 합니다.

- 에이전트가 대상에 설치되어 있습니다. SSM 에이전트는 일부 Amazon Machine Images(AMI)에 기본적으로 설치됩니다. 그렇지 않으면 인스턴스에 SSM 에이전트를 설치할 수 있습니다. 자세한 정보는 AWS Systems Manager 사용 설명서의 [EC2 인스턴스에 수동으로 SSM 에이전트 설치](#)를 참조하세요.

- Systems Manager에는 인스턴스에서 작업을 수행할 수 있는 권한이 있습니다. IAM 인스턴스 프로파일을 사용하여 액세스 권한을 부여합니다. 자세한 내용은 AWS Systems Manager 사용 설명서의 [Systems Manager용 IAM 인스턴스 프로파일 생성](#) 및 [EC2 인스턴스에 IAM 인스턴스 프로파일 연결](#)을 참조하세요.

aws:ssm:send-command 작업을 사용

SSM 문서는 Systems Manager가 관리형 인스턴스에서 실행하는 작업을 정의합니다. Systems Manager에는 사전 구성된 여러 문서가 포함되어 있으며 사용자가 직접 만들 수도 있습니다. SSM 문서를 직접 만드는 방법에 대한 자세한 내용은 AWS Systems Manager 사용 설명서의 [Systems Manager 문서 생성](#)을 참조하세요. SSM 문서 전반에 대한 자세한 내용은 AWS Systems Manager 사용 설명서의 [AWS Systems Manager 문서](#)를 참조하세요.

AWS FIS는 사전 구성된 SSM 문서를 제공합니다. [콘솔의 문서 \(https://console.aws.amazon.com/systems-manager/documents\)](https://console.aws.amazon.com/systems-manager/documents)에서 사전 구성된 SSM 문서를 볼 수 있습니다. AWS Systems Manager FIS 콘솔에서 사전 구성된 다양한 문서 중에서 선택할 수도 있습니다. AWS 자세한 정보는 [사전 구성된 AWS FIS SSM 문서](#)을 참조하세요.

AWS FIS 실험에서 SSM 문서를 사용하려면 작업을 사용하면 됩니다. [aws:ssm:send-command](#) 이 작업은 대상 인스턴스에서 지정된 SSM 문서를 가져와 실행합니다.

실험 템플릿에서 `aws:ssm:send-command` 작업을 사용할 때는 다음을 포함하여 작업에 대한 추가 파라미터를 지정해야 합니다.

- `documentArn` - 필수입니다. SSM 문서의 Amazon 리소스 이름(ARN)입니다.
- `documentParameters` - 조건부. SSM 문서에 허용되는 필수 및 선택 파라미터. 형식은 키가 문자열이고 값이 문자열 또는 문자열 배열인 JSON 객체입니다.
- `documentVersion` - 선택 사항입니다. 실행할 SSM 문서의 버전입니다.

Systems Manager 콘솔 또는 명령줄을 사용하여 SSM 문서에 대한 정보(문서 파라미터 포함)를 볼 수 있습니다.

콘솔을 사용하여 SSM 문서에 대한 정보를 확인하려면

1. <https://console.aws.amazon.com/systems-manager/>에서 AWS Systems Manager 콘솔을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 문서를 선택하고 세부 정보 탭을 선택합니다.

명령줄을 사용하여 SSM 문서에 대한 정보를 확인하려면

SSM [describe-document](#) 명령을 사용합니다.

사전 구성된 AWS FIS SSM 문서

실험 템플릿의 작업과 함께 사전 구성된 AWS FIS SSM 문서를 사용할 수 있습니다. `aws:ssm:send-command`

요구 사항

- AWS FIS에서 제공하는 사전 구성된 SSM 문서는 다음 운영 체제에서만 지원됩니다.
 - Amazon Linux 2023, Amazon Linux 2, Amazon Linux
 - Ubuntu
 - RHEL 7, 8, 9
 - CentOS 7, 8, 9
- AWS FIS에서 제공하는 사전 구성된 SSM 문서는 EC2 인스턴스에서만 지원됩니다. 온프레미스 서버 등, 다른 유형의 관리형 노드에서는 지원되지 않습니다.

이러한 SSM 문서를 ECS 작업 실험에 사용하려면 해당 [the section called “Amazon ECS 작업”](#)를 사용하세요. 예를 들어, `aws:ecs:task-cpu-stress` 작업에는 `AWSFIS-Run-CPU-Stress` 문서가 사용됩니다.

문서

- [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](#)

AWSFIS-Run-CPU-Stress

stress-ng 도구를 사용하여 인스턴스에서 CPU 스트레스를 실행합니다. [-RUN-CPU-Stress SSM 문서를 사용합니다.](#) [AWSFIS](#)

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- DurationSeconds - 필수입니다. CPU 스트레스 테스트의 지속 시간(초)입니다.
- CPU - 선택 사항입니다. 사용할 CPU 스트레스 요인의 수입니다. 기본값은 0이며, 모든 CPU 스트레스 요인을 사용합니다.
- LoadPercent - 선택 사항입니다. 0(무부하)에서 100(완전 부하)까지의 목표 CPU 부하 백분율. 기본값은 100입니다.
- InstallDependencies - 선택 사항입니다. 값이 True이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 stress-ng입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-Disk-Fill

인스턴스의 루트 볼륨에 디스크 공간을 할당하여 디스크 전체 오류를 시뮬레이션합니다. [AWSFIS-Run-Disk-Fill SSM](#) 문서를 사용합니다.

이 오류를 발생시키는 실험이 수동 또는 중지 조건을 통해 중지되면 AWS FIS는 실행 중인 SSM 문서를 취소하여 롤백을 시도합니다. 그러나 오류 또는 오류와 애플리케이션 작업으로 인해 디스크가 100% 가득 차면 Systems Manager에서 취소 작업을 완료하지 못할 수 있습니다. 따라서 실험을 중단해야 하는 경우 디스크가 100% 가득 차지 않도록 하세요.

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- **DurationSeconds** - 필수입니다. 디스크 채우기 테스트의 지속 시간(초)입니다.
- **Percent** - 선택 사항입니다. 디스크 채우기 테스트 중에 할당할 디스크의 비율입니다. 기본값은 95%입니다.
- **InstallDependencies** - 선택 사항입니다. 값이 True이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 atd 및 fallocate입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-IO-Stress

stress-ng 도구를 사용하여 인스턴스에서 IO 스트레스를 실행합니다. [-RUN-IO-stress SSM 문서를 사용합시다. AWSFIS](#)

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- **DurationSeconds** - 필수입니다. IO 스트레스 테스트의 지속 시간(초)입니다.
- **Workers** - 선택 사항입니다. 순차, 임의 및 메모리 매핑된 읽기/쓰기 작업, 강제 동기화 및 캐시 삭제를 혼합하여 수행하는 작업자 수. 여러 하위 프로세스가 동일한 파일에서 다양한 I/O 작업을 수행합니다. 기본 값은 1입니다.
- **Percent** - 선택 사항입니다. IO 스트레스 테스트 중에 사용할 파일 시스템의 여유 공간 비율입니다. 기본값은 80%입니다.

- `InstallDependencies` – 선택 사항입니다. 값이 `True`이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 `True`입니다. 종속성은 `stress-ng`입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-Kill-Process

`killall` 명령을 사용하여 인스턴스에서 지정된 프로세스를 중지합니다. [-Run-Kill-Process SSM 문서를 사용하십시오. AWSFIS](#)

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- `ProcessName` - 필수입니다. 중지할 프로세스의 이름입니다.
- `Signal` – 선택 사항입니다. 명령과 함께 전송할 신호입니다. 가능한 값은 `SIGTERM`(수신자가 무시하도록 선택할 수 있는 값)과 `SIGKILL`(무시할 수 없는 값)입니다. 기본값은 `SIGTERM`입니다.
- `InstallDependencies` – 선택 사항입니다. 값이 `True`이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 `True`입니다. 종속성은 `killall`입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-Memory-Stress

`stress-ng` 도구를 사용하여 인스턴스에 메모리 스트레스를 실행합니다. [AWSFIS-런-메모리-스트레스 SSM](#) 문서를 사용하십시오.

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- DurationSeconds - 필수입니다. 메모리 스트레스 테스트의 지속 시간(초)입니다.
- Workers - 선택 사항입니다. 가상 메모리 스트레스 요인의 수. 기본 값은 1입니다.
- Percent - 필수입니다. 메모리 스트레스 테스트 중에 사용할 가상 메모리의 비율입니다.
- InstallDependencies - 선택 사항입니다. 값이 True이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 stress-ng입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-Network-Blackhole-Port

iptables 도구를 사용하여 프로토콜 및 포트에 대한 인바운드 또는 아웃바운드 트래픽을 삭제합니다. [Run-Network-Blackhole-Port SSM 문서를 사용하십시오. AWSFIS](#)

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- Protocol - 필수입니다. 프로토콜. 가능한 값은 tcp와 udp입니다.
- Port - 필수입니다. 포트 번호입니다.

- **TrafficType** – 선택 사항입니다. 트래픽 유형입니다. 가능한 값은 `ingress`와 `egress`입니다. 기본값은 `ingress`입니다.
- **DurationSeconds** - 필수입니다. 네트워크 블랙홀 테스트 시간(초)입니다.
- **InstallDependencies** – 선택 사항입니다. 값이 `True`이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 `True`입니다. 종속성은 `atd`, `dig` 및 `iptables`입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-Network-Latency

`tc` 도구를 사용하여 네트워크 인터페이스에 지연 시간을 추가합니다. [AWSFIS-Run-Network-Latency SSM](#) 문서를 사용합니다.

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- **Interface** – 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 `eth0`입니다.
- **DelayMilliseconds** – 선택 사항입니다. 밀리 초 단위의 지연 시간입니다. 기본값은 200입니다.
- **DurationSeconds** - 필수입니다. 네트워크 지연 시간 테스트 시간(초)입니다.
- **InstallDependencies** – 선택 사항입니다. 값이 `True`이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 `True`입니다. 종속성은 `atd`, `dig` 및 `tc`입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-Network-Latency-Sources

특정 소스에서 오가는 트래픽을 처리하는 tc 도구를 사용하여 네트워크 인터페이스에 지연 시간과 지터를 추가합니다. [AWSFIS-Run-Network-Latency-Sources](#) SSM 문서를 사용합니다.

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- **Interface** – 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 eth0입니다.
- **DelayMilliseconds** – 선택 사항입니다. 밀리 초 단위의 지연 시간입니다. 기본값은 200입니다.
- **JitterMilliseconds** – 선택 사항입니다. 밀리 초 단위의 지터입니다. 기본값은 10.
- **Sources** - 필수입니다. 쉘표로 구분된 소스입니다. 가능한 값은 IPv4 주소, IPv4 CIDR 블록, 도메인 이름, DYNAMODB 및 S3입니다. DYNAMODB 또는 S3를 지정하는 경우 이는 현재 리전의 리전 엔드포인트에만 적용됩니다.
- **TrafficType** – 선택 사항입니다. 트래픽 유형입니다. 가능한 값은 ingress와 egress입니다. 기본값은 ingress입니다.
- **DurationSeconds** - 필수입니다. 네트워크 지연 시간 테스트 시간(초)입니다.
- **InstallDependencies** – 선택 사항입니다. 값이 True이면 Systems Manager가 대상 인스턴스에 필요한 종속성을 설치합니다(아직 설치되지 않은 경우). 기본값은 True입니다. 종속성은 atd, dig, jq 및 tc입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

```
{"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

tc 도구를 사용하여 네트워크 인터페이스에 패킷 손실을 추가합니다. [AWSFIS-Run-Network-Packet-Loss](#) SSM 문서를 사용합니다.

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- Interface – 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 eth0입니다.
- LossPercent – 선택 사항입니다. 패킷 손실의 백분율. 기본값은 7%입니다.
- DurationSeconds - 필수입니다. 네트워크 패킷 손실 테스트 시간(초)입니다.
- InstallDependencies – 선택 사항입니다. 값이 True인 경우 Systems Manager는 대상 인스턴스에 필요한 종속성을 설치합니다. 기본값은 True입니다. 종속성은 atd, dig 및 tc입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

AWSFIS-Run-Network-Packet-Loss-Sources

특정 소스에서 오가는 트래픽을 처리하는 tc 도구를 사용하여 네트워크 인터페이스에 패킷 손실을 추가합니다. [AWSFIS-Run-Network-Packet-Loss-Sources SSM](#) 문서를 사용합니다.

작업 유형(콘솔 전용)

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

ARN

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

문서 파라미터

- Interface – 선택 사항입니다. 네트워크 인터페이스입니다. 기본값은 eth0입니다.
- LossPercent – 선택 사항입니다. 패킷 손실의 백분율. 기본값은 7%입니다.
- Sources - 필수입니다. 쉘표로 구분된 소스입니다. 가능한 값은 IPv4 주소, IPv4 CIDR 블록, 도메인 이름, DYNAMODB 및 S3입니다. DYNAMODB 또는 S3를 지정하는 경우 이는 현재 리전의 리전 엔드포인트에만 적용됩니다.

- **TrafficType** – 선택 사항입니다. 트래픽 유형입니다. 가능한 값은 `ingress`와 `egress`입니다. 기본값은 `ingress`입니다.
- **DurationSeconds** - 필수입니다. 네트워크 패킷 손실 테스트 시간(초)입니다.
- **InstallDependencies** – 선택 사항입니다. 값이 `True`인 경우 **Systems Manager**는 대상 인스턴스에 필요한 종속성을 설치합니다. 기본값은 `True`입니다. 종속성은 `atd`, `dig`, `jq` 및 `tc`입니다.

다음은 콘솔에 입력할 수 있는 문자열의 예입니다.

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

예제

실험 템플릿 예시를 보려면 [the section called “사전 구성된 AWS FIS SSM 문서 실행” 단원을 참조하세요.](#)

자습서 예시는 [인스턴스에서 CPU 스트레스 실행](#) 단원을 참조하세요.

문제 해결

다음 절차를 사용하여 문제를 해결하세요.

SSM 문서 관련 문제를 해결하려면

1. <https://console.aws.amazon.com/systems-manager/> 에서 **AWS Systems Manager 콘솔을 엽니다.**
2. 탐색 창에서 **노드 관리**, **명령 실행**을 선택합니다.
3. 명령 기록 탭에서 필터를 사용하여 문서 실행 위치를 찾습니다.
4. 명령의 ID를 선택하여 세부 정보 페이지를 엽니다.
5. 인스턴스의 ID를 선택합니다. 각 단계의 출력 및 오류를 검토하세요.

AWS FIS `aws:ecs:태스크` 액션을 사용하세요.

`aws:ecs:task` 작업을 사용하여 Amazon ECS 태스크에 오류를 주입할 수 있습니다.

이러한 작업은 SSM 에이전트를 사이드카 컨테이너로 사용하여 결합 주입을 수행할 SSM 문서를 실행하고 사이드카 컨테이너를 통해 Amazon ECS 태스크를 SSM 관리 인스턴스로 등록합니다. 이러

한 작업을 사용하려면 SSM 에이전트를 사이드카 컨테이너로 추가하여 실행 중인 태스크를 SSM 관리 인스턴스로 등록할 수 있도록 Amazon ECS 태스크 정의를 업데이트해야 합니다. AWS FIS 실험 타겟팅을 `aws:ecs:task` 실행하면 AWS FIS는 관리형 인스턴스에 추가된 리소스 태그를 사용하여 AWS FIS 실험 템플릿에서 지정한 대상 Amazon ECS 작업을 SSM 관리형 인스턴스 세트에 매핑합니다. `ECS_TASK_ARN` 이 태그 값은 SSM 문서가 실행되어야 하는 연결된 Amazon ECS 태스크의 ARN 이므로 실험을 실행할 때 제거해서는 안 됩니다.

작업

- [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”](#)

제한 사항

- 다음 작업은 사용할 수 없습니다. AWS Fargate
 - `aws:ecs:task-kill-process`
 - `aws:ecs:task-network-blackhole-port`
 - `aws:ecs:task-network-latency`
 - `aws:ecs:task-network-packet-loss`
- ECS Exec을 활성화한 경우, 이 작업을 사용할 수 있으려면 이 기능을 먼저 비활성화해야 합니다.

요구 사항

- AWS FIS [실험 역할에](#) 다음 권한을 추가합니다.
 - `ssm:SendCommand`
 - `ssm:ListCommands`
 - `ssm:CancelCommand`
- 다음 권한을 Amazon ECS [작업 IAM 역할에](#) 추가합니다.
 - `ssm:CreateActivation`

- `ssm:AddTagsToResource`
- `iam:PassRole`

참고로 관리형 인스턴스 역할의 ARN을 `iam:PassRole`의 리소스로 지정할 수 있습니다.

- [Amazon ECS 작업 실행 IAM 역할을 생성하고 AmazonECS 관리형 정책을 추가합니다.](#)
[TaskExecutionRolePolicy](#)
- 관리형 인스턴스로 등록된 작업에 연결된 관리형 인스턴스 역할에 다음 권한을 추가합니다.
 - `ssm:DeleteActivation`
 - `ssm:DeregisterManagedInstance`
- 관리형 인스턴스로 등록된 작업에 연결된 `ManagedInstanceCore` 관리형 인스턴스 역할에 [AmazonSSM](#) 관리형 정책을 추가합니다.
- 환경 변수 `MANAGED_INSTANCE_ROLE_NAME`를 관리형 인스턴스 역할 이름으로 설정합니다.
- SSM 에이전트 컨테이너를 ECS 작업 정의에 추가합니다. 명령 스크립트는 ECS 작업을 관리형 인스턴스로 등록합니다.

```
{
  "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; yum upgrade -y; yum 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_SIDEAR,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": []
}

```

더 읽기 쉬운 스크립트 버전은 [the section called “스크립트의 참조 버전”](#)을 참조하세요.

- `aws:ecs:task-network-blackhole-port`, `aws:ecs:task-network-latency` 및 `aws:ecs:task-network-packet-loss` 작업을 사용하는 경우 다음 옵션 중 하나를 사용하여 ECS 작업 정의의 SSM 에이전트 컨테이너를 업데이트해야 합니다.
- 옵션 1 - 특정 Linux 기능을 추가합니다.

```
"linuxParameters": {
  "capabilities": {
    "add": [
      "NET_ADMIN"
    ]
  }
},
```

- 옵션 2 - 모든 Linux 기능을 추가합니다.

```
"privileged": true,
```

- `aws:ecs:task-kill-process`, `aws:ecs:task-network-blackhole-port`, `aws:ecs:task-network-latency` 및 `aws:ecs:task-network-packet-loss` 작업을 사용하는 경우 ECS 작업 정의에서 `pidMode`가 `task`로 설정되어 있어야 합니다.

스크립트의 참조 버전

다음은 참조용으로 요구 사항 섹션에 있는 더 읽기 쉬운 버전의 스크립트입니다.

```
#!/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_SIDE CAR: 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

yum upgrade -y
yum 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_SIDEDECAR,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
```

실험 템플릿 예시

다음은 [the section called "aws:ecs:task-cpu-stress"](#) 작업에 대한 예제 실험 템플릿입니다.

```
{
  "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": {}
}
```

}

FIS aws:eks:pod 액션을 사용하십시오. AWS

aws:eks:pod 작업을 사용하여 EKS 클러스터에서 실행되는 Kubernetes 포드에 오류를 주입할 수 있습니다.

작업

- [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”](#)

제한 사항

- 다음 조치는 사용할 수 없습니다. AWS Fargate
 - aws:eks:pod-network-blackhole-port
 - aws:eks:pod-network-latency
 - aws:eks:pod-network-packet-loss
- 다음 작업은 bridge [네트워크 모드](#)를 지원하지 않습니다.
 - aws:eks:pod-network-blackhole-port
 - aws:eks:pod-network-latency
 - aws:eks:pod-network-packet-loss
- 실험 템플릿에서는 리소스 ARN 또는 리소스 태그를 사용하여 aws:eks:pod 유형의 대상을 식별할 수 없습니다. 필수 리소스 파라미터를 사용하여 대상을 식별해야 합니다.
- aws:eks:pod-network-latency 및 aws:eks:pod-network-packet-loss 작업은 동일한 포드를 대상으로 병렬로 실행해서는 안 됩니다. 지정한 maxErrors 파라미터의 값에 따라 작업이 완료 또는 실패 상태로 종료될 수 있습니다.
 - maxErrorsPercent가 0(기본값)이면 작업이 failed 상태로 종료됩니다.

- 그렇지 않으면 실패가 `maxErrorsPercent` 예산에 합산됩니다. 실패한 주입 횟수가 제공된 `maxErrors`에 도달하지 않으면 작업이 완료된 상태로 종료됩니다.
- 대상 포드에 주입된 임시 컨테이너의 로그에서 이러한 실패를 식별할 수 있습니다. `Exit Code: 16`와 함께 실패합니다.
- `aws:eks:pod-network-blackhole-port` 작업은 동일한 포드를 대상으로 하고 동일한 `trafficType`을 사용하는 다른 작업과 병렬로 실행해서는 안 됩니다. 서로 다른 트래픽 유형을 사용하는 병렬 작업이 지원됩니다.
- FIS는 대상 포드의 `securityContext`가 `readOnlyRootFilesystem: false`로 설정된 경우에만 결합 주입 상태를 모니터링할 수 있습니다. 이 구성이 없으면 모든 EKS 포드 작업이 실패합니다.

요구 사항

- AWS CLI 컴퓨터에 를 설치합니다. 이는 AWS CLI 를 사용하여 IAM 역할을 생성하는 경우에만 필요합니다. 자세한 내용은 [AWS CLI 설치 또는 업데이트](#)를 참조하세요.
- 컴퓨터에 `kubectl`를 설치합니다. 이는 EKS 클러스터와 상호 작용하여 대상 애플리케이션을 구성하거나 모니터링하는 경우에만 필요합니다. 자세한 내용은 <https://kubernetes.io/docs/tasks/tools/>를 참조하세요.
- 현재 지원되는 최소 EKS 버전은 1.23입니다.

Kubernetes 서비스 계정에 대한 서비스 역할 생성

서비스 역할로 사용할 IAM 역할을 생성합니다. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.

Kubernetes 서비스 계정 구성

지정된 Kubernetes 네임스페이스의 대상으로 실험을 실행하도록 Kubernetes 서비스 계정을 구성합니다. 다음 예제에서 서비스 계정은 `myserviceaccount`이고 네임스페이스는 `###`입니다. 참고로 `default`는 표준 Kubernetes 네임스페이스 중 하나입니다.

Kubernetes 서비스 계정을 구성하려면

1. 이름이 `rbac.yaml`인 파일을 만들고 다음을 추가합니다.

```
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. 다음 명령을 실행합니다.

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

실험 역할을 Kubernetes 사용자에게 매핑

다음 명령을 사용하여 ID 매핑을 생성합니다. 자세한 내용은 eksctl 설명서의 [IAM 사용자 및 역할 관리](#)를 참조하세요.

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

포드 컨테이너 이미지

AWS FIS에서 제공하는 포드 컨테이너 이미지는 Amazon ECR에서 호스팅됩니다. Amazon ECR의 이미지를 참조할 때는 전체 이미지 URI를 사용해야 합니다.

AWS 리전	이미지 URI
미국 동부(오하이오)	051821878176.dkr.ecr.us-east-2.amazonaws.com/aws-fis-pod:0.1
미국 동부(버지니아 북부)	731367659002.dkr.ecr.us-east-1.amazonaws.com/aws-fis-pod:0.1
미국 서부(캘리포니아 북부)	080694859247.dkr.ecr.us-west-1.amazonaws.com/aws-fis-pod:0.1
미국 서부(오레곤)	864386544765.dkr.ecr.us-west-2.amazonaws.com/aws-fis-pod:0.1
아프리카(케이프타운)	056821267933.dkr.ecr.af-south-1.amazonaws.com/aws-fis-pod:0.1

AWS 리전	이미지 URI
아시아 태평양(홍콩)	246405402639.dkr.ecr.ap-east-1.amazonaws.com/aws-fis-pod:0.1
아시아 태평양(뭄바이)	524781661239.dkr.ecr.ap-south-1.amazonaws.com/aws-fis-pod:0.1
아시아 태평양(서울)	526524659354.dkr.ecr.ap-northeast-2.amazonaws.com/aws-fis-pod:0.1
아시아 태평양(싱가포르)	316401638346.dkr.ecr.ap-southeast-1.amazonaws.com/aws-fis-pod:0.1
아시아 태평양(시드니)	488104106298.dkr.ecr.ap-southeast-2.amazonaws.com/aws-fis-pod:0.1
아시아 태평양(도쿄)	635234321696.dkr.ecr.ap-northeast-1.amazonaws.com/aws-fis-pod:0.1
캐나다(중부)	490658072207.dkr.ecr.ca-central-1.amazonaws.com/aws-fis-pod:0.1
유럽(프랑크푸르트)	713827034473.dkr.ecr.eu-central-1.amazonaws.com/aws-fis-pod:0.1
유럽(아일랜드)	205866052826.dkr.ecr.eu-west-1.amazonaws.com/aws-fis-pod:0.1
유럽(런던)	327424803546.dkr.ecr.eu-west-2.amazonaws.com/aws-fis-pod:0.1
유럽(밀라노)	478809367036.dkr.ecr.eu-south-1.amazonaws.com/aws-fis-pod:0.1
유럽(파리)	154605889247.dkr.ecr.eu-west-3.amazonaws.com/aws-fis-pod:0.1
유럽(스톡홀름)	263175118295.dkr.ecr.eu-north-1.amazonaws.com/aws-fis-pod:0.1

AWS 리전	이미지 URI
중동(바레인)	065825543785.dkr.ecr.me-south-1.amazonaws.com/ aws-fis-pod:0.1
남아메리카(상파울루)	767113787785.dkr.ecr.sa-east-1.amazonaws.com/aws- fis-pod:0.1
AWS GovCloud (미국 동부)	246533647532.dkr.ecr.us-gov-east-1.amazonaws.com/ aws-fis-pod:0.1
AWS GovCloud (미국 서부)	246529956514.dkr.ecr.us-gov-west-1.amazonaws.com/ aws-fis-pod:0.1

실험 템플릿 예시

다음은 [the section called "aws:eks:pod-network-latency"](#) 작업에 대한 예제 실험 템플릿입니다.

```
{
  "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"
}
}

```

다음을 사용하여 AWS FIS 작업을 나열하십시오. AWS CLI

AWS Command Line Interface (AWS CLI) 를 사용하여 AWS FIS 지원하는 작업에 대한 정보를 볼 수 있습니다.

전제 조건

AWS CLI 컴퓨터에 를 설치합니다. 시작하려면 [AWS Command Line Interface 사용 설명서](#)를 참조하십시오. 의 명령에 대한 자세한 내용은 AWS CLI 명령 참조서의 [fis](#)를 참조하십시오. AWS FIS

예: 모든 작업의 이름 나열

다음과 같이 [list-actions](#) 명령을 사용하여 모든 작업의 이름을 나열할 수 있습니다.

```
aws fis list-actions --query "actions[*].[id]" --output text | sort
```

출력의 예제는 다음과 같습니다.

```
aws:cloudwatch:assert-alarm-state
aws:dynamodb:encrypted-global-table-pause-replication
aws:ebs:pause-volume-io
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:ecs:drain-container-instances
aws:ecs:stop-task
aws:eks:inject-kubernetes-custom-resource
aws:eks:terminate-nodegroup-instances
aws:elasticache:interrupt-cluster-az-power
aws:fis:inject-api-internal-error
aws:fis:inject-api-throttle-error
aws:fis:inject-api-unavailable-error
aws:fis:wait
aws:network:disrupt-connectivity
aws:network:route-table-disrupt-cross-region-connectivity
aws:network:transit-gateway-disrupt-cross-region-connectivity
aws:rds:failover-db-cluster
aws:rds:reboot-db-instances
aws:s3:bucket-pause-replication
aws:ssm:send-command
aws:ssm:start-automation-execution
```

예: 작업에 대한 정보 보기

작업의 이름을 알고 나면 다음과 같이 [get-action](#) 명령을 사용하여 작업에 대한 세부 정보를 볼 수 있습니다.

```
aws fis get-action --id aws:ec2:reboot-instances
```

출력의 예제는 다음과 같습니다.

```
{
  "action": {
    "id": "aws:ec2:reboot-instances",
    "description": "Reboot the specified EC2 instances.",
    "targets": {
      "Instances": {
        "resourceType": "aws:ec2:instance"
      }
    },
    "tags": {}
  }
}
```

}

AWS FIS용 실험 템플릿

실험 템플릿에는 실험 중에 지정된 대상에서 실행할 수 있는 하나 이상의 작업이 포함되어 있습니다. 또한 실험이 범위를 벗어나는 것을 방지하는 중지 조건도 포함되어 있습니다. 실험 템플릿을 만든 후 이를 사용하여 실험을 실행할 수 있습니다.

템플릿 구성 요소

다음 구성 요소를 사용하여 실험 템플릿을 만들 수 있습니다.

작업 세트

실행하려는 [AWS FIS 작업](#). 지정한 순서대로 작업을 실행하거나 동시에 실행할 수 있습니다. 자세한 정보는 [작업 세트](#)를 참조하세요.

대상

특정 작업이 수행되는 데 사용되는 AWS 리소스. 자세한 정보는 [대상](#)을 참조하세요.

중지 조건

애플리케이션 성능이 만족스럽지 않은 임계값을 정의하는 CloudWatch 경보. 실험이 실행되는 동안 중지 조건이 트리거되면 AWS FIS는 실험을 중단합니다. 자세한 정보는 [중지 조건](#)을 참조하세요.

실험 역할

FIS가 사용자를 대신하여 실험을 실행할 수 있도록 필요한 권한을 AWS FIS에 부여하는 IAM 역할. 자세한 정보는 [실험 역할](#)을 참조하세요.

실험 옵션

실험 템플릿의 옵션입니다. 자세한 정보는 [실험 옵션](#)을 참조하세요.

계정에는 FIS 관련 할당량이 있습니다. AWS 예를 들어 실험 템플릿당 작업 수에는 할당량이 있습니다. 자세한 정보는 [제한 사항 및 할당량](#)을 참조하세요.

템플릿 구문

다음은 실험 템플릿의 구문입니다.

```
{
```

```

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

```

예를 보려면 [템플릿 예제](#) 섹션을 참조하십시오.

시작

를 사용하여 실험 템플릿을 만들려면 [을 참조하십시오](#). AWS Management Console [실험 템플릿 만들기](#)

를 사용하여 실험 템플릿을 만들려면 AWS CLI을 참조하십시오 [AWS FIS 실험 템플릿 예시](#).

AWS FIS용 액션 세트

실험 템플릿을 만들려면 작업 세트를 구성하는 작업을 하나 이상 정의해야 합니다. AWS FIS에서 제공하는 사전 정의된 동작 목록은 [을 참조하십시오](#). [작업](#)

실험 중에는 작업을 한 번만 실행할 수 있습니다. 동일한 실험에서 동일한 AWS FIS 동작을 두 번 이상 실행하려면 다른 이름을 사용하여 템플릿에 여러 번 추가하십시오.

내용

- [작업 구문](#)
- [작업 기간](#)
- [작업 예](#)

작업 구문

다음은 작업 세트에 대한 구문입니다.

```

{
  "actions": {
    "action_name": {
      "actionId": "aws:service:action-type",

```

```

    "description": "string",
    "parameters": {
      "name": "value"
    },
    "startAfter": ["action_name", ...],
    "targets": {
      "resource_type": "target_name"
    }
  }
}
}
}

```

작업을 정의할 때 다음을 제공합니다.

action_name

작업의 이름입니다.

actionId

[작업 식별자.](#)

description

설명(선택 사항)입니다.

parameters

[작업 파라미터.](#)

startAfter

이 작업을 시작하기 전에 완료해야 하는 모든 작업입니다. 그렇지 않으면 실험이 시작될 때 이 작업이 실행됩니다.

targets

[작업 대상.](#)

예를 보려면 [the section called “작업 예”](#) 섹션을 참조하십시오.

작업 기간

작업에 작업 기간을 지정하는 데 사용할 수 있는 파라미터가 포함된 경우 기본적으로 지정된 기간이 경과한 후에만 작업이 완료된 것으로 간주됩니다. emptyTargetResolutionMode 실험 옵션을

skip로 설정한 경우, 확인된 대상이 없는 경우 작업이 'skipped' 상태로 즉시 완료됩니다. 예를 들어 기간을 5분으로 지정하면 AWS FIS는 5분 후에 작업이 완료된 것으로 간주합니다. 그런 다음 모든 작업이 완료되어야 다음 작업을 시작합니다.

기간은 작업 조건이 유지되는 기간 또는 지표가 모니터링되는 기간일 수 있습니다. 예를 들어, 지정된 기간 동안 지연 시간이 주입됩니다. 인스턴스 종료와 같이 거의 즉각적인 작업 유형의 경우, 지정된 기간 동안 중지 조건이 모니터링됩니다.

작업 파라미터 내에 사후 작업이 포함된 작업의 경우, 사후 작업은 작업이 완료된 후에 실행됩니다. 사후 작업을 완료하는 데 걸리는 시간으로 인해 지정된 작업 기간과 다음 작업의 시작(또는 다른 모든 작업이 완료된 경우 실험 종료) 시점 사이에 지연이 발생할 수 있습니다.

작업 예

작업의 예를 들면 다음과 같습니다.

예제

- [EC2 인스턴스 중지](#)
- [스팟 인스턴스 중단](#)
- [네트워크 트래픽 방해](#)
- [EKS 작업자 종료](#)

예: EC2 인스턴스 중지

다음 작업은 *targetInstances*라는 대상을 사용하여 식별된 EC2 인스턴스를 중지합니다. 2분 후에 대상 인스턴스를 다시 시작합니다.

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

예: 스팟 인스턴스 중단

다음 작업은 이름이 지정된 대상을 사용하여 식별된 스팟 인스턴스를 중지합니다. *targetSpotInstances* 스팟 인스턴스가 중단될 때까지 2분 정도 기다립니다.

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

예: 네트워크 트래픽 방해

다음 작업은 대상 서브넷과 다른 가용 영역에 있는 서브넷 간의 트래픽을 거부합니다.

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

예: EKS 작업자 종료

다음 작업은 이름이 지정된 대상을 사용하여 식별된 EKS 클러스터의 EC2 인스턴스 중 50% 를 종료합니다. *targetNodeGroups*

```
"actions": {
```

```

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

```

AWS FIS 대상

대상은 실험 중에 AWS FIS (AWS 결함 주입 서비스)가 조치를 수행하는 하나 이상의 AWS 리소스입니다. 대상은 실험과 동일한 AWS 계정에 있을 수도 있고, 다중 계정 실험을 사용하여 다른 계정에 있을 수도 있습니다. 다른 계정의 리소스를 타겟팅하는 방법에 대해 자세히 알아보려면 [다중 계정 실험 단원](#)을 참조하세요.

[실험 템플릿을 생성](#)할 때 대상을 정의합니다. 실험 템플릿에서 동일한 대상을 여러 작업에 사용할 수 있습니다.

AWS FIS는 실험 시작 시 액션 세트의 작업을 시작하기 전에 모든 대상을 식별합니다. AWS FIS는 전체 실험에서 선택한 대상 리소스를 사용합니다. 대상이 발견되지 않으면 실험이 실패합니다.

목차

- [대상 구문](#)
- [리소스 유형](#)
- [대상 리소스 식별](#)
 - [리소스 필터](#)
 - [리소스 파라미터](#)
- [선택 모드](#)
- [예시 대상](#)
- [필터 예제](#)

대상 구문

다음은 대상에 대한 구문입니다.


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

대상을 정의할 때 다음을 제공합니다.

target_name

대상의 이름입니다.

resourceType

[리소스 유형](#).

resourceArns

특정 리소스의 Amazon 리소스 이름(ARN)입니다.

resourceTags

특정 리소스에 적용된 태그.

parameters

특정 속성을 사용하여 대상을 식별하는 [파라미터](#).

filters

[리소스 필터](#)는 특정 속성을 사용하여 식별된 대상 리소스의 범위를 지정합니다.

selectionMode

식별된 리소스의 [선택 모드](#).

예를 보려면 [the section called “예시 대상”](#) 섹션을 참조하십시오.

리소스 유형

각 AWS FIS 작업은 특정 AWS 리소스 유형에서 수행됩니다. 대상을 정의할 때 리소스 유형을 하나만 지정해야 합니다. 작업의 대상을 지정하는 경우, 대상은 작업에서 지원하는 리소스 유형이어야 합니다.

AWS FIS는 다음과 같은 리소스 유형을 지원합니다.

- aws:dynamodb: encrypted-global-table — 고객 관리 키로 암호화된 글로벌 테이블
- aws:ec2:autoscaling-group – Amazon EC2 Auto Scaling 그룹
- aws:ec2:ebs-volume – Amazon EBS 볼륨
- aws:ec2:instance – Amazon EC2 인스턴스
- aws:ec2:spot-instance – Amazon EC2 스팟 인스턴스
- aws:ec2:subnet – Amazon VPC 서브넷
- aws:ec2:transit-gateway – 전송 게이트웨이
- aws:ecs:cluster – Amazon ECS 클러스터
- aws:ecs:task – Amazon ECS 태스크
- aws:eks:cluster – Amazon EKS 클러스터
- aws:eks:nodegroup – Amazon EKS 노드 그룹
- aws:eks:pod – Kubernetes 포드
- aws:엘라스틱캐시:레디스-레플리케이션 그룹 — 레디스 리플리케이션 그룹 ElastiCache
- aws:iam:role – IAM 역할
- aws:rds:cluster – Amazon Aurora DB 클러스터
- aws:rds:db – Amazon RDS DB 인스턴스
- aws:s3:bucket – Amazon S3 버킷

대상 리소스 식별

AWS FIS 콘솔에서 대상을 정의할 때 타겟팅할 특정 리소스 (특정 리소스 유형) 를 선택할 수 있습니다. AWS 또는 제공된 기준에 따라 AWS FIS가 리소스 그룹을 식별하도록 할 수 있습니다.

대상 리소스를 식별하기 위해 다음을 지정할 수 있습니다.

- 리소스 ID — 특정 AWS 리소스의 리소스 ID입니다. 모든 리소스 ID는 동일한 유형의 리소스를 나타내야 합니다.
- 리소스 태그 — 특정 AWS 리소스에 적용되는 태그입니다.
- 리소스 필터 - 특정 속성을 가진 리소스를 나타내는 경로와 값입니다. 자세한 설명은 [리소스 필터](#) 섹션을 참조하세요.
- 리소스 파라미터 - 특정 기준을 충족하는 리소스를 나타내는 파라미터입니다. 자세한 설명은 [리소스 파라미터](#) 섹션을 참조하세요.

고려 사항

- 동일한 대상에 리소스 ID와 리소스 태그를 모두 지정할 수는 없습니다.
- 동일한 대상에 리소스 ID와 리소스 태그를 모두 지정할 수는 없습니다.
- 빈 태그 값으로 리소스 태그를 지정하는 경우 와일드카드와 동일하지 않습니다. 지정된 태그 키와 빈 태그 값이 있는 태그가 있는 리소스를 매칭합니다.

리소스 필터

리소스 필터는 특정 속성에 따라 대상 리소스를 식별하는 쿼리입니다. AWS FIS는 지정한 AWS 리소스 유형에 따라 리소스의 표준 설명이 포함된 API 작업의 출력에 쿼리를 적용합니다. 쿼리와 일치하는 속성을 가진 리소스는 대상 정의에 포함됩니다.

각 필터는 속성 경로 및 가능한 값으로 표현됩니다. 경로는 마침표로 구분된 일련의 요소로, 리소스에 대한 설명 작업 출력의 속성에 도달하기 위한 경로를 설명합니다. 리소스에 대한 Describe 작업의 출력이 카멜 표기법인 경우에도 각 요소는 파스칼 표기법으로 표현해야 합니다. 예를 들어, `availabilityZone`이 아니라 `AvailabilityZone`을 속성 요소로 사용해야 합니다.

```
"filters": [
  {
    "path": "component.component.component",
```

```

    "values": [
      "string"
    ]
  }
],

```

다음 표에는 각 리소스 유형에 대한 표준 설명을 가져오는 데 사용할 수 있는 API 작업 및 AWS CLI 명령이 포함되어 있습니다. AWS FIS는 사용자를 대신하여 이러한 작업을 실행하여 지정된 필터를 적용합니다. 해당 설명서에는 기본적으로 결과에 포함되는 리소스가 설명되어 있습니다. 예를 들어 DescribeInstances 설명서에는 최근에 종료된 인스턴스가 결과에 표시될 수 있다고 나와 있습니다.

리소스 유형	API 작업	AWS CLI 명령
aws:ec2:autoscaling-group	DescribeAutoScalingGroups	describe-auto-scaling-groups
aws:ec2:ebs-volume	DescribeVolumes	describe-volumes
aws:ec2:instance	DescribeInstances	describe-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	describe-tasks
aws:eks:cluster	DescribeClusters	describe-clusters
aws:eks:nodegroup	DescribeNodegroup	describe-nodegroup
aws:elasticache:redis-replicationgroup	DescribeReplicationGroups	describe-replication-groups
aws:iam:role	ListRoles	list-roles
aws:rds:cluster	DescribeDBClusters	describe-db-clusters
aws:rds:db	DescribeDBInstances	describe-db-instances
aws:s3:bucket	ListBuckets	list-buckets

다음 로직은 모든 리소스 필터에 적용됩니다.

- 필터 내 값 - OR
- 필터 전반의 값 - AND

예를 보려면 [the section called “필터 예제”](#) 섹션을 참조하십시오.

리소스 파라미터

리소스 파라미터는 특정 기준에 따라 대상 리소스를 식별합니다.

다음 리소스 유형은 파라미터를 지원합니다.

aws:ec2:ebs-volume

- `availabilityZoneIdentifier` - 대상 볼륨이 포함된 가용 영역의 코드(예: us-east-1a)입니다.

aws:ec2:subnet

- `availabilityZoneIdentifier` - 대상 서브넷이 포함된 가용 영역의 코드(예: us-east-1a) 또는 AZ ID(예: use1-az1)입니다.
- `vpc` - 대상 서브넷이 포함된 VPC입니다. 계정당 하나 이상의 VPC를 지원하지 않습니다.

aws:ecs:task

- `cluster` - 대상 작업이 포함된 클러스터입니다.
- `service` - 대상 작업이 포함된 서비스입니다.

aws:eks:pod

- `availabilityZoneIdentifier` - 선택 사항입니다. 대상 포드가 포함된 가용 영역입니다. 예를 들어 us-east-1d입니다. 포드의 호스트 IP와 클러스터 서브넷의 CIDR을 비교하여 포드의 가용 영역을 결정합니다.
- `clusterIdentifier` - 필수입니다. 대상 EKS 클러스터의 이름 또는 ARN.
- `namespace` - 필수입니다. 대상 포드의 Kubernetes 네임스페이스.
- `selectorType` - 필수입니다. 선택기 유형. 가능한 값은 `labelSelector`, `deploymentName`, `podName`입니다.
- `selectorValue` - 필수입니다. 선택기 값입니다. 이 값은 `selectorType`의 값에 따라 달라집니다.

- `targetContainerName` - 선택 사항입니다. 포드 사양에 정의된 대상 컨테이너의 이름입니다. 기본값은 각 대상 포드 사양에 정의된 첫 번째 컨테이너입니다.

`aws:rds:cluster`

- `writerAvailabilityZoneIdentifiers` - 선택 사항입니다. DB 클러스터 라이터의 가용 영역입니다. 가능한 값은 쉼표로 구분된 가용 영역 식별자 목록, `a11`입니다.

`aws:rds:db`

- `availabilityZoneIdentifiers` - 선택 사항입니다. 영향을 받을 DB 인스턴스의 가용 영역입니다. 가능한 값은 쉼표로 구분된 가용 영역 식별자 목록, `a11`입니다.

`aws:elasticache:redis-replicationgroup`

- `availabilityZoneIdentifier` - 필수입니다. 대상 노드가 포함된 가용성 영역의 코드(예: `us-east-1a`) 또는 AZ ID(예: `use1-az1`)입니다.

선택 모드

선택 모드를 지정하여 식별된 리소스의 범위를 지정합니다. AWS FIS는 다음과 같은 선택 모드를 지원합니다.

- ALL - 모든 대상에 대해 작업을 실행합니다.
- COUNT(*n*) - 식별된 대상에서 무작위로 선택한 지정된 수의 대상에 대해 작업을 실행합니다. 예를 들어, COUNT(1)은 식별된 대상 중 하나를 선택합니다.
- PERCENT(*n*) - 식별된 대상에서 무작위로 선택한 지정된 비율의 대상에 대해 작업을 실행합니다. 예를 들어, PERCENT(25)는 식별된 대상의 25%를 선택합니다.

리소스 수가 홀수이고 50% 를 지정하는 경우 AWS FIS는 반올림됩니다. 예를 들어 Amazon EC2 인스턴스 5개를 대상으로 추가하고 범위를 50% 로 설정하면 AWS FIS는 인스턴스 2개로 반올림합니다. 리소스 1개보다 작은 비율을 지정할 수 없습니다. 예를 들어 Amazon EC2 인스턴스 4개를 추가하고 범위를 5% 로 제한하는 경우 AWS FIS는 인스턴스를 선택할 수 없습니다.

동일한 대상 리소스 유형을 사용하여 여러 대상을 정의하는 경우 AWS FIS는 동일한 리소스를 여러 번 선택할 수 있습니다.

어떤 선택 모드를 사용하든, 지정한 범위에 리소스가 없는 것으로 확인되면 실험은 실패합니다.

예시 대상

예를 들면 다음과 같습니다.

예

- [지정된 태그가 있는 지정된 VPC의 인스턴스](#)
- [지정된 파라미터가 있는 작업](#)

예: 지정된 태그가 있는 지정된 VPC의 인스턴스

이 예제의 가능한 대상은 env=prod 태그가 있는 지정된 VPC의 Amazon EC2 인스턴스입니다. 선택 모드에서는 AWS FIS가 이러한 대상 중 하나를 임의로 선택하도록 지정합니다.

```
{
  "targets": {
    "randomInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceTags": {
        "env": "prod"
      },
      "filters": [
        {
          "path": "VpcId",
          "values": [
            "vpc-aabbcc11223344556"
          ]
        }
      ],
      "selectionMode": "COUNT(1)"
    }
  }
}
```

예: 지정된 파라미터가 있는 작업

이 예제의 가능한 대상은 지정된 클러스터 및 서비스를 사용하는 Amazon ECS 작업입니다. 선택 모드에서는 AWS FIS가 이러한 대상 중 하나를 무작위로 선택하도록 지정합니다.

```
{
  "targets": {
    "randomTask": {
      "resourceType": "aws:ecs:task",
      "parameters": {
```

```

        "cluster": "myCluster",
        "service": "myService"
    },
    "selectionMode": "COUNT(1)"
}
}
}

```

필터 예제

예를 들면 다음과 같습니다.

예

- [EC2 인스턴스](#)
- [DB 클러스터](#)

예: EC2 인스턴스

aws:ec2:instance 리소스 유형을 지원하는 작업에 대한 필터를 지정하면 AWS FIS는 Amazon EC2 describe-instances 명령을 사용하고 필터를 적용하여 대상을 식별합니다.

describe-instances 명령은 각 인스턴스가 Instances 아래 구조인 JSON 출력을 반환합니다. 다음은 **####**로 표시된 필드를 포함하는 부분 출력입니다. 이러한 필드를 사용하여 JSON 출력 구조에서 속성 경로를 지정하는 예제를 제공합니다.

```

{
  "Reservations": [
    {
      "Groups": [],
      "Instances": [
        {
          "ImageId": "ami-0011111111111111",
          "InstanceId": "i-00aaaaaaaaaaaaaaaa",
          "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-00bbbbbbbbbbbbbbbb",
    ...
  },
  ...
  {
    ...
  }
],
"OwnerId": "123456789012",
"ReservationId": "r-aaaaaabb111111"
},
...
]
}

```

리소스 필터를 사용하여 특정 가용 영역의 인스턴스를 선택하려면 AvailabilityZone의 속성 경로와 가용 영역의 코드를 값으로 지정하세요. 예:

```

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

```

리소스 필터를 사용하여 특정 서브넷의 인스턴스를 선택하려면 SubnetId의 속성 경로와 서브넷의 ID를 값으로 지정하세요. 예:

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

특정 인스턴스 상태에 있는 인스턴스를 선택하려면 Name의 속성 경로와 다음 상태 이름 중 하나를 값으로 지정하세요. pending | running | shutting-down | terminated | stopping | stopped. 예:

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

예: Amazon RDS 클러스터(DB 클러스터)

aws:rds:cluster 리소스 유형을 지원하는 작업에 대한 필터를 지정하면 FIS는 Amazon RDS describe-db-clusters 명령을 AWS 실행하고 필터를 적용하여 대상을 식별합니다.

describe-db-clusters 명령은 각 DB 클러스터에 대해 다음과 유사한 JSON 출력을 반환합니다. 다음은 ####로 표시된 필드를 포함하는 부분 출력입니다. 이러한 필드를 사용하여 JSON 출력 구조에서 속성 경로를 지정하는 예제를 제공합니다.

```
[
  {
    "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",
    ...
  }
]

```

특정 DB 엔진을 사용하는 DB 클러스터만 반환하는 리소스 필터를 적용하려면 다음 예와 같이 속성 경로를 Engine로 지정하고 값을 aurora-postgresql로 지정하세요.

```

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

```

특정 가용 영역의 DB 클러스터만 반환하는 리소스 필터를 적용하려면 다음 예와 같이 속성 경로와 값을 지정하세요.

```

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

```

AWS FIS의 중지 조건

AWS Fault Injection Service(AWS FIS)는 AWS 워크로드에서 실험을 안전하게 실행할 수 있는 제어 및 가드레일을 제공합니다. 중지 조건은 Amazon CloudWatch 경보로 정의한 임계값에 도달할 경우 실험을 중단하는 메커니즘입니다. 실험 중에 중지 조건이 트리거되면 AWS FIS에서 실험을 중지합니다. 중지된 실험은 재개할 수 없습니다.

중지 조건을 만들려면 먼저 애플리케이션 또는 서비스의 안정 상태를 정의하세요. 안정 상태는 애플리케이션이 최적의 성능을 발휘하는 시점으로, 비즈니스 또는 기술 지표로 정의됩니다. 지연 시간, CPU 부하, 재시도 횟수 등을 예로 들 수 있습니다. 애플리케이션 또는 서비스가 성능이 허용되지 않는 상태

에 도달할 경우 정상 상태를 사용하여 실험을 중단할 수 있는 CloudWatch 경보를 생성할 수 있습니다. 자세한 내용은 Amazon 사용 설명서의 [Amazon CloudWatch 경보 사용](#)을 참조하십시오. CloudWatch 사용자 계정에는 실험 템플릿에 지정할 수 있는 중지 조건 수에 대한 할당량이 있습니다. 자세한 설명은 [AWS 장애 주입 서비스의 할당량 및 한도](#) 섹션을 참조하세요.

중지 조건 구문

실험 템플릿을 생성할 때는 생성한 CloudWatch 경보를 지정하여 중지 조건을 하나 이상 지정합니다.

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

다음 예는 실험 템플릿에 중지 조건이 지정되어 있지 않음을 나타냅니다.

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

자세히 알아보기

CloudWatch 알람을 만들고 실험 템플릿에 중지 조건을 추가하는 방법을 보여주는 자습서는 [여기](#)를 참조하십시오. [인스턴스에서 CPU 스트레스 실행](#)

AWS FIS에서 지원하는 리소스 유형에 사용할 수 있는 CloudWatch 지표에 대한 자세한 내용은 [다음](#)을 참조하십시오.

- [클라우드워치 사용하여 인스턴스를 모니터링하십시오. CloudWatch](#)
- [아마존 ECS 메트릭 CloudWatch](#)
- [클라우드워치 사용하여 Amazon RDS 메트릭을 모니터링합니다. CloudWatch](#)
- [다음과 같이 사용하여 실행 명령 메트릭을 모니터링합니다. CloudWatch](#)

AWS FIS 실험을 위한 IAM 역할

AWS Identity and Access Management(IAM)은 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어할 수 있도록 지원하는 AWS 서비스입니다. AWS FIS를 사용하려면 AWS FIS가 사용자를 대신하여 실험을 실행할 수 있도록 AWS FIS에 필요한 권한을 부여하는 IAM 역할을 생성해야 합니다. 실험 템플릿을 생성할 때 이 실험 역할을 지정합니다. 단일 계정 실험의 경우 실험 역할에 대한 IAM 정책은 실험 템플릿에서 대상으로 지정한 리소스를 수정할 수 있는 권한을 부여해야 합니다. 다중 계정 실험의 경우 실험 역할은 오케스트레이터 역할에 각 대상 계정에 대한 IAM 역할을 맡을 수 있는 권한을 부여해야 합니다. 자세한 설명은 [다중 계정 실험에 대한 권한](#) 섹션을 참조하세요.

최소 권한 부여 표준 보안 관행을 따르는 것이 좋습니다. 정책에 특정 리소스 ARN 또는 태그를 지정하여 이 작업을 수행할 수 있습니다.

AWS FIS를 빠르게 시작할 수 있도록 실험 역할을 만들 때 지정할 수 있는 AWS 관리형 정책을 제공합니다. 또는 자체 인라인 정책 문서를 만들 때 이러한 정책을 모델로 사용할 수도 있습니다.

내용

- [사전 조건](#)
- [옵션 1: 실험 역할 생성 및 AWS 관리형 정책 연결](#)
- [옵션 2: 실험 역할 생성 및 인라인 정책 문서 추가](#)

사전 조건

시작하기 전에 AWS CLI를 설치하고 필요한 신뢰 정책을 생성합니다.

AWS CLI 설치

시작하기 전에 AWS CLI을(를) 설치하고 구성합니다. AWS CLI를 구성하면 AWS 보안 인증 정보를 입력하라는 메시지가 표시됩니다. 이 절차의 예제에서는 기본 리전도 구성했다고 가정합니다. 그렇지 않을 경우 각 명령에 --region 옵션을 적용합니다. 자세한 내용은 [AWS CLI 설치 또는 업데이트 및 AWS CLI 구성](#)을 참조하세요.

신뢰 관계 정책 만들기

실험 역할에는 AWS FIS 서비스가 역할을 수임할 수 있도록 허용하는 신뢰 관계가 있어야 합니다. fis-role-trust-policy.json라는 텍스트 파일을 만들어 다음 신뢰 관계 정책을 추가합니다.

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

```

    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "fis.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

[혼동된 대리자 문제](#)로부터 자신을 보호하기 위하여 `aws:SourceAccount` 및 `aws:SourceArn` 조건 키를 사용할 것을 권장합니다. 소스 계정은 실험의 소유자이고 소스 ARN은 실험의 ARN입니다. 예를 들어 신뢰 정책에 다음 조건 블록을 추가해야 합니다.

```

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

```

대상 계정 역할을 맡을 수 있는 권한 추가(다중 계정 실험만 해당)

다중 계정 실험의 경우 오케스트레이터 계정이 대상 계정 역할을 맡을 수 있는 권한이 필요합니다. 다음 예제를 수정하여 인라인 정책 문서로 추가하여 대상 계정 역할을 가정할 수 있습니다.

```

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

```

옵션 1: 실험 역할 생성 및 AWS 관리형 정책 연결

AWS FIS의 AWS 관리형 정책 중 하나를 사용하여 빠르게 시작할 수 있습니다.

실험 역할을 만들고 AWS 관리형 정책을 연결하려면

1. 실험에 AWS FIS 작업에 대한 관리형 정책이 있는지 확인하세요. 그렇지 않으면 자체 인라인 정책 문서를 만들어야 합니다. 자세한 설명은 [the section called “AWS 관리형 정책”](#) 섹션을 참조하세요.
2. 다음 [create-role](#) 명령을 사용하여 역할을 만들고 사전 조건에 따라 만든 신뢰 정책을 추가합니다.

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

3. 다음 [attach-role-policy](#) 명령을 사용하여 AWS 관리형 정책을 연결합니다.

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

fis-policy-arn 위치는 다음 중 하나입니다.

- 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

옵션 2: 실험 역할 생성 및 인라인 정책 문서 추가

관리형 정책이 없는 작업이나 특정 실험에 필요한 권한만 포함하는 작업에 이 옵션을 사용하세요.

실험을 만들고 인라인 정책 문서를 추가하려면

1. 다음 [create-role](#) 명령을 사용하여 역할을 만들고 사전 조건에 따라 만든 신뢰 정책을 추가합니다.

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

2. *fis-role-permissions-policy.json*라는 텍스트 파일을 만들고 권한 정책을 추가합니다. 시작점으로 사용할 수 있는 예제를 보려면 다음을 참조하세요.

- 오류 주입 작업 - 다음 정책에서 시작합니다.

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

- Amazon EBS 작업 - 다음 정책에서 시작합니다.

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

- Amazon EC2 조치 — 정책부터 시작하십시오. [AWSFaultInjectionSimulatorEC2Access](#)
- Amazon ECS 조치 — [AWSFaultInjectionSimulatorECSAccess](#) 정책부터 시작하십시오.
- Amazon EKS 조치 — [AWSFaultInjectionSimulatorEKSAccess](#) 정책부터 시작하십시오.

- 네트워크 작업 — [AWSFaultInjectionSimulatorNetworkAccess](#) 정책에서 시작.
 - Amazon RDS 조치 — [AWSFaultInjectionSimulatorRDSAccess](#) 정책에서 시작하십시오.
 - Systems Manager 작업 — [AWSFaultInjectionSimulatorSSMAccess](#) 정책부터 시작하십시오.
3. 다음 [put-role-policy](#) 명령을 사용하여 이전 단계에서 만든 권한 정책을 추가합니다.

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

실험 옵션

실험 옵션은 실험에 대한 선택적 설정입니다. 실험 템플릿에서 특정 실험 옵션을 정의할 수 있습니다. 실험을 시작하면 추가 실험 옵션이 설정됩니다.

다음은 실험 템플릿에 정의하는 실험 옵션의 구문입니다.

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

실험 템플릿을 만들 때 실험 옵션을 지정하지 않으면 각 옵션의 기본값이 사용됩니다.

다음은 실험을 시작할 때 설정하는 실험 옵션의 구문입니다.

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

실험을 시작할 때 실험 옵션을 지정하지 않으면 run-all 기본값이 사용됩니다.

내용

- [계정 타겟팅](#)
- [빈 대상 확인 모드](#)
- [액션 모드](#)

계정 타겟팅

실험에서 리소스를 대상으로 지정하려는 AWS 계정이 여러 개 있는 경우 계정 타겟팅 실험 옵션을 사용하여 다중 계정 실험을 정의할 수 있습니다. 여러 대상 계정의 리소스에 영향을 주는 오케스트레이터 계정에서 다중 계정 실험을 실행합니다. Orchestrator 계정은 실험 템플릿과 AWS FIS 실험을 소유합니다. 대상 계정은 AWS FIS 실험의 영향을 받을 수 있는 리소스가 있는 개별 AWS 계정입니다. 자세한 정보는 [에 대한 다중 계정 실험 AWS FIS](#)을 참조하세요.

계정 타겟팅을 사용하여 대상 리소스의 위치를 표시합니다. 계정 타겟팅에 두 가지 값을 제공할 수 있습니다:

- 단일 계정 - 기본값입니다. 실험은 실험이 실행되는 AWS AWS FIS 계정의 리소스만 대상으로 합니다.
- 다중 계정 - 실험이 여러 AWS 계정의 리소스를 대상으로 할 수 있습니다.

대상 계정 구성

다중 계정 실험을 실행하려면 하나 이상의 대상 계정 구성을 정의해야 합니다. 대상 계정 구성은 실험의 대상이 되는 리소스가 있는 각 계정에 대한 accountId, roleArn 및 설명을 지정합니다. 실험 템플릿에 대한 대상 계정 구성의 계정 ID는 고유해야 합니다.

다중 계정 실험 템플릿을 만들면 실험 템플릿은 실험 템플릿에 대한 모든 대상 계정 구성의 개수인 읽기 전용 필드인 targetAccountConfigurationsCount를 반환합니다.

다음은 대상 계정 구성에 대한 구문입니다.

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

대상 계정 구성을 만들 때 다음을 입력합니다.

accountId

대상 계정의 12자리 AWS 계정 ID입니다.

roleArn

대상 계정에서 작업을 수행할 AWS FIS 권한을 부여하는 IAM 역할.

description

설명(선택 사항)입니다.

대상 계정 구성으로 작업하는 방법에 대해 자세히 알아보려면 [the section called “다중 계정 실험 작업” 단원을 참조하세요.](#)

빈 대상 확인 모드

이 모드에서는 대상 리소스가 확인되지 않은 경우에도 실험을 완료할 수 있는 옵션을 제공합니다.

- fail - 기본값입니다. 대상에 확인된 리소스가 없는 경우 실험이 failed 상태로 즉시 종료됩니다.
- skip - 대상에 확인된 리소스가 없는 경우 실험이 계속되고 확인된 대상이 없는 모든 작업이 건너뛰어집니다. ARN과 같은 고유 식별자를 사용하여 정의된 대상이 있는 작업은 건너뛸 수 없습니다. 고유 식별자를 사용하여 정의된 대상을 찾을 수 없는 경우 실험은 failed 상태로 즉시 종료됩니다.

액션 모드

액션 모드는 실험을 시작할 때 지정할 수 있는 선택적 매개변수입니다. 대상 리소스에 결함을 주입하기 전에 대상 미리 보기를 skip-all 생성하도록 작업 모드를 설정할 수 있습니다. 대상 미리보기를 통해 다음을 확인할 수 있습니다.

- 예상한 리소스를 타겟팅하도록 실험 템플릿을 구성했는지 확인하세요. 리소스가 임의로 제거, 업데이트 또는 샘플링될 수 있으므로 이 실험을 시작할 때 목표로 하는 실제 리소스는 미리 보기와 다를 수 있습니다.
- 로깅 구성이 올바르게 설정되었는지 확인하십시오.
- 다중 계정 실험의 경우 각 대상 계정 구성에 대해 IAM 역할을 올바르게 설정했다는 의미입니다.

Note

이 skip-all 모드에서는 AWS FIS 실험을 실행하고 리소스에 대한 조치를 취하는 데 필요한 권한이 있는지 확인할 수 없습니다.

actions mode 매개변수는 다음 값을 허용합니다.

- run-all- (기본값) 실험은 대상 리소스에 대해 조치를 취합니다.

- skip-all- 실험은 대상 리소스에 대한 모든 작업을 건너뛰게 됩니다.

실험을 시작할 때 액션 모드 파라미터를 설정하는 방법에 대한 자세한 내용은 [을 참조하십시오](#) [실험 템플릿에서 목표 미리 보기를 생성합니다..](#)

FIS 실험 템플릿으로 작업하세요. AWS

AWS FIS 콘솔 또는 명령줄을 사용하여 실험 템플릿을 만들고 관리할 수 있습니다. 실험 템플릿을 만든 후 이를 사용하여 실험을 실행할 수 있습니다.

Tasks

- [실험 템플릿 만들기](#)
- [실험 템플릿 보기](#)
- [실험 템플릿에서 목표 미리 보기를 생성합니다.](#)
- [템플릿에서 실험 시작](#)
- [실험 템플릿 업데이트](#)
- [실험 템플릿에 태그 지정](#)
- [실험 템플릿 만들기](#)

실험 템플릿 만들기

시작하기 전에 다음 작업을 완료하세요.

- [실험 계획 세우기.](#)
- AWS FIS 서비스에 사용자 대신 작업을 수행할 권한을 부여하는 IAM 역할을 생성하십시오. 자세한 정보는 [AWS FIS 실험을 위한 IAM 역할](#)을 참조하세요.
- FIS에 액세스할 수 있는지 확인하세요. AWS 자세한 내용을 알아보려면 [AWS FIS 정책 예제](#)를 참조하세요.

콘솔을 사용하여 실험 템플릿을 생성하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿 생성을 선택합니다.

4. (선택 사항) 계정 타겟팅의 경우 여러 계정을 선택하여 다중 계정 실험 템플릿을 구성합니다.
5. 계정 타겟팅에 대해 확인을 선택합니다.
6. 이름 및 설명에 템플릿에 대한 이름과 설명을 입력합니다.
7. 작업 경우 템플릿의 작업 세트를 지정합니다. 각 작업에 대해 작업 추가를 선택하고 다음을 완료합니다.

- 이름에서 작업의 이름을 입력합니다.

허용되는 문자는 영숫자, 하이픈(-), 밑줄(_)입니다. 이름은 문자로 시작해야 합니다. 공백은 사용할 수 없습니다. 각 작업 이름은 이 템플릿에서 고유해야 합니다.

- (선택 사항) 설명에, 작업에 대한 설명을 입력합니다. 최대 길이는 512자입니다.
 - (선택 사항) 시작하기 전에서, 현재 작업이 시작되기 전에 완료되어야 하는 이 템플릿에 정의된 다른 작업을 선택합니다. 그렇지 않으면 실험이 시작될 때 이 작업이 실행됩니다.
 - 작업 유형에서 AWS FIS 작업을 선택합니다.
 - 대상의 경우 대상 섹션에서 정의한 대상을 선택합니다. 이 작업의 대상을 아직 정의하지 않은 경우 AWS FIS가 새 대상을 생성합니다.
 - 작업 파라미터에는 작업의 파라미터를 지정합니다. 이 AWS 섹션은 FIS 작업에 매개 변수가 있는 경우에만 나타납니다.
 - 저장을 선택합니다.
8. 대상의 경우 작업을 수행할 대상 리소스를 정의합니다. 대상으로 하나 이상의 리소스 ID 또는 하나의 리소스 태그를 지정해야 합니다. 편집을 선택하여 AWS FIS가 이전 단계에서 생성한 대상을 편집하거나 대상 추가를 선택합니다. 각 대상에서 다음을 수행합니다.

- 이름에 대상의 이름을 입력합니다.

허용되는 문자는 영숫자, 하이픈(-), 밑줄(_)입니다. 이름은 문자로 시작해야 합니다. 공백은 사용할 수 없습니다. 각 대상 이름은 이 템플릿에서 고유해야 합니다.

- 리소스 유형에서는 작업에 지원되는 리소스 유형을 선택합니다.
- 대상 메서드에서 다음 중 하나를 수행합니다.
 - 리소스 ID를 선택한 다음 리소스 ID를 선택하거나 추가합니다.
 - 리소스 태그, 필터, 파라미터를 선택한 다음 필요한 태그와 필터를 추가합니다. 자세한 정보는 [the section called “대상 리소스 식별”](#)을 참조하세요.
- 선택 모드에서는 개수를 선택하여 지정된 수의 식별된 대상에 대해 작업을 실행하거나 백분율을 선택하여 지정된 비율의 식별된 대상에 대해 작업을 실행합니다. 기본적으로 작업은 식별된 모든 대상에서 실행됩니다.

- 저장을 선택합니다.
9. 생성한 대상으로 작업을 업데이트하려면 작업 아래에서 작업을 찾아 편집을 선택한 다음 대상을 업데이트하세요. 실험 템플릿에서 동일한 대상을 여러 작업에 사용할 수 있습니다.
 10. (다중 계정 실험만 해당) 대상 계정 구성의 경우 각 대상 계정에 대해 역할 ARN과 선택적 설명을 추가합니다. 대상 계정 역할 ARN을 CSV 파일로 업로드하려면 모든 대상 계정에 대한 역할 ARN 업로드를 선택한 다음 .CSV 파일 선택을 선택합니다.
 11. 서비스 액세스의 경우 기존 IAM 역할 사용을 선택한 다음 이 자습서의 사전 조건에 설명된 대로 생성한 IAM 역할을 선택합니다. 역할이 표시되지 않는 경우 해당 역할에 필요한 신뢰 관계가 있는지 확인하세요. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.
 12. (선택 사항) 중지 조건의 경우 중지 조건에 대한 Amazon CloudWatch 경보를 선택합니다. 자세한 정보는 [AWS FIS의 중지 조건](#)을 참조하세요.
 13. (선택 사항) 로그의 경우 대상 옵션을 구성합니다. S3 버킷으로 로그를 보내려면 Amazon S3 버킷으로 전송을 선택하고 버킷 이름과 접두사를 입력합니다. 로그를 Logs로 CloudWatch 보내려면 Send to CloudWatch Logs를 선택하고 로그 그룹을 입력합니다.
 14. (선택 사항) 태그의 경우 새 태그 추가를 선택하고 태그 키와 태그 값을 지정합니다. 추가한 태그는 템플릿을 사용하여 실행되는 실험이 아니라 실험 템플릿에 적용됩니다.
 15. 실험 템플릿 생성을 선택합니다. 확인 메시지가 나타나면 **create**을 입력하고 실험 템플릿 생성을 선택합니다.

CLI를 사용하여 실험 템플릿을 만들려면

[create-experiment-template](#) 명령을 사용합니다.

JSON 파일에서 실험 템플릿을 로드할 수 있습니다.

--cli-input-json 파라미터를 사용합니다.

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

자세한 내용은 AWS Command Line Interface 사용 설명서의 [CLI 스텀레톤 템플릿 생성](#)을 참조하세요. 템플릿 예시를 보려면 [AWS FIS 실험 템플릿 예시](#) 단원을 참조하세요.

실험 템플릿 보기

생성한 실험 템플릿을 볼 수 있습니다.

콘솔을 사용하여 실험 템플릿을 보려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 특정 템플릿에 대한 정보를 보려면 실험 템플릿 ID를 선택합니다.
4. 세부 정보 섹션에서 템플릿의 설명 및 중지 조건을 볼 수 있습니다.
5. 실험 템플릿의 작업을 보려면 작업을 선택합니다.
6. 실험 템플릿의 대상을 보려면 대상을 선택합니다.
7. 실험 템플릿의 태그를 보려면 태그를 선택합니다.

CLI를 사용하여 실험 템플릿을 보려면

[list-experiment-templates](#) 명령을 사용하여 실험 템플릿 목록을 가져오고 [get-experiment-template](#) 명령을 사용하여 특정 실험 템플릿에 대한 정보를 가져올 수 있습니다.

실험 템플릿에서 목표 미리 보기를 생성합니다.

실험을 시작하기 전에 대상 미리보기를 생성하여 실험 템플릿이 예상 리소스를 타겟팅하도록 구성되어 있는지 확인할 수 있습니다. 리소스가 임의로 제거, 업데이트 또는 샘플링될 수 있으므로 실제 실험을 시작할 때 타겟팅되는 리소스는 미리 보기에 있는 리소스와 다를 수 있습니다. 대상 미리보기를 생성하면 모든 작업을 건너뛰는 실험이 시작됩니다.

Note

대상 미리보기를 생성해도 리소스에 대한 조치를 취하는 데 필요한 권한이 있는지 확인할 수 없습니다.

콘솔을 사용하여 대상 미리 보기를 시작하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿의 대상을 보려면 대상을 선택합니다.
4. 실험 템플릿의 대상 리소스를 확인하려면 미리 보기 생성을 선택합니다. 실험을 실행하면 이 목표 미리보기가 가장 최근 실험의 대상으로 자동 업데이트됩니다.

CLI를 사용하여 대상 미리 보기를 시작하려면

- 다음 [실험 시작](#) 명령을 실행합니다. 기울임꼴로 표시된 값을 원하는 값으로 바꿉니다.

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

템플릿에서 실험 시작

실험 템플릿을 만든 후 해당 템플릿을 사용하여 실험을 시작할 수 있습니다.

실험을 시작하면 지정된 템플릿의 스냅샷을 만들고 이 스냅샷을 사용하여 실험을 실행합니다. 따라서 실험이 실행되는 동안 실험 템플릿이 업데이트되거나 삭제되더라도 해당 변경 사항은 진행 중인 실험에 영향을 주지 않습니다.

실험을 시작하면 AWS FIS가 사용자를 대신하여 서비스 연결 역할을 생성합니다. 자세한 정보는 [장애 인젝션 서비스에 서비스 연결 역할을 사용하십시오](#). AWS를 참조하세요.

실험을 시작한 후에는 언제든지 중지할 수 있습니다. 자세한 정보는 [실험 중지](#)를 참조하세요.

콘솔을 사용하여 실험을 시작하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. (선택 사항) 목표 확인을 위한 미리보기를 생성하려면:
 - 타겟을 선택합니다.
 - 미리 보기 생성을 선택합니다.
4. 실험 템플릿을 선택하고 실험 시작을 선택합니다.
5. (선택 사항) 실험에 태그를 추가하려면 새 태그 추가를 선택하고 태그 키와 태그 값을 입력합니다.
6. 실험 시작을 선택합니다. 확인 메시지가 나타나면 **start**을 입력하고 실험 시작을 선택합니다.

CLI를 사용하여 실험을 시작하려면

[start-experiment](#) 명령을 사용합니다.

실험 템플릿 업데이트

기존 실험 템플릿을 업데이트할 수 있습니다. 실험 템플릿을 업데이트해도 해당 템플릿을 사용하는 실행 중인 실험에는 변경 내용이 영향을 주지 않습니다.

콘솔을 사용하여 실험 템플릿을 업데이트하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 업데이트를 선택합니다.
4. 필요에 따라 템플릿 세부 정보를 수정하고 실험 템플릿 업데이트를 선택합니다.

CLI를 사용하여 실험 템플릿을 업데이트하려면

[update-experiment-template](#) 명령을 사용합니다.

실험 템플릿에 태그 지정

실험에 태그를 적용하여 실험 템플릿을 체계적으로 구성할 수 있습니다. 또한 [태그 기반 IAM 정책](#)을 구현하여 실험 템플릿에 대한 액세스를 제어할 수 있습니다.

콘솔을 사용하여 실험 템플릿에 태그를 지정하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 태그 관리를 선택합니다.
4. 새 태그를 추가하려면 새 태그 추가를 선택하고 키와 값을 지정합니다.

태그를 제거하려면 해당 태그에서 제거를 선택합니다.

5. 저장을 선택합니다.

CLI를 사용하여 실험 템플릿에 태그를 지정하려면

[tag-resource](#) 명령을 사용합니다.

실험 템플릿 만들기

실험 템플릿이 더 이상 필요하지 않으면 삭제할 수 있습니다. 실험 템플릿을 삭제해도 해당 템플릿을 사용하는 실행 중인 실험은 변경 내용의 영향을 받지 않습니다. 실험은 완료되거나 중지될 때까지 계속 실행됩니다. 하지만 삭제된 실험 템플릿은 콘솔의 실험 페이지에서 볼 수 없습니다.

콘솔을 사용하여 실험 템플릿을 삭제하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 삭제를 선택합니다.
4. 확인 메시지가 나타나면 **delete**를 입력하고 실험 템플릿 삭제를 선택합니다.

CLI를 사용하여 실험 템플릿을 삭제하려면

[delete-experiment-template](#) 명령을 사용합니다.

AWS FIS 실험 템플릿 예시

AWSFIS API 또는 명령줄 도구를 사용하여 실험 템플릿을 만드는 경우 JavaScript 개체 표기법 (JSON) 으로 템플릿을 구성할 수 있습니다. 실험 템플릿의 구성 요소에 대한 자세한 내용은 [템플릿 구성 요소](#) 단원을 참조하세요.

예제 템플릿 중 하나를 사용하여 실험을 만들려면 JSON 파일 (예:my-template.json) 에 저장하고 ##### 표시된 자리 표시자 값을 원하는 값으로 바꾼 후 다음 명령을 실행합니다. [create-experiment-template](#)

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

템플릿 예제

- [필터를 기반으로 EC2 인스턴스 중지](#)
- [지정된 수의 EC2 인스턴스 중지](#)
- [사전 구성된 AWS FIS SSM 문서 실행](#)
- [사전 정의된 자동화 런북 실행](#)
- [대상 IAM 역할을 사용하여 EC2 인스턴스에서의 API 작업을 제한하세요.](#)
- [Kubernetes 클러스터 내 포드의 CPU 스트레스 테스트](#)

필터를 기반으로 EC2 인스턴스 중지

다음 예제에서는 지정된 리전의 모든 실행 중인 VPC에서 지정된 태그를 가진 모든 Amazon EC2 인스턴스를 중지합니다. 2분 후에 인스턴스가 다시 시작됩니다.

```
{
  "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"
}

```

지정된 수의 EC2 인스턴스 중지

다음 예제에서는 지정된 태그를 가진 세 개의 인스턴스를 중지합니다. AWS FIS는 중지할 특정 인스턴스를 임의로 선택합니다. 2분 후에 이러한 인스턴스를 다시 시작합니다.

```
{
  "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"
}
```

사전 구성된 AWS FIS SSM 문서 실행

[다음 예제에서는 사전 구성된 AWS FIS SSM 문서인 -run-cpu-stress를 사용하여 지정된 EC2 인스턴스에서 60초 동안 CPU 장애 삽입을 실행합니다. AWSFIS](#) AWS FIS는 2분 동안 실험을 모니터링합니다.

```
{
  "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"
}

```

사전 정의된 자동화 런북 실행

다음 예제는 Systems Manager에서 제공하는 런북인 [AWS-PublishSNSNotification](#)을 사용하여 Amazon SNS에 알림을 게시합니다. 역할에는 지정된 SNS 주제에 알림을 게시할 권한이 있어야 합니다.

```
{
```

```

"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"
}

```

대상 IAM 역할을 사용하여 EC2 인스턴스에서의 API 작업을 제한하세요.

다음 예시에서는 지정된 IAM 역할을 사용하는 EC2 인스턴스에서 지정된 API 작업에 대한 호출의 비율을 100% 제한합니다.

```

{
  "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"
}

```

Kubernetes 클러스터 내 포드의 CPU 스트레스 테스트

다음 예시에서는 Chaos Mesh를 사용하여 Amazon EKS Kubernetes 클러스터의 포드 CPU에 1분 동안 스트레스 테스트를 실시합니다.

```

{
  "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": {}
}

```

다음 예시에서는 Litmus를 사용하여 Amazon EKS Kubernetes 클러스터의 포드 CPU에 1분 동안 스트레스 테스트를 실시합니다.

```

{
  "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": "{\n  \"engineState\": \"active\",\n  \"appinfo\": {\n    \"appns\": \"default\",\n    \"applabel\": \"run=nginx\",\n    \"appkind\": \"deployment\",\n    \"chaosServiceAccount\": \"litmus-admin\",\n    \"experiments\": [\n      {\n        \"name\": \"pod-cpu-hog\",\n        \"spec\": {\n          \"components\": {\n            \"env\": [\n              {\n                \"name\": \"TOTAL_CHAOS_DURATION\",\n                \"value\": \"60\"\n              },\n              {\n                \"name\": \"CPU_CORES\",\n                \"value\": \"1\"\n              },\n              {\n                \"name\": \"PODS_AFFECTED_PERC\",\n                \"value\": \"100\"\n              },\n              {\n                \"name\": \"CONTAINER_RUNTIME\",\n                \"value\": \"docker\"\n              },\n              {\n                \"name\": \"SOCKET_PATH\",\n                \"value\": \"/var/run/docker.sock\"\n              }\n            ],\n            \"probe\": []\n          }\n        }\n      }\n    ],\n    \"annotationCheck\": \"false\"\n  },\n  \"targets\": {\n    \"Cluster\": \"MyCluster\"\n  }\n}\n},\n  \"stopConditions\": [\n    {\n      \"source\": \"none\"\n    }\n  ],\n  \"roleArn\": \"arn:aws:iam::111122223333:role/role-name\",\n  \"tags\": {}\n}
```

에 대한 다중 계정 실험 AWS FIS

다중 계정 실험을 통해 지역 내 여러 AWS 계정을 포괄하는 애플리케이션에서 실제 장애 시나리오를 설정하고 실행할 수 있습니다. 여러 대상 계정의 리소스에 영향을 미치는 오케스트레이터 계정에서 다중 계정 실험을 실행합니다.

다중 계정 실험을 실행하면 영향을 받는 리소스가 있는 대상 계정은 AWS Health Dashboard를 통해 알림을 받게 되며, 대상 계정의 사용자에게 알림을 제공합니다. 다중 계정 실험을 통해 다음을 수행할 수 있습니다:

- 제공되는 중앙 제어 및 가이드를 사용하여 여러 계정에 걸쳐 있는 애플리케이션에서 실제 장애 시나리오를 실행하세요. AWS FIS
- 각 대상의 범위를 정의하는 세분화된 권한과 태그가 있는 IAM 역할을 사용하여 다중 계정 실험의 효과를 제어하세요.
- 로그부터 로그까지 각 계정에서 AWS FIS 수행한 작업을 중앙에서 볼 수 AWS Management Console 있습니다. AWS FIS
- AWS의 각 계정에서 AWS FIS 이루어진 API 호출을 모니터링하고 CloudTrail 감사하십시오.

이 섹션은 다중 계정 실험을 시작하는 데 도움이 됩니다.

주제

- [다중 계정 실험에 관한 개념](#)
- [다중 계정 실험을 위한 사전 조건](#)
- [다중 계정 실험 작업](#)

다중 계정 실험에 관한 개념

다음은 다중 계정 실험의 핵심 개념입니다.

오케스트레이터 계정

Orchestrator 계정은 AWS FIS 콘솔에서 실험을 구성 및 관리하고 로깅을 중앙 집중화하는 중앙 계정 역할을 합니다. Orchestrator 계정은 실험 템플릿과 실험을 소유합니다. AWS FIS

대상 계정

대상 계정은 AWS FIS 다중 계정 실험의 영향을 받을 수 있는 리소스가 있는 개별 AWS 계정입니다.

대상 계정 구성

실험 템플릿에 대상 계정 구성을 추가하여 실험의 일부가 되는 대상 계정을 정의합니다. 대상 계정 구성은 다중 계정 실험에 필요한 실험 템플릿의 요소입니다. 계정 ID, IAM 역할 및 설명 (선택 사항) 을 설정하여 각 대상 AWS 계정에 대해 하나를 정의합니다.

다중 계정 실험을 위한 사전 조건

다중 계정 실험에 중지 조건을 사용하려면 먼저 교차 계정 경보를 구성해야 합니다. IAM 역할은 다중 계정 실험 템플릿을 만들 때 정의됩니다. 템플릿을 생성하기 전에 필요한 IAM 역할을 생성할 수 있습니다.

내용

- [다중 계정 실험에 대한 권한](#)
- [다중 계정 실험의 중지 조건\(선택 사항\)](#)

다중 계정 실험에 대한 권한

멀티-계정 실험은 IAM 역할 체인을 사용하여 대상 계정의 리소스에 대한 작업을 수행할 수 있는 권한을 AWS FIS 에 부여합니다. 다중 계정 실험의 경우, 각 대상 계정과 오케스트레이터 계정에서 IAM 역할을 설정합니다. 이러한 IAM 역할을 사용하려면 대상 계정과 오케스트레이터 계정 및 오케스트레이터 계정과 AWS FIS 간의 신뢰 관계가 필요합니다.

대상 계정의 IAM 역할에는 리소스에 대한 작업을 수행하는 데 필요한 권한이 포함되어 있으며, 대상 계정 구성을 추가하여 실험 템플릿에 대해 생성됩니다. 대상 계정의 역할을 맡을 수 있는 권한이 있는 오케스트레이터 계정에 대한 IAM 역할을 생성하고 AWS FIS와 신뢰 관계를 설정합니다. 이 IAM 역할은 실험 템플릿의 roleArn으로 사용됩니다.

역할 체인에 대해 자세히 알아보려면 [IAM 사용 설명서에서 역할 용어 및 개념](#)을 참조하세요.

다음 예에서는 오케스트레이터 계정 A가 대상 계정 B에서 `aws:ebs:pause-volume-io`로 실험을 실행할 수 있도록 권한을 설정합니다.

1. 계정 B에서 작업을 실행하는 데 필요한 권한이 있는 IAM 역할을 만듭니다. 각 작업에 필요한 권한은 [the section called “작업 참조”](#)를 참조하세요. 다음 예는 EBS 볼륨 IO 일시 중지 작업 [the section called “aws:ebs:pause-volume-io”](#)를 실행하기 위해 대상 계정에서 부여하는 권한을 보여줍니다.

```
{
  "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. 다음으로 계정 A와 신뢰 관계를 만드는 신뢰 정책을 계정 B에 추가합니다. 3단계에서 만들 계정 A에 대한 IAM 역할의 이름을 선택합니다.

```
{
  "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. 계정 A에서 IAM 역할을 생성합니다. 이 역할 이름은 2단계의 신뢰 정책에서 지정한 역할과 일치해야 합니다. 여러 계정을 타겟팅하려면 오케스트레이터에게 각 역할을 맡을 수 있는 권한을 부여합니다. 다음 예는 계정 A가 계정 B를 가정할 수 있는 권한을 보여 줍니다. 대상 계정이 더 있는 경우 이 정책에 역할 ARN을 추가합니다. 대상 계정당 하나의 역할 ARN만 가질 수 있습니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": [
        "arn:aws:iam::accountIdB:role/role_name"
      ]
    }
  ]
}

```

4. 계정 A에 대한 이 IAM 역할은 실험 템플릿의 `roleArn`으로 사용됩니다. 다음 예는 오케스트레이터 계정인 계정 A를 수입할 AWS FIS 권한을 부여하는 IAM 역할에 필요한 신뢰 정책을 보여줍니다.

```

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

```

```

    "Action": "sts:AssumeRole",
  }
]
}

```

Stacksets를 사용하여 한 번에 여러 IAM 역할을 프로비저닝할 수도 있습니다. 사용하려면 CloudFormation StackSets 계정에 필요한 StackSet 권한을 설정해야 합니다. AWS 자세히 알아보려면 [AWS 사용](#)을 참조하십시오 CloudFormation StackSets.

다중 계정 실험의 중지 조건(선택 사항)

중지 조건은 경보로 정의한 임계값에 도달할 경우 실험을 중지하는 메커니즘입니다. 다중 계정 실험에 대한 중지 조건을 설정하려면 교차 계정 경보를 사용할 수 있습니다. 오케스트레이터 계정에서 읽기 전용 권한을 사용하여 경보를 사용할 수 있도록 하려면 각 대상 계정에서 공유를 활성화해야 합니다. 공유가 완료되면 지표 수식을 사용하여 서로 다른 대상 계정의 지표를 결합할 수 있습니다. 그런 다음 이 경보를 실험의 중지 조건으로 추가할 수 있습니다.

교차 계정 대시보드에 대해 자세히 알아보려면 [에서 교차 계정 기능 활성화를 참조하십시오.](#)
CloudWatch

다중 계정 실험 작업

AWS FIS 콘솔 또는 명령줄을 사용하여 다중 계정 실험 템플릿을 만들고 관리할 수 있습니다. 계정 대상 실험 옵션을 "multi-account"으로 지정하고 대상 계정 구성을 추가하여 다중 계정 실험을 만듭니다. 실험 템플릿을 만든 후 이를 사용하여 다중 계정 실험을 실행할 수 있습니다.

내용

- [다중 계정 실험에 관한 모범 사례](#)
- [다중 계정 실험 템플릿 생성](#)
- [대상 계정 구성 업데이트](#)
- [대상 계정 구성 삭제](#)

다중 계정 실험에 관한 모범 사례

다중 계정 실험에 관한 모범 사례는 다음과 같습니다.

- 다중 계정 실험의 대상을 구성할 때는 모든 대상 계정에서 일관된 리소스 태그를 사용하여 타겟팅하는 것이 좋습니다. AWS FIS 실험을 통해 각 대상 계정에서 일관된 태그가 있는 리소스를 확인할 수 있습니다. `emptyTargetResolutionMode`가 `skip`로 설정된 실험을 제외하고, 작업은 모든 대상 계정에서 하나 이상의 대상 리소스를 확인해야 하며 그렇지 않으면 실패합니다. 작업 할당량은 계정 별로 적용됩니다. 리소스 ARN을 기준으로 리소스를 타겟팅하려는 경우에는 작업당 동일한 단일 계정 제한이 적용됩니다.
- 파라미터나 필터를 사용하여 하나 이상의 가용성 영역에서 리소스를 타겟팅하는 경우에는 AZ 이름이 아닌 AZ ID를 지정해야 합니다. AZ ID는 계정 전체의 가용 영역에 대한 고유하고 일관된 식별자입니다. 계정의 가용 영역에 대한 AZ ID를 찾는 방법을 알아보려면 [AWS 리소스에 대한 가용 영역 ID](#)를 참조하세요.

다중 계정 실험 템플릿 생성

를 통해 실험 템플릿을 만드는 방법을 배우려면 AWS Management Console

[실험 템플릿 만들기](#) 섹션을 참조하십시오.

CLI를 사용하여 실험 템플릿을 만들려면

1. 열기 AWS Command Line Interface
2. 계정 타겟팅 실험 옵션을 "multi-account" (예:my-template.json) 로 설정한 상태로 저장된 JSON 파일에서 실험을 만들려면 ##### 표시된 자리 표시자 값을 원하는 값으로 바꾸고 다음 명령어를 실행합니다. [create-experiment-template](#)

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

그러면 응답에 실험 템플릿이 반환됩니다. 응답에서 실험 템플릿의 ID인 `id`를 복사합니다.

3. [create-target-account-configuration](#) 명령을 실행하여 대상 계정 구성을 실험 템플릿에 추가합니다. #####로 표시된 자리 표시자 값을 사용자 고유의 값으로 바꾸고, 2단계의 `id`를 `--experiment-template-id` 파라미터의 값으로 사용한 후 다음을 실행합니다. `--description` 파라미터는 선택 항목입니다. 각 대상 계정에 대해 이 단계를 반복합니다.

```
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. [get-target-account-configuration](#) 명령을 실행하여 특정 대상 계정 구성의 세부 정보를 검색합니다.


```
aws fis get-target-account-configuration --experiment-template-id EXTxxxxxxxxx --
account-id 111122223333
```

5. 대상 계정 구성을 모두 추가한 후에는 [list-target-account-configurations](#) 명령 명령을 실행하여 대상 계정 구성이 생성되었는지 확인할 수 있습니다.

```
aws fis list-target-account-configurations --experiment-template-id EXTxxxxxxxxx
```

또한 [get-experiment-template](#) 명령을 실행하여 대상 계정 구성을 추가했는지 확인할 수 있습니다. 이 템플릿은 실험 템플릿에 있는 모든 대상 계정 구성의 개수를 나타내는 읽기 전용 필드 `targetAccountConfigurationsCount`를 반환합니다.

6. 준비가 되면 [start-experiment](#) 명령을 사용하여 실험 템플릿을 실행할 수 있습니다.

```
aws fis start-experiment --experiment-template-id EXTxxxxxxxxx
```

대상 계정 구성 업데이트

계정에 대한 역할 ARN 또는 설명을 변경하려는 경우 기존 대상 계정 구성을 업데이트할 수 있습니다. 대상 계정 구성을 업데이트하면 템플릿을 사용하는 실행 중인 실험에는 변경 사항이 영향을 미치지 않습니다.

를 사용하여 대상 계정 구성을 업데이트하려면 AWS Management Console

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 업데이트를 선택합니다.
4. 대상 계정 구성을 수정하고 실험 템플릿 업데이트를 선택합니다.

CLI를 사용하여 대상 계정 구성을 업데이트하려면 다음을 수행합니다.

[update-target-account-configuration](#) 명령을 실행하여 ##### 표시된 자리 표시자 값을 원하는 값으로 바꿉니다. `--role-arn` 및 `--description` 파라미터는 선택 사항이며, 포함하지 않으면 업데이트되지 않습니다.

```
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"
```

대상 계정 구성 삭제

대상 계정 구성이 더 이상 필요하지 않은 경우 삭제할 수 있습니다. 대상 계정 구성을 삭제해도 템플릿을 사용하여 실행 중인 모든 실험은 영향을 받지 않습니다. 실험은 완료되거나 중지될 때까지 계속 실행됩니다.

를 사용하여 대상 계정 구성을 삭제하려면 AWS Management Console

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 업데이트를 선택합니다.
4. 대상 계정 구성에서 삭제하려는 대상 계정 역할 ARN에 대해 제거를 선택합니다.

CLI를 사용하여 대상 계정 구성을 삭제하려면 다음을 수행합니다.

[delete-target-account-configuration](#) 명령을 실행하여 ##### 표시된 자리 표시자 값을 원하는 값으로 바꿉니다.

```
aws fis update-target-account-configuration --experiment-template-id EXTxxxxxxxxx --
account-id 111122223333
```

AWS FIS 시나리오 라이브러리

시나리오는 애플리케이션이 실행되는 컴퓨팅 리소스의 중단과 같이 고객이 애플리케이션의 복원력을 테스트하기 위해 적용할 수 있는 이벤트 또는 조건을 정의합니다. 시나리오는 AWS에서 생성하고 소유하며, 일반적인 애플리케이션 장애에 대해 사전 정의된 대상 그룹과 장애 조치(예: 자동 크기 조정 그룹 내 인스턴스 30% 중지)를 제공하여 차별화되지 않은 작업의 부담을 최소화합니다.

주제

- [AWS FIS 시나리오 작업](#)
- [AWS FIS 시나리오 라이브러리의 시나리오](#)
- [AZ Availability: Power Interruption](#)
- [Cross-Region: Connectivity](#)

AWS FIS 시나리오 작업

시나리오는 콘솔 전용 시나리오 라이브러리를 통해 제공되며 AWS FIS 실험 템플릿을 사용하여 실행합니다. 시나리오를 사용하여 실험을 실행하려면 라이브러리에서 시나리오를 선택하고 워크로드 세부 정보와 일치하는 파라미터를 지정한 다음 계정에 실험 템플릿으로 저장합니다.

주제

- [시나리오 보기](#)
- [시나리오 사용](#)
- [시나리오 내보내기](#)

시나리오 보기

콘솔을 사용하여 시나리오 보기

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 시나리오 라이브러리를 선택합니다.
3. 특정 시나리오에 대한 정보를 보려면 시나리오 카드를 선택하여 분할 패널을 불러옵니다.
 - 페이지 하단의 분할 패널에 있는 설명 탭에서 시나리오에 대한 간략한 설명을 볼 수 있습니다. 또한 필요한 대상 리소스의 요약과 시나리오에 사용할 리소스를 준비하기 위해 취해야 하는 조치가 포함된 사전 요구 사항에 대한 간략한 요약을 찾을 수 있습니다. 마지막으로 시나리오의 대상 및

작업에 대한 추가 정보뿐만 아니라 기본 설정으로 실험이 성공적으로 실행될 때의 예상 기간도 확인할 수 있습니다.

- 페이지 하단의 분할 패널에 있는 콘텐츠 탭에서 시나리오에서 생성될 실험 템플릿의 일부가 채워진 버전을 미리 볼 수 있습니다.
- 페이지 하단의 분할 패널에 있는 세부 정보 탭에서 시나리오 구현 방법에 대한 자세한 설명을 찾을 수 있습니다. 여기에는 시나리오의 개별 측면을 근사화하는 방법에 대한 자세한 정보가 포함될 수 있습니다. 해당하는 경우, 어떤 지표를 중지 조건으로 사용하고 관찰성을 제공하여 실험에서 배울 수 있는지에 대해서도 읽어볼 수 있습니다. 마지막으로 결과 실험 템플릿을 확장하는 방법에 대한 권장 사항을 찾을 수 있습니다.

시나리오 사용

콘솔을 사용하여 시나리오 사용:

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 시나리오 라이브러리를 선택합니다.
3. 특정 시나리오에 대한 정보를 보려면 시나리오 카드를 선택하여 분할 패널을 불러옵니다.
4. 시나리오를 사용하려면 시나리오 카드를 선택하고 시나리오로 템플릿 생성을 선택합니다.
5. 실험 템플릿 생성 보기에서 누락된 항목을 모두 채웁니다.
 - a. 일부 시나리오에서는 여러 작업 또는 대상 간에 공유되는 파라미터를 일괄 편집할 수 있습니다. 대량 파라미터 편집에 의한 변경을 포함하여 시나리오를 변경하면 이 기능이 비활성화됩니다. 이 기능을 사용하려면 대량 파라미터 편집 버튼을 선택합니다. 모달에서 파라미터를 편집하고 저장 버튼을 선택합니다.
 - b. 일부 실험 템플릿에는 작업 또는 대상 파라미터가 누락되어 있을 수 있으며, 각 작업 및 대상 카드에 강조 표시되어 있습니다. 각 카드의 편집 버튼을 선택하고 누락된 정보를 추가한 다음 카드에서 저장 버튼을 선택합니다.
 - c. 모든 템플릿에는 서비스 액세스 실행 역할이 필요합니다. 이 실험 템플릿에 대해 기존 역할을 선택하거나 새로운 역할을 생성할 수 있습니다.
 - d. 기존 AWS CloudWatch 경보를 선택하여 하나 이상의 선택적 중지 조건을 정의하는 것이 좋습니다. [AWS FIS의 중지 조건](#) 단원에 대해 자세히 알아보세요. 경보가 아직 구성되지 않은 경우 [Amazon CloudWatch Alarms 사용의](#) 지침에 따라 나중에 실험 템플릿을 업데이트할 수 있습니다.
 - e. Amazon 로그 또는 Amazon S3 버킷에 대한 선택적 실험 CloudWatch 로그를 활성화하는 것이 좋습니다. [AWS FIS에 대한 실험 로깅](#)에 대해 자세히 알아보십시오. 적절한 리소스가 아직 구성되지 않은 경우 나중에 실험 템플릿을 업데이트할 수 있습니다.

6. 실험 템플릿 생성에서 실험 템플릿 생성을 선택합니다.
7. AWS FIS 콘솔의 실험 템플릿 보기에서 실험 시작을 선택합니다. [FIS를 위한 AWS 실험](#) 단원에 대해 자세히 알아보세요.

시나리오 내보내기

시나리오는 콘솔 전용 환경입니다. 실험 템플릿과 비슷하지만 시나리오는 완전한 실험 템플릿이 아니므로 AWS FIS로 직접 가져올 수 없습니다. 시나리오를 자체 자동화의 일부로 사용하려는 경우 다음 두 가지 경로 중 하나를 사용할 수 있습니다.

1. [시나리오 사용](#)의 단계에 따라 유효한 AWS FIS 실험 템플릿을 만들고 해당 템플릿을 내보내세요.
2. [시나리오 보기](#)의 절차를 따르고, 3단계에 콘텐츠 탭에서 시나리오 콘텐츠를 복사 및 저장한 다음 누락된 파라미터를 수동으로 추가하여 유효한 실험 템플릿을 생성합니다.

AWS FIS 시나리오 라이브러리의 시나리오

시나리오 라이브러리에 포함된 시나리오는 가능한 경우 [태그](#)를 사용하도록 설계되었으며, 각 시나리오는 시나리오 설명의 사전 조건 및 작동 방식 섹션에서 필수 태그를 설명합니다. 이러한 사전 정의된 태그로 리소스에 태그를 지정하거나 대량 파라미터 편집 환경을 사용하여 자체 태그를 설정할 수 있습니다 ([시나리오 사용](#) 참조).

이 참조에서는 AWS FIS 시나리오 라이브러리의 일반적인 시나리오에 대해 설명합니다. AWS FIS 콘솔을 사용하여 지원되는 시나리오를 나열할 수도 있습니다.

자세한 내용은 [시나리오 작업](#) 단원을 참조하세요.

AWS FIS는 다음과 같은 Amazon EC2 시나리오를 지원합니다. 이러한 시나리오는 [태그](#)를 사용하는 인스턴스를 대상으로 합니다. 자체 태그를 사용하거나 시나리오에 포함된 기본 태그를 사용할 수 있습니다. 이러한 시나리오 중 일부는 [SSM 문서를 사용](#)합니다.

- EC2 스트레스: 인스턴스 오류 - 하나 이상의 EC2 인스턴스를 중지하여 인스턴스 오류가 미치는 영향을 살펴봅니다.

현재 리전에서 특정 태그가 부착된 인스턴스를 대상으로 합니다. 이 시나리오에서는 이러한 인스턴스를 중지하고 작업 기간이 끝날 때(기본적으로 5분) 다시 시작합니다.

- EC2 스트레스: 디스크 - 디스크 사용률 증가가 EC2 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 특정 태그가 부착된 현재 리전의 EC2 인스턴스를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EC2 인스턴스에 주입되는 디스크 사용량 증가를 사용자 지정할 수 있습니다(기본적으로 각 디스크 스트레스 작업에 대해 5분).

- EC2 스트레스: CPU - CPU 증가가 EC2 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 특정 태그가 부착된 현재 리전의 EC2 인스턴스를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EC2 인스턴스에 주입되는 CPU 스트레스 양 증가를 사용자 지정할 수 있습니다(기본적으로 각 CPU 스트레스 작업에 대해 5분).

- EC2 스트레스: 메모리 - 메모리 사용률 증가가 EC2 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 특정 태그가 부착된 현재 리전의 EC2 인스턴스를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EC2 인스턴스에 주입되는 메모리 스트레스 양 증가를 사용자 지정할 수 있습니다(기본적으로 각 메모리 스트레스 작업에 대해 5분).

- EC2 스트레스: 네트워크 지연 시간 - 네트워크 지연 시간 증가가 EC2 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 특정 태그가 부착된 현재 리전의 EC2 인스턴스를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EC2 인스턴스에 주입되는 네트워크 지연 시간 증가를 사용자 지정할 수 있습니다(기본적으로 각 지연 시간 작업에 대해 5분).

AWS FIS는 다음과 같은 Amazon EKS 시나리오를 지원합니다. 이 시나리오는 Kubernetes 애플리케이션 레이블을 사용하는 EKS 포드를 대상으로 합니다. 자체 레이블을 사용하거나 시나리오에 포함된 기본 레이블을 사용할 수 있습니다. FIS를 사용하는 EKS에 대한 자세한 내용은 [EKS 포드 작업 사용](#) 단원을 참조하세요.

- EKS 스트레스: 포드 삭제 - 하나 이상의 포드를 삭제하여 EKS 포드 오류의 영향을 살펴보세요.

이 시나리오에서는 애플리케이션 레이블과 연결된 현재 리전의 포드를 대상으로 합니다. 이 시나리오에서는 일치하는 모든 포드를 종료합니다. 포드의 재생성은 kubernetes 구성에 의해 제어됩니다.

- EKS 스트레스: CPU - CPU 증가가 EKS 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 애플리케이션 레이블과 연결된 현재 리전의 포드를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EKS 포드에 주입되는 CPU 스트레스 양 증가를 사용자 지정할 수 있습니다(기본적으로 각 CPU 스트레스 작업에 대해 5분).

- EKS 스트레스: 디스크 - 디스크 사용률 증가가 EKS 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 애플리케이션 레이블과 연결된 현재 리전의 포드를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EKS 포드에 주입되는 디스크 스트레스 양 증가를 사용자 지정할 수 있습니다(기본적으로 각 CPU 스트레스 작업에 대해 5분).

- EKS 스트레스: 메모리 - 메모리 사용률 증가가 EKS 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 애플리케이션 레이블과 연결된 현재 리전의 포드를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EKS 포드에 주입되는 메모리 스트레스 양 증가를 사용자 지정할 수 있습니다(기본적으로 각 메모리 스트레스 작업에 대해 5분).

- EKS 스트레스: 네트워크 지연 시간 - 네트워크 지연 시간 증가가 EKS 기반 애플리케이션에 미치는 영향을 살펴봅니다.

이 시나리오에서는 애플리케이션 레이블과 연결된 현재 리전의 포드를 대상으로 합니다. 이 시나리오에서는 작업 기간 동안 대상 EKS 포드에 주입되는 네트워크 지연 시간 증가를 사용자 지정할 수 있습니다(기본적으로 각 지연 시간 작업에 대해 5분).

AWS FIS는 다중 AZ 및 다중 리전 애플리케이션에 대해 다음 시나리오를 지원합니다. 이러한 시나리오는 여러 리소스 유형을 대상으로 합니다.

- AZ Availability: Power Interruption - 가용 영역(AZ)에서 전원이 완전히 중단되었을 때 예상되는 증상을 입력합니다. [AZ Availability: Power Interruption](#) 단원에 대해 자세히 알아보세요.
- Cross-Region: Connectivity - 실험 리전에서 대상 리전으로의 애플리케이션 네트워크 트래픽을 차단하고 리전 간 데이터 복제를 일시 중지합니다. [Cross-Region: Connectivity](#)를 사용하는 방법에 대해 자세히 알아보세요.

AZ Availability: Power Interruption

AZ Availability: Power Interruption 시나리오를 사용하여 가용 영역(AZ)에서 전원이 완전히 중단될 때 예상되는 증상을 유도할 수 있습니다.

이 시나리오는 한 번의 완전한 AZ 전원 중단 시 다중 AZ 애플리케이션이 예상대로 작동하는 것을 입증하는 데 사용할 수 있습니다. 여기에는 영역 컴퓨팅 손실 (Amazon EC2, EKS 및 ECS), AZ에서의 컴퓨팅 재조정 없음, 서브넷 연결 손실, RDS 장애 조치, 장애 조치 ElastiCache, 응답하지 않는 EBS 볼륨 등이 포함됩니다. 기본적으로 대상이 발견되지 않는 작업은 건너뛰게 됩니다.

작업

다음 조치를 함께 취하면 단일 AZ에서 완전한 정전 시 예상되는 여러 가지 증상이 나타납니다. AZ 가용성: 전원 중단은 단일 AZ 전원 중단 중에 영향을 받을 것으로 예상되는 서비스에만 영향을 줍니다. 기본적으로 시나리오는 30분 동안 전원 중단 증상을 주입한 다음 추가로 30분 동안 복구 중에 발생할 수 있는 증상을 주입합니다.

Stop-Instances

AZ 전원이 중단되는 동안 영향을 받는 AZ의 EC2 인스턴스는 종료됩니다. 전원이 복구되면 인스턴스가 재부팅됩니다. AZ Availability: Power Interruption에는 중단 기간 동안 영향을 받는 AZ의 모든 인스턴스를 중지하는 [aws:ec2:stop-instances](#)가 포함됩니다. 기간이 지나면 인스턴스가 다시 시작됩니다. Amazon EKS에서 관리하는 EC2 인스턴스를 중지하면 종속된 EKS 포드가 삭제됩니다. Amazon ECS에서 관리하는 EC2 인스턴스를 중지하면 종속된 ECS 태스크가 중지됩니다.

이 작업은 영향을 받는 AZ에서 실행 중인 EC2 인스턴스를 대상으로 합니다. 기본적으로 이 작업은 값이 StopInstances인 AzImpairmentPower라는 이름의 태그가 있는 인스턴스를 대상으로 합니다. 이 태그를 인스턴스에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 인스턴스를 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

Stop-ASG-Instances

AZ 전원이 중단되는 동안 영향을 받는 AZ의 Auto Scaling 그룹에서 관리하는 EC2 인스턴스가 종료됩니다. 전원이 복구되면 인스턴스가 재부팅됩니다. AZ Availability: Power Interruption에는 중단 기간 동안 영향을 받는 AZ에서 Auto Scaling으로 관리되는 인스턴스를 포함한 모든 인스턴스를 중지하는 [aws:ec2:stop-instances](#)가 포함됩니다. 기간이 지나면 인스턴스가 다시 시작됩니다.

이 작업은 영향을 받는 AZ에서 실행 중인 EC2 인스턴스를 대상으로 합니다. 기본적으로 이 작업은 값이 IceAsg인 AzImpairmentPower라는 이름의 태그가 있는 인스턴스를 대상으로 합니다. 이 태그를 인스턴스에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 인스턴스를 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

인스턴스 시작 일시 중지

AZ 전원이 중단되는 동안에는 AZ에서 용량을 프로비저닝하기 위한 EC2 API 호출이 실패합니다. 특히 다음 API가 영향을 받습니다:,,. ec2:StartInstances ec2:CreateFleet ec2:RunInstances
AZ Availability: Power Interruption includes 영향을 받는 AZ에서 새 인스턴스가 프로비저닝되지 않도록 하는 [aws:ec2: api-insufficient-instance-capacity -error](#)가 포함되어 있습니다.

이 작업은 인스턴스 프로비저닝에 사용되는 IAM 역할을 대상으로 합니다. 이러한 역할은 ARN을 사용하여 타겟팅해야 합니다. 기본적으로 유효한 IAM 역할을 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

ASG 스케일링 일시 중지

AZ 전원 중단 시 Auto Scaling 컨트롤 플레인이 AZ에서 손실된 용량을 복구하기 위해 호출하는 EC2 API는 실패합니다. 특히,,, 다음과 같은 API가 영향을 받습니다. `ec2:StartInstances` `ec2:CreateFleet` `ec2:RunInstances` AZ Availability: Power Interruption 영향을 받는 AZ에서 새 인스턴스가 프로비저닝되지 않도록 하는 [aws:ec2: asg-insufficient-instance-capacity -error가](#) 포함되어 있습니다. 또한 영향을 받는 AZ에서 Amazon EKS 및 Amazon ECS를 확장할 수 없습니다.

이 작업은 Auto Scaling 그룹을 대상으로 합니다. 기본적으로 이 작업은 값이 `IceAsg인` `AzImpairmentPower`라는 이름의 태그가 있는 Auto Scaling 그룹을 대상으로 합니다. 이 태그를 Auto Scaling 그룹에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 Auto Scaling 그룹을 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

네트워크 연결 일시 중지

AZ 전원이 중단되는 동안에는 AZ의 네트워킹을 사용할 수 없습니다. 이 경우 일부 AWS 서비스는 영향을 받는 AZ의 프라이빗 엔드포인트를 사용할 수 없음을 반영하여 DNS를 업데이트하는 데 최대 몇 분 정도 걸릴 수 있습니다. 이 시간 동안 DNS 조회에서 액세스할 수 없는 IP 주소가 반환될 수 있습니다. AZ Availability: Power Interruption에는 영향을 받는 AZ의 모든 서브넷에 대한 모든 네트워크 연결을 2분 동안 차단하는 [aws:network:disrupt-connectivity](#)가 포함됩니다. 이렇게 하면 대부분의 애플리케이션에서 시간 초과 및 DNS 새로 고침이 강제로 실행됩니다. 2분 후에 이 작업을 종료하면 AZ를 계속 사용할 수 없는 동안 리전 서비스 DNS를 나중에 복구할 수 있습니다.

이 작업은 서브넷을 대상으로 합니다. 기본적으로 이 작업은 값이 `DisruptSubnet인` `AzImpairmentPower`라는 이름의 태그가 있는 클러스터를 대상으로 합니다. 이 태그를 서브넷에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 서브넷을 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

장애 조치 RDS

AZ 전원이 중단되는 동안 영향을 받는 AZ의 RDS 노드는 종료됩니다. 영향을 받는 AZ의 단일 AZ RDS 노드는 완전히 사용할 수 없게 됩니다. 다중 AZ 클러스터의 경우, 쓰기 노드는 영향을 받지 않는 AZ로 장애 조치되고 영향을 받는 AZ의 읽기 노드는 사용할 수 없게 됩니다. 다중 AZ 클러스터의 경우 작성이 영향을 받는 AZ에 있는 경우 장애 조치를 위한 [failover-db-clusteraws:rds:를 AZ Availability: Power Interruption](#) 포함합니다.

이 작업은 RDS 클러스터를 대상으로 합니다. 기본적으로 이 작업은 값이 `DisruptRds`인 `AzImpairmentPower`라는 이름의 태그가 있는 클러스터를 대상으로 합니다. 이 태그를 클러스터에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 클러스터를 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

ElastiCache Redis를 일시 중지합니다.

AZ 전원이 차단되는 동안에는 AZ의 ElastiCache 노드를 사용할 수 없습니다. AZ Availability: Power Interruption 영향을 받는 AZ의 노드를 `interrupt-cluster-az-power` ElastiCache 종료하기 위한 [aws:elasticache](#):가 포함되어 있습니다. 중단이 지속되는 동안, 영향을 받는 AZ에서 새 인스턴스가 프로비저닝되지 않으므로 클러스터는 감소된 용량으로 유지됩니다.

이 작업은 클러스터를 대상으로 합니다. ElastiCache 기본적으로 이 작업은 값이 `ElasticacheImpact`인 `AzImpairmentPower`라는 이름의 태그가 있는 클러스터를 대상으로 합니다. 이 태그를 클러스터에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 클러스터를 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다. 영향을 받는 AZ에 쓰기 노드가 있는 클러스터만 유효한 대상으로 간주됩니다.

EBS I/O 일시 중지

AZ 전원 중단 후 전원이 복구되면 극히 일부 인스턴스에서 EBS 볼륨이 응답하지 않는 현상이 발생할 수 있습니다. AZ Availability: Power Interruption에는 [aws:efs:pause-io](#)가 포함되어 EBS 볼륨 1개를 응답하지 않는 상태로 남겨둡니다.

기본적으로 인스턴스가 종료된 후에도 지속되도록 설정된 볼륨만 대상이 됩니다. 이 작업은 값이 `APIPauseVolume`인 `AzImpairmentPower`라는 태그가 있는 볼륨을 대상으로 합니다. 이 태그를 볼륨에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 볼륨을 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

제한 사항

- 이 시나리오에는 [중지 조건](#)이 포함되어 있지 않습니다. 애플리케이션에 맞는 올바른 중지 조건을 실험 템플릿에 추가해야 합니다.
- AWS Fargate에서 실행되는 Amazon EKS 포드는 지원되지 않습니다.
- AWS Fargate에서 실행되는 Amazon ECS 태스크는 지원되지 않습니다.
- 읽기 가능한 대기 DB 인스턴스 2개가 있는 [Amazon RDS 다중AZ](#)는 지원되지 않습니다. 이 경우 인스턴스가 종료되고, RDS가 장애 조치되며, 영향을 받는 AZ에서 용량이 즉시 다시 프로비저닝됩니다. 영향을 받는 AZ의 읽기 가능한 대기는 계속 사용할 수 있습니다.

요구 사항

- AWS FIS [실험 역할](#)에 필요한 권한을 추가합니다.
- 리소스 태그는 실험의 대상이 되는 리소스에 적용해야 합니다. 자체 태그 지정 규칙 또는 시나리오에 정의된 기본 태그를 사용할 수 있습니다.

권한

다음 정책은 AZ Availability: Power Interruption 시나리오로 실험을 실행하는 데 필요한 권한을 AWS FIS에 부여합니다. 이 정책은 [실험 역할](#)에 연결되어야 합니다.

```
{
  "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": [
      "*"
    ]
  }
]
}

```

시나리오 콘텐츠

다음 콘텐츠는 시나리오를 정의합니다. 이 JSON을 저장하고 AWS 명령줄 인터페이스 (AWS CLI) 의 [create-experiment-template](#) 명령을 사용하여 [실험 템플릿](#)을 생성하는 데 사용할 수 있습니다. 최신 버전의 시나리오를 보려면 FIS 콘솔의 시나리오 라이브러리를 방문하세요.

```

{
  "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:redis-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:interrupt-cluster-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

Cross-Region: Connectivity 시나리오를 사용하여 실험 리전에서 대상 리전으로의 애플리케이션 네트워크 트래픽을 차단하고 Amazon S3 및 Amazon DynamoDB에 대한 교차 리전 복제를 일시 중지할 수 있습니다. 교차 리전: 연결은 실험을 실행하는 리전(실험 리전)에서의 아웃바운드 애플리케이션 트래픽에 영향을 미칩니다. 실험 리전(대상 리전)에서 격리하려는 리전으로부터의 상태 비저장 인바운드 트래픽은 차단되지 않을 수 있습니다. WS 관리형 서비스의 트래픽은 차단되지 않을 수 있습니다.

이 시나리오는 실험 리전에서 대상 리전의 리소스에 액세스할 수 없는 경우 다중 리전 애플리케이션이 예상대로 작동하는지 시연하는 데 사용할 수 있습니다. 여기에는 전송 게이트웨이와 라우트 테이블을 대상으로 실험 리전에서 대상 리전으로의 네트워크 트래픽을 차단하는 것이 포함됩니다. 또한 S3 및

DynamoDB에 대한 리전 간 복제를 일시 중지합니다. 기본적으로 대상이 발견되지 않는 작업은 건너뛰게 됩니다.

작업

다음 작업은 함께 포함된 AWS 서비스에 대한 교차 리전 연결을 차단합니다. 작업은 병렬로 실행됩니다. 기본적으로 시나리오는 3시간 동안 트래픽을 차단하며, 최대 12시간까지 늘릴 수 있습니다.

Transit Gateway 연결 중단

Cross Region: Connectivity 실험 지역의 VPC에서 [트랜짓 게이트웨이로 연결된 대상 지역의 VPC로의 지역 간 네트워크 트래픽을 차단하는 aws:network: transit-gateway-disrupt-cross -region-connectivity 기능이](#) 포함되어 있습니다. 이 작업은 실험 리전 내의 VPC 엔드포인트에 대한 액세스에는 영향을 미치지 않지만, 실험 리전에서 대상 리전의 VPC 엔드포인트로 향하는 트래픽을 차단합니다.

이 작업은 실험 리전과 대상 리전을 연결하는 전송 게이트웨이를 대상으로 합니다. 기본적으로 이 작업은 값이 Allowed인 DisruptTransitGateway라는 이름의 [태그](#)가 있는 전송 게이트웨이를 대상으로 합니다. 이 태그를 전송 게이트웨이에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 전송 게이트웨이를 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

서브넷 연결 중단

Cross Region: Connectivity 실험 지역의 VPC에서 [대상 지역의 퍼블릭 AWS IP 블록으로의 지역 간 네트워크 트래픽을 차단하기 위한 aws:network: route-table-disrupt-cross -region 연결을](#) 포함합니다. 이러한 퍼블릭 IP 블록에는 대상 리전의 AWS 서비스 엔드포인트(예: S3 리전 엔드포인트)와 관리형 서비스용 AWS IP 블록(예: 로드 밸런서 및 Amazon API Gateway에 사용되는 IP 주소)이 포함됩니다. 이 작업은 실험 리전에서 대상 리전으로의 교차 리전 VPC 피어링 연결을 통한 네트워크 연결도 차단합니다. 이는 실험 리전 내의 VPC 엔드포인트에 대한 액세스에는 영향을 미치지 않지만, 실험 리전에서 대상 리전의 VPC 엔드포인트로 향하는 트래픽을 차단합니다.

이 작업은 실험 리전의 서브넷을 대상으로 합니다. 기본적으로 이 작업은 값이 Allowed인 DisruptSubnet라는 이름의 [태그](#)가 있는 서브넷을 대상으로 합니다. 이 태그를 서브넷에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 서브넷을 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

S3 복제 일시 중지

Cross Region: Connectivity 대상 [버킷에 대해 실험 지역에서 대상 bucket-pause-replication 지역으로의 S3 복제를 일시 중지하는 aws:s3:가](#) 포함되어 있습니다. 대상 리전에서 실험 리전으로의 복제는 영

향을 받지 않습니다. 시나리오가 종료되면 일시 중지된 시점부터 버킷 복제가 다시 시작됩니다. 복제가 모든 객체를 동기화 상태로 유지하는 데 걸리는 시간은 실험 기간과 버킷에 대한 객체 업로드 속도에 따라 달라질 수 있습니다.

이 작업은 대상 리전의 S3 버킷에 [교차 리전 복제\(CPR\)](#)를 사용하도록 설정한 실험 리전의 S3 버킷을 대상으로 합니다. 기본적으로 이 작업은 값이 Allowed인 DisruptS3라는 이름의 [태그](#)가 있는 버킷을 대상으로 합니다. 이 태그를 버킷에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 버킷을 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

DynamoDB 복제 일시 중지

Cross-Region: Connectivity 실험 지역과 대상 지역을 포함한 다른 모든 지역 간의 [encrypted-global-table-pause](#) 복제를 일시 중지하는 `aws:dynamodb: -replication`이 포함되어 있습니다. 이렇게 하면 실험 리전 안팎으로의 복제는 방지되지만 다른 리전 간의 복제에는 영향을 미치지 않습니다. 시나리오가 종료되면 일시 중지된 시점부터 테이블 복제가 다시 시작됩니다. 복제가 모든 데이터를 동기화 상태로 유지하는 데 걸리는 시간은 실험 기간과 테이블에 대한 변경 속도에 따라 달라질 수 있습니다.

이 작업은 [고객 관리형 키](#)로 암호화된 실험 리전의 [암호화된 DynamoDB](#) 글로벌 테이블을 대상으로 합니다. 기본적으로 이 작업은 값이 Allowed인 DisruptDynamoDb라는 이름의 [태그](#)가 있는 테이블을 대상으로 합니다. 이 태그를 테이블에 추가하거나 실험 템플릿에서 기본 태그를 사용자 지정 태그로 바꿀 수 있습니다. 기본적으로 유효한 글로벌 테이블을 찾을 수 없는 경우 이 작업은 건너뛰게 됩니다.

제한 사항

- 이 시나리오에는 [중지 조건](#)이 포함되어 있지 않습니다. 애플리케이션에 맞는 올바른 중지 조건을 실험 템플릿에 추가해야 합니다.

요구 사항

- AWS FIS [실험 역할](#)에 필요한 권한을 추가합니다.
- 리소스 태그는 실험의 대상이 되는 리소스에 적용해야 합니다. 자체 태그 지정 규칙 또는 시나리오에 정의된 기본 태그를 사용할 수 있습니다.

권한

다음 정책은 Cross-Region: Connectivity 시나리오로 실험을 실행하는 데 필요한 권한을 AWS FIS에 부여합니다. 이 정책은 [실험 역할](#)에 연결되어야 합니다.

```
{
  "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": "DdbCrossRegion2",
  "Effect": "Allow",
  "Action": [
    "dynamodb:DescribeTable",
    "dynamodb:DescribeGlobalTable"
  ],
  "Resource": [
    "arn:aws:dynamodb:*:*:table/*",
```

```

        "arn:aws:dynamodb:*:*:global-table/*"
    ]
},
{
    "Sid": "DdbCrossRegion3",
    "Effect": "Allow",
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:PutKeyPolicy"
    ],
    "Resource": "arn:aws:kms:*:*:key/*"
}
]
}

```

시나리오 콘텐츠

다음 콘텐츠는 시나리오를 정의합니다. 이 JSON을 저장하고 AWS 명령줄 인터페이스 (AWS CLI) 의 [create-experiment-template](#) 명령을 사용하여 [실험 템플릿](#)을 생성하는 데 사용할 수 있습니다. 최신 버전의 시나리오를 보려면 FIS 콘솔의 시나리오 라이브러리를 방문하세요.

```

{
    "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:encrypted-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:encrypted-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"
}
```


FIS를 위한 AWS 실험

AWS FIS를 사용하면 AWS 워크로드에 대한 오류 주입 실험을 수행할 수 있습니다. 시작하려면 [실험 템플릿](#)을 만드세요. 실험 템플릿을 만든 후 이를 사용하여 실험을 시작할 수 있습니다.

다음 중 한 경우에 실험이 끝납니다.

- 템플릿의 모든 [작업](#)이 성공적으로 완료되었습니다.
- [중지 조건](#)이 트리거됩니다.
- 오류로 인해 작업을 완료할 수 없습니다. [대상](#)을 찾을 수 없는 경우를 예로 들 수 있습니다.
- 실험은 [수동으로 중지](#)됩니다.

중지되거나 실패한 실험은 재개할 수 없습니다. 또한 완료된 실험은 다시 실행할 수 없습니다. 하지만 동일한 실험 템플릿에서 새 실험을 시작할 수 있습니다. 새 실험에서 다시 지정하기 전에 실험 템플릿을 선택적으로 업데이트할 수도 있습니다.

Tasks

- [실험 시작](#)
- [실험 보기](#)
- [실험에 태그 지정](#)
- [실험 중지](#)
- [확인된 대상 나열](#)

실험 시작

실험 템플릿에서 실험을 시작합니다. 자세한 정보는 [템플릿에서 실험 시작](#)을 참조하세요.

Amazon EventBridge를 사용하여 실험을 일회성 작업 또는 반복 작업으로 예약할 수 있습니다. 자세한 정보는 [자습서: 반복 실험 예약](#)을 참조하세요.

다음 기능 중 하나를 사용하여 실험을 모니터링할 수 있습니다.

- AWS FIS 콘솔에서 실험을 볼 수 있습니다. 자세한 정보는 [실험 보기](#)을 참조하세요.
- 실험의 대상 리소스에 대한 Amazon CloudWatch 지표 또는 AWS FIS 사용 지표를 확인하십시오. 자세한 정보는 [CloudWatch를 사용한 모니터링](#)을 참조하세요.

- 실험 로깅을 활성화하여 실행 중인 실험에 대한 세부 정보를 캡처할 수 있습니다. 자세한 내용은 [실험 로깅](#) 섹션을 참조하세요.

실험 보기

실행 중인 실험의 진행 상황을 볼 수 있으며 완료, 중지 또는 실패한 실험을 볼 수 있습니다.

중지, 완료, 실패한 실험은 120일 후 계정에서 자동으로 제거됩니다.

콘솔을 사용하여 실험을 보려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험을 선택합니다.
3. 실험의 실험 ID를 선택하여 세부 정보 페이지를 엽니다.
4. 다음 중 한 개 이상을 수행할 수 있습니다.
 - 세부 정보, 상태에서 [실험 상태](#)를 확인합니다.
 - 실험 작업에 대한 정보를 보려면 작업 탭을 선택합니다.
 - 실험 대상 대한 정보를 보려면 대상 탭을 선택합니다.
 - 타임라인 탭을 선택하면 시작 및 종료 시간을 기준으로 작업을 시각적으로 표현할 수 있습니다.

콘솔을 사용하여 실험을 보려면

[list-experiments](#) 명령을 사용하여 실험 목록을 가져오고 [get-experiment](#) 명령을 사용하여 특정 실험에 대한 정보를 가져올 수 있습니다.

실험 상태

실험은 다음 상태 중 하나일 수 있습니다.

- 보류 중 - 실험이 보류 중입니다.
- 시작 중 - 실험 시작 준비 중입니다.
- 실행 중 - 실험이 진행 중입니다.
- 완료 - 실험의 모든 작업이 성공적으로 완료되었습니다.
- 중지 중 - 중지 조건이 트리거되었거나 실험이 수동으로 중지되었습니다.
- 중지됨 - 실험에서 실행 중이거나 보류 중인 모든 작업이 중지됩니다.

- 실패 - 권한이 충분하지 않거나 구문이 잘못되는 등의 오류로 인해 실험이 실패했습니다.

작업 상태

작업은 다음 상태 중 하나일 수 있습니다.

- 보류 중 - 실험이 아직 시작되지 않았거나 실험 후반부에 작업을 시작해야 하기 때문에 작업이 보류 중입니다.
- 시작 중 - 작업 시작 준비 중입니다.
- 실행 중 - 작업이 실행 중입니다.
- 완료됨 - 작업이 성공적으로 완료되었습니다.
- 취소됨 - 작업이 시작되기 전에 실험이 중지되었습니다.
- 건너뛴 - 작업을 건너뛰었습니다.
- 중지 중 - 작업을 중지하는 중입니다.
- 중지됨 - 실험에서 실행 중이거나 보류 중인 모든 작업이 중지됩니다.
- 실패 - 권한이 충분하지 않거나 구문이 잘못되는 등의 클라이언트 오류로 인해 작업이 실패했습니다.

실험에 태그 지정

실험에 태그를 적용하여 실험을 구성하는 데 도움을 줄 수 있습니다. 또한 [태그 기반 IAM 정책](#)을 구현하여 실험에 대한 액세스를 제어할 수 있습니다.

콘솔을 사용하여 실험에 태그를 지정하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험을 선택합니다.
3. 실험을 선택한 다음 작업, 태그 관리를 선택합니다.
4. 새 태그를 추가하려면 새 태그 추가를 선택하고 키와 값을 지정합니다.

태그를 제거하려면 해당 태그에서 제거를 선택합니다.

5. 저장을 선택합니다.

CLI를 사용하여 실험에 태그를 지정하려면

[tag-resource](#) 명령을 사용합니다.

실험 중지

실행 중인 실험을 언제든지 중지할 수 있습니다. 실험을 중지하면 해당 작업에 대해 완료되지 않은 모든 게시 작업은 실험이 중지되기 전에 완료됩니다. 중지된 실험은 재개할 수 없습니다.

콘솔을 사용하여 실험을 중지하려면

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험을 선택합니다.
3. 실험을 선택하고 실험 중지를 선택합니다.
4. 확인 대화 상자에서 실험 중지를 선택합니다.

CLI를 사용하여 실험을 중지하려면

[stop-experiment](#) 명령을 사용합니다.

확인된 대상 나열

대상 확인이 종료된 후 실험의 확인된 대상에 대한 정보를 볼 수 있습니다.

콘솔을 사용하여 확인된 대상을 보려면 다음을 수행하세요.

1. <https://console.aws.amazon.com/fis/> 에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험을 선택합니다.
3. 실험을 선택하고 보고서를 선택합니다.
4. 리소스에서 확인된 대상 정보를 봅니다.

CLI를 사용하여 확인된 대상을 보려면 다음을 수행하세요.

[list-experiment-resolved-targets](#) 명령을 사용합니다.

실험 스케줄러

AWS Fault Injection Service(FIS)를 통해 AWS 워크로드에 대한 오류 주입 실험을 수행할 수 있습니다. 이러한 실험은 지정된 대상에서 실행할 하나 이상의 작업이 포함된 템플릿에서 실행됩니다. 이제 FIS 콘솔에서 기본적으로 실험을 일회성 작업 또는 반복 작업으로 예약할 수 있습니다. 이제 FIS는 [예약된 규칙](#) 외에도 새로운 일정 예약 기능을 제공합니다. FIS는 이제 EventBridge 스케줄러와 통합되어 사용자를 대신하여 규칙을 생성합니다. EventBridge Scheduler는 하나의 중앙 관리형 서비스에서 작업을 생성, 실행 및 관리할 수 있는 서버리스 스케줄러입니다.

Important

AWS GovCloud (미국 동부) 및 AWS (미국 서부)에서는 실험 스케줄러를 사용할 수 없습니다.
AWS Fault Injection Service GovCloud

주제

- [시작하기](#)
- [FIS 실험 예약](#)
- [콘솔을 사용하여 일정을 업데이트하려면](#)
- [실험 일정 업데이트](#)
- [콘솔을 사용하여 실험 실행을 비활성화하거나 삭제](#)

시작하기

실행 역할은 AWS Fault Injection Service가 스케줄러와 상호 작용하고 Event Bridge EventBridge 스케줄러가 FIS 실험을 시작하기 위해 맡는 IAM 역할입니다. 이 역할에 권한 정책을 추가하여 EventBridge 스케줄러에게 FIS Experiment를 호출할 수 있는 액세스 권한을 부여합니다. 다음 단계에서는 새 실행 역할을 만드는 방법과 실험 시작을 허용하는 EventBridge 정책을 설명합니다.

AWS CLI를 사용하여 스케줄러 역할 생성

이는 Event Bridge가 고객을 대신하여 실험 일정을 잡을 수 있도록 하는 데 필요한 IAM 역할입니다.

1. 다음 역할 수임 JSON 정책을 복사하고 로컬에 `fis-execution-role.json`로 저장하세요. 이 신뢰 정책을 통해 EventBridge Scheduler는 사용자를 대신하여 역할을 맡을 수 있습니다.

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

2. AWS Command Line Interface(AWS CLI)를 사용하여 다음 명령을 입력하여 새 역할을 생성합니다. FisSchedulerExecutionRole를 이 역할에 부여하려는 이름으로 바꾸세요.

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

성공하면 다음과 같은 결과가 출력됩니다.

```
{
  "Role": {
    "Path": "/",
    "RoleName": "FisSchedulerExecutionRole",
    "RoleId": "AROAZL22PDN5A6WKRBNUN",
    "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. EventBridge Scheduler가 실험을 호출할 수 있도록 허용하는 새 정책을 만들려면 다음 JSON을 복사하고 로컬에 다른 이름으로 저장합니다. `fis-start-experiment-permissions.json` 다음 정책은 EventBridge Scheduler가 계정의 모든 실험 템플릿에서 `fis:StartExperiment` 작업을 호출하도록 허용합니다. 역할을 단일 실험 템플릿으로 제한하려면 `"arn:aws:fis:*:*:experiment-template/*"` 끝에 있는 `*`를 실험 템플릿의 ID로 바꾸세요.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "fis:StartExperiment",
      "Resource": [
        "arn:aws:fis:*:*:experiment-template/*",
        "arn:aws:fis:*:*:experiment/*"
      ]
    }
  ]
}
```

4. 다음 명령을 실행하여 새 권한 정책을 생성합니다. `FisSchedulerPolicy`를 이 정책에 부여하려는 이름으로 바꾸세요.

```
aws iam create-policy --policy-name FisSchedulerPolicy --policy-document file://fis-start-experiment-permissions.json
```

성공하면 다음과 같은 결과가 출력됩니다. 정책 ARN을 기록합니다. 다음 단계에서 이 ARN을 사용하여 정책을 실행 역할에 연결합니다.

```
{
  "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. 실행 역할에 정책을 연결하려면 다음 명령을 실행하세요. `your-policy-arn`을 이전 단계에서 생성한 정책의 ARN으로 변경합니다. `FisSchedulerExecutionRole`을 실행 역할의 이름으로 바꿉니다.

```

aws iam attach-role-policy --policy-arn your-policy-arn --role-name
FisSchedulerExecutionRole

```

이 `attach-role-policy` 작업은 명령줄에서 응답을 반환하지 않습니다.

6. 특정 태그 값이 있는 AWS FIS 실험만 실행하도록 스케줄러를 제한할 수 있습니다. 예를 들어 다음 정책은 모든 AWS FIS 실험 템플릿에 대한 `fis:StartExperiment` 권한을 부여하지만 스케줄러는 `Purpose=Schedule` 태그가 지정된 실험만 실행하도록 제한합니다.

```

{
  "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"
        }
      }
    }
  ]
}

```


FIS 실험 예약

실험 일정을 잡기 전에 일정이 간접적으로 호출할 하나 이상의 [실험 템플릿](#)이 필요합니다. 기존의 AWS 리소스를 사용하거나 새로 생성할 수 있습니다.

실험 템플릿이 생성되면 작업을 클릭하고 실험 예약을 선택합니다. 실험 예약 페이지로 리디렉션됩니다. 일정의 이름이 자동으로 채워집니다.

일정 패턴 섹션에 따라 일회성 일정 또는 반복 일정을 선택하세요. 필수 입력 필드를 채우고 권한으로 이동하세요.

The screenshot shows the AWS FIS 'Schedule pattern' configuration interface. It includes the following sections:

- Schedule pattern**
 - Occurrence**: Radio buttons for 'One-time schedule' (selected) and 'Recurring schedule'.
 - Date and time**: Fields for 'YYYY/MM/DD', 'hh:mm', and a 'Timezone' dropdown (set to '(UTC -04:00) America/New...').
 - Flexible time window**: A dropdown menu labeled 'Select'.
- Schedule state**
 - Enable schedule**: A toggle switch set to 'Enable'.

일정 상태는 기본적으로 활성화되어 있습니다. 참고: 일정 상태를 비활성화하면 일정을 생성하더라도 실험이 예약되지 않습니다.

AWS FIS [실험 스케줄러는 스케줄러 위에 구축되었습니다](#). [EventBridge](#) 지원되는 다양한 일정 유형에 대한 설명서를 참조할 수 있습니다.

콘솔을 사용하여 일정을 업데이트하려면

1. [AWS FIS 콘솔](#)을 엽니다.
2. 왼쪽 탐색 창에서 실험 템플릿을 선택합니다.

3. 일정을 생성하려는 실험 템플릿을 선택합니다.
4. 작업을 클릭하고 드롭다운에서 실험 예약을 선택합니다.
 - a. 일정 이름 아래에 이름이 자동으로 채워집니다.
 - b. 일정 패턴에서 반복 일정을 선택합니다.
 - c. 일정 유형에서 요금 기반 일정을 선택할 수 있습니다. [일정 유형](#)을 참조하세요.
 - d. rate 표현식에서 실험 실행 시간보다 느린 속도(예: 5분)를 선택합니다.
 - e. 기간에서 시간대를 선택합니다.
 - f. 시작 날짜 및 시간에서 시작 날짜 및 시간을 지정합니다.
 - g. 종료 날짜 및 시간에서 종료 날짜 및 시간을 지정합니다.
 - h. 일정 상태에서 일정 활성화 옵션을 토글합니다.
 - i. 권한 아래에서 기존 역할 사용을 선택한 다음 `FisSchedulerExecutionRole`을 검색합니다.
 - j. 다음을 선택합니다.
5. 일정 검토 및 생성을 선택하고 스케줄러 세부 정보를 검토한 다음 일정 생성을 선택합니다.

실험 일정 업데이트

실험 일정이 적합한 특정 날짜와 시간에 발생하도록 업데이트할 수 있습니다.

콘솔을 사용하여 실험 실행을 업데이트하려면

1. [Amazon FIS 콘솔](#)을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 일정이 이미 생성된 리소스 유형: 실험 템플릿을 선택합니다.
4. 템플릿의 실험 ID를 클릭합니다. 그런 다음 일정 탭으로 이동합니다.
5. 실험과 관련된 기존 일정이 있는지 확인하세요. 관련 일정을 선택하고 일정 업데이트 버튼을 클릭합니다.

콘솔을 사용하여 실험 실행을 비활성화하거나 삭제

실험이 일정에 따라 실행되거나 실행되지 않도록 하려면 규칙을 삭제하거나 비활성화할 수 있습니다. 다음 단계는 실험 실행을 삭제하거나 비활성화하는 방법을 안내합니다.

규칙을 삭제하거나 비활성화하려면

1. [Amazon FIS 콘솔](#)을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 일정이 이미 생성된 리소스 유형: 실험 템플릿을 선택합니다.
4. 템플릿의 실험 ID를 클릭합니다. 그런 다음 일정 탭으로 이동합니다.
5. 실험과 관련된 기존 일정이 있는지 확인하세요. 관련 일정을 선택하고 일정 업데이트 버튼을 클릭합니다.
6. 다음 중 하나를 수행합니다.
 - a. 일정을 삭제하려면 일정 삭제 규칙 옆의 버튼을 선택합니다. delete을 입력하고 일정 삭제 버튼을 클릭합니다.
 - b. 일정을 비활성화하려면 일정 비활성화 규칙 옆의 버튼을 선택합니다. disable을 입력하고 일정 비활성화 버튼을 클릭합니다.

AWS FIS 모니터링

다음 도구를 사용하여 AWS Fault Injection Service(AWS FIS) 실험의 진행 상황과 영향을 모니터링할 수 있습니다.

AWS FIS 콘솔 및 AWS CLI

AWS FIS 콘솔 또는 AWS CLI를 사용하여 실행 중인 실험의 진행 상황을 모니터링할 수 있습니다. 실험의 각 작업 상태와 각 작업의 결과를 볼 수 있습니다. 자세한 설명은 [the section called “실험 보기”](#) 섹션을 참조하세요.

CloudWatch 사용량 지표 및 경보

CloudWatch 사용량 지표를 사용하면 계정의 리소스 사용량을 파악할 수 있습니다. AWS FIS 사용량 지표는 AWS Service Quotas에 해당합니다. 사용량이 서비스 할당량에 가까워지면 경고하는 경보를 구성할 수 있습니다. 자세한 설명은 [CloudWatch를 사용한 모니터링](#) 섹션을 참조하세요.

AWS 실험이 범위를 벗어나는 시기를 정의하는 CloudWatch 경보를 만들어 FIS 실험에 대한 중지 조건을 만들 수도 있습니다. 경보가 트리거되면 실험이 중지됩니다. 자세한 설명은 [중지 조건](#) 섹션을 참조하세요. CloudWatch 경보 생성에 대한 자세한 내용은 Amazon CloudWatch User Guide의 [정적 임계값 기반 CloudWatch CloudWatch 경보 생성 및 예외 항목 탐지에 기반한 경보](#) 생성을 참조하십시오.

AWS FIS 실험 로깅

실험 로깅을 활성화하여 실행 중인 실험에 대한 세부 정보를 캡처할 수 있습니다. 자세한 내용은 [실험 로깅](#) 섹션을 참조하세요.

실험 상태 변경 이벤트

EventBridge Amazon에서는 시스템 이벤트 또는 리소스 변경에 자동으로 대응할 수 있습니다. AWS 실험 상태가 변경되면 FIS에서 알림을 보냅니다. 이벤트가 규칙과 일치할 때 수행할 자동화된 작업을 지정하는 규칙을 관심 있는 이벤트에 대해 생성할 수 있습니다. Amazon SNS 주제에 알림을 보내거나 Lambda 함수를 호출하는 경우를 예로 들 수 있습니다. 자세한 설명은 [다음을 사용하여 모니터링하십시오. EventBridge](#) 섹션을 참조하세요.

CloudTrail 로그

AWS FIS API에 대한 호출에 대한 세부 정보를 캡처하고 Amazon S3에 로그 파일로 저장하는 데 사용합니다. AWS CloudTrail. CloudTrail 또한 실험을 실행 중인 리소스의 서비스 API에 대한 호출도 기록합니다. 이러한 CloudTrail 로그를 사용하여 어떤 호출이 이루어졌는지, 호출이 발생한 소스 IP 주소, 전화를 건 사람, 언제 전화를 걸었는지 등을 확인할 수 있습니다.

AWS Health Dashboard 알림

AWS Health는 리소스 성능과 AWS 서비스 및 계정의 가용성에 대해 지속적인 가시성을 제공합니다. 실험을 시작하면 AWS FIS가 AWS Health Dashboard에 알림을 전송합니다. 이 알림은 다중 계정 실험을 포함하여 실험 대상 리소스가 포함된 각 계정에서 실험이 진행되는 동안 표시됩니다. `aws:ssm:start-automation-execution` 및 `aws:fis:wait`와 같이 대상이 포함되지 않은 작업만 있는 다중 계정 실험은 알림을 보내지 않습니다. 실험을 허용하는 데 사용된 역할에 대한 정보는 영향을 받는 리소스 아래에 나열됩니다. AWS Health Dashboard에 대해 자세히 알아보려면 AWS Health 사용 설명서에서 [AWS Health Dashboard](#)를 참조하세요.

Note

AWS Health는 최선을 다해 이벤트를 제공합니다.

Amazon CloudWatch를 사용한 AWS FIS 사용량 지표 모니터링

Amazon CloudWatch를 사용하여 AWS FIS 실험이 대상에 미치는 영향을 모니터링할 수 있습니다. AWS FIS 사용량을 모니터링할 수도 있습니다.

실험 상태 보기에 대한 자세한 내용은 [실험 보기](#) 섹션을 참조하세요.

AWS FIS 실험 모니터링

AWS FIS 실험을 계획할 때 실험의 대상 리소스 유형에 대한 기준선 또는 “안정 상태”를 식별하는 데 사용할 수 있는 CloudWatch 지표를 식별하세요. 실험을 시작한 후 실험 템플릿을 통해 선택한 대상의 CloudWatch 지표를 모니터링할 수 있습니다.

AWS FIS에서 지원하는 대상 리소스 유형에 사용 가능한 CloudWatch 지표에 대한 자세한 내용은 다음을 참조하세요.

- [CloudWatch를 사용하여 인스턴스 모니터링](#)
- [Amazon ECS CloudWatch 지표](#)
- [CloudWatch를 사용한 Amazon RDS 지표 모니터링](#)
- [CloudWatch를 사용하여 명령 실행 지표 모니터링](#)

AWS FIS 사용량 지표

CloudWatch 사용량 지표를 사용하여 계정의 리소스 사용량을 확인할 수 있습니다. 이러한 지표를 사용하여 CloudWatch 그래프 및 대시보드에서 현재 서비스 사용량을 시각화합니다.

AWS FIS 사용량 지표는 AWS Service Quotas에 해당합니다. 사용량이 서비스 할당량에 가까워지면 경고하는 경보를 구성할 수 있습니다. CloudWatch 경보에 대한 자세한 내용은 [Amazon CloudWatch 사용 설명서](#)를 참조하세요.

AWS FIS는 AWS/Usage 네임스페이스에 다음 지표를 게시합니다.

지표	설명
ResourceCount	계정에서 실행 중인 지정된 리소스의 총 수입니다. 리소스는 지표와 연결된 차원으로 정의됩니다.

다음 차원은 AWS FIS에 의해 게시되는 사용량 지표를 구체화하는 데 사용됩니다.

차원	설명
Service	리소스가 포함된 AWS 서비스의 이름 AWS FIS 사용량 지표의 경우 이 차원 값은 FIS입니다.
Type	보고되는 엔터티의 유형입니다. 현재 AWS FIS 사용량 지표에 대한 유일한 유효 값은 Resource입니다.
Resource	실행 중인 리소스의 유형입니다. 가능한 값은 실험 템플릿의 ExperimentTemplates , 진행 중인 실험의 ActiveExperiments 입니다.
Class	이 차원은 나중에 사용하기 위해 예약되어 있습니다.

Amazon을 사용하여 AWS FIS 실험을 모니터링하세요 EventBridge

실험 상태가 변경되면 AWS FIS는 알림을 보냅니다. 이러한 알림은 Amazon EventBridge (이전의 Events) 을 통해 CloudWatch 이벤트로 제공됩니다. AWS FIS는 최선의 노력을 다해 이러한 이벤트를 내보냅니다. 이벤트는 거의 EventBridge 실시간으로 전달됩니다.

를 사용하면 이벤트에 대한 응답으로 EventBridge 프로그래밍 작업을 트리거하는 규칙을 만들 수 있습니다. 예를 들어 SNS 주제를 호출하여 이메일 알림을 보내거나 Lambda 함수를 호출하여 조치를 취하는 규칙을 구성할 수 있습니다.

에 대한 EventBridge 자세한 내용은 Amazon EventBridge 사용 설명서의 [Amazon EventBridge 시작하기](#)를 참조하십시오.

다음은 실험 상태 변경 이벤트의 구문입니다.

```
{
  "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

상태가 변경된 실험의 ID입니다.

experiment-template-id

실험에 사용된 실험 템플릿의 ID입니다.

new_value

실험의 새로운 상태입니다. 가능한 값은 다음과 같습니다.

- completed
- failed
- initiating
- running
- stopped
- stopping

old_value

실험의 이전 상태입니다. 가능한 값은 다음과 같습니다.

- initiating
- pending
- running
- stopping

AWS FIS에 대한 실험 로깅

실험 로깅을 사용하여 실행 중인 실험에 대한 세부 정보를 캡처할 수 있습니다.

각 로그 대상 유형과 관련된 비용을 기준으로 실험 로깅 요금이 부과됩니다. 자세한 내용은 [Amazon CloudWatch 요금](#) (유료 티어, 로그, 벤드 로그 아래) 및 [Amazon S3 요금](#)을 참조하십시오.

권한

구성한 각 로그 대상으로 로그를 전송할 수 있는 권한을 AWS FIS에 부여해야 합니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 다음을 참조하십시오.

- [로그로 전송된 CloudWatch 로그](#)
- [Amazon S3로 보낸 로그](#)

로그 스키마

다음은 실험 로깅에 사용되는 스키마입니다. 현재 스키마 버전은 2입니다. details의 필드는 log_type의 값에 따라 달라집니다. resolved_targets의 필드는 target_type의 값에 따라 달라집니다. 자세한 설명은 [the section called “로그 레코드 예”](#) 섹션을 참조하세요.

```
{
  "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-ddTth:mm:ssZ",
  "version": "2",
  "details": {
    "account_id": "123456789012",
    "action_end_time": "yyyy-mm-ddTth:mm:ssZ",
    "action_id": "String",
    "action_name": "String",
    "action_start_time": "yyyy-mm-ddTth: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-ddTth:mm:ssZ",
    "experiment_state": {
      "status": "pending | initiating | running | completed | stopping | stopped
| failed",
      "reason": "String"
    },
    "experiment_start_time": "yyyy-mm-ddTth: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-ddT hh:mm:ssZ",
    "target_resolution_start_time": "yyyy-mm-ddT hh:mm:ssZ",
    "target_type": "String",
    "total_pages": Number,
    "total_resolved_targets_count": Number
  }
}

```

릴리스 정보

- 버전 2에서는 다음 사항이 도입됩니다.
 - `target_type` 필드를 변경하고 `resolved_targets` 필드를 ARN 목록에서 객체 목록으로 변경합니다. `resolved_targets` 객체의 유효한 필드는 대상의 [리소스 유형](#)인 `target_type`의 값에 따라 달라집니다.
 - `account_id` 필드를 추가하는 `action-error` 및 `target-resolution-detail` 이벤트 유형입니다.
- 버전 1이 초기 릴리스입니다.

로그 대상

AWS FIS는 다음 대상으로의 로그 전달을 지원합니다.

- Amazon S3 버킷
- 아마존 CloudWatch 로그 로그 그룹

S3 로그 전달

로그는 다음 위치로 전달됩니다.

```

bucket-and-optional-prefix/AWSLogs/account-id/fis/region/experiment-
id/YYYY/MM/DD/account-id_awsfislogs_region_experiment-id_YYYYMMDDHHMMZ_hash.log

```

로그가 버킷으로 전달되려면 몇 분 정도 걸릴 수 있습니다.

CloudWatch 로그 로그 전달

로그는 `/aws/fis/experiment-id`라는 로그 스트림으로 전송됩니다.

로그는 1분 이내에 로그 그룹에 전달됩니다.

로그 레코드 예

다음은 무작위로 선택한 EC2 인스턴스에서 `aws:ec2:reboot-instances` 작업을 실행하는 실험의 예제 로그 기록입니다.

레코드

- [experiment-start](#)
- [target-resolution-start](#)
- [target-resolution-detail](#)
- [target-resolution-end](#)
- [action-start](#)
- [action-end](#)
- [action-error](#)
- [experiment-end](#)

experiment-start

다음은 `experiment-start` 이벤트의 예시 레코드입니다.

```
{
  "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

다음은 target-resolution-start 이벤트의 예시 레코드입니다.

```
{
  "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

다음은 target-resolution-detail 이벤트의 예시 레코드입니다. 대상 확인에 실패할 경우 기록에는 error_information 필드도 포함됩니다.

```
{
  "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

대상 확인에 실패할 경우 기록에는 error_information 필드도 포함됩니다. total_pages이 1보다 크면 확인된 대상 수가 한 레코드의 크기 제한을 초과했습니다. 나머지 확인된 대상이 포함된 추가 target-resolution-end 레코드가 있습니다.

다음은 EC2 작업에 대한 target-resolution-end 이벤트의 예시 레코드입니다.

```
{
  "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
  }
}
```

다음은 EKS 작업에 대한 target-resolution-end 이벤트의 예시 레코드입니다.

```
{
  "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

다음은 action-start 이벤트의 예시 레코드입니다. 실험 템플릿에 작업 파라미터가 지정되어 있는 경우 레코드에는 parameters 필드도 포함됩니다.

```
{
  "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-error

다음은 action-error 이벤트의 예시 레코드입니다. 작업이 실패할 때만 반환되는 이벤트입니다. 작업이 실패한 각 계정에 대해 반환됩니다.

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

action-end

다음은 action-end 이벤트의 예시 레코드입니다.

```
{
  "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."
    }
  }
}
```

experiment-end

다음은 experiment-end 이벤트의 예시 레코드입니다.

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

실험 로깅 활성화

실험 로깅은 기본적으로 비활성화되어 있습니다. 실험에 대한 실험 로그를 받으려면 로깅이 활성화된 실험 템플릿에서 실험을 생성해야 합니다. 이전에 로깅에 사용하지 않은 대상을 사용하도록 구성된 실

험을 처음 실행하면 이 대상으로의 로그 전달을 구성하기 위해 실험이 지연되며, 약 15초가 소요됩니다.

콘솔을 이용하여 실험 로그를 활성화하려면

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 업데이트를 선택합니다.
4. 로그의 경우 대상 옵션을 구성합니다. S3 버킷으로 로그를 보내려면 Amazon S3 버킷으로 전송을 선택하고 버킷 이름과 접두사를 입력합니다. 로그를 Logs로 CloudWatch 보내려면 Send to CloudWatch Logs를 선택하고 로그 그룹을 입력합니다.
5. 실험 템플릿 업데이트를 선택합니다.

AWS CLI를 사용하여 실험 로깅을 활성화하려면

[update-experiment-template](#) 명령을 사용하고 로그 구성을 지정합니다.

실험 로깅 비활성화

실험에 대한 로그를 더 이상 받지 않으려면 실험 로깅을 비활성화할 수 있습니다.

콘솔을 이용하여 실험 로깅을 비활성화하려면

1. <https://console.aws.amazon.com/fis/>에서 AWS FIS 콘솔을 엽니다.
2. 탐색 창에서 실험 템플릿을 선택합니다.
3. 실험 템플릿을 선택하고 작업, 실험 템플릿 업데이트를 선택합니다.
4. 로그의 경우 Amazon S3 버킷으로 보내기 및 CloudWatch 로그로 보내기의 선택을 취소하십시오.
5. 실험 템플릿 업데이트를 선택합니다.

AWS CLI를 이용하여 실험 로깅을 비활성화하려면

[update-experiment-template](#) 명령을 사용하여 빈 로그 구성을 지정합니다.

AWS CloudTrail을 사용하여 API 호출 로그

AWSAWSFIS (Fault Injection Service) 는 FIS에서 AWS 사용자, 역할 또는 AWS 서비스가 수행한 작업의 기록을 제공하는 서비스와 통합됩니다. AWS CloudTrail CloudTrail AWSFIS에 대한 모든 API 호

출을 이벤트로 캡처합니다. 캡처되는 호출에는 AWS FIS 콘솔에서 수행한 호출과 AWS FIS API 작업에 대한 코드 호출이 포함됩니다. 트레일을 생성하면 AWS FIS CloudTrail 이벤트를 포함하여 Amazon S3 버킷으로 이벤트를 지속적으로 전송할 수 있습니다. 트레일을 구성하지 않아도 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 계속 볼 수 있습니다. 에서 수집한 CloudTrail 정보를 사용하여 AWS FIS에 이루어진 요청, 요청이 이루어진 IP 주소, 요청한 사람, 요청 시기 및 추가 세부 정보를 확인할 수 있습니다.

자세한 CloudTrail 내용은 [AWS CloudTrail사용 설명서를](#) 참조하십시오.

사용 CloudTrail

CloudTrail 계정을 만들 AWS 계정 때 활성화됩니다. AWSFIS에서 활동이 발생하면 해당 활동이 CloudTrail 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 이벤트 [기록으로 CloudTrail 이벤트 보기를](#) 참조하십시오.

AWS FIS에 대한 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 생성합니다. 트레일을 사용하면 CloudTrail Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 지역에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 지역의 이벤트를 로깅하고 지정된 Amazon S3 버킷으로 로그 파일을 전송합니다. 또한 CloudTrail 로그에서 수집된 이벤트 데이터를 추가로 분석하고 이에 따라 조치를 취하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- [AWS 계정에 대한 추적 생성](#)
- [CloudTrail 지원되는 서비스 및 통합](#)
- [에 대한 Amazon SNS 알림 구성 CloudTrail](#)
- [여러 지역에서 CloudTrail 로그 파일 수신 및 여러 계정으로부터 CloudTrail 로그 파일 수신](#)

모든 AWS FIS 작업은 [AWS장애 주입 서비스 API](#) 참조에 의해 CloudTrail 기록되고 문서화됩니다. 대상 리소스에서 수행된 실험 작업은 리소스를 소유한 서비스의 API 참조 설명서를 참조하세요. 예를 들어, Amazon EC2 인스턴스에서 수행되는 작업은 [Amazon EC2 API 참조](#)를 참조하세요.

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에게 대한 정보가 들어 있습니다. 보안 인증 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 사용자 보안 인증으로 했는지 여부.
- 역할 또는 페더레이션 사용자에게 대한 임시 보안 인증을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지.

자세한 내용은 [CloudTrail userIdentity 요소](#)를 참조하세요.

AWS FIS 로그 파일 항목 이해

트레일은 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 전송할 수 있는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함되어 있습니다. 이벤트는 모든 소스의 단일 요청을 나타내며 요청된 작업, 작업 날짜 및 시간, 요청 매개 변수 등에 대한 정보를 포함합니다. CloudTrail 로그 파일은 공개 API 호출의 정렬된 스택 트race가 아니므로 특정 순서로 표시되지 않습니다.

다음은 AWS FIS StopExperiment 작업 호출에 대한 예제 CloudTrail 로그 항목입니다.

```
{
  "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"
}

```

다음은 FIS 작업이 포함된 실험의 일부로 AWS FIS가 호출한 API 작업에 대한 예제 CloudTrail 로그 항목입니다. `aws:ssm:send-command` AWS `userIdentity` 요소는 역할을 맡아 얻은 임시 보안 인증 정보로 이루어진 요청을 반영합니다. 위임된 역할의 이름은 `userName`에 표시됩니다. 실험의 ID인 `EXP21nT17WMzA6dnUgz`는 맡은 역할의 ARN의 일부 및 `principalId`에 나타납니다.

```

{
  "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"
}
```

AWS 장애 인젝션 서비스의 보안

클라우드 AWS 보안이 최우선 과제입니다. AWS 고객은 가장 보안에 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

보안은 기업과 귀사 간의 공동 책임입니다. AWS [공동 책임 모델](#)은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호하는 역할을 합니다. AWS 또한 안전하게 사용할 수 있는 서비스를 제공합니다. AWS Fault Injection Service에 적용되는 규정 준수 프로그램에 대해 자세히 알아보려면 규정 준수 [프로그램별 범위 내 AWS 서비스 \(규정 준수 프로그램별\)](#) 를 참조하십시오.
- 클라우드에서의 보안 — 사용하는 AWS 서비스에 따라 책임이 결정됩니다. 또한 귀하는 귀사의 데이터의 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 AWS FIS를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 항목은 보안 및 규정 준수 목표를 충족하도록 AWS FIS를 구성하는 방법을 보여줍니다. 또한 AWS FIS 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법도 알아봅니다.

내용

- [AWS 폴트 인젝션 서비스의 데이터 보호](#)
- [AWS 장애 주입 서비스의 ID 및 액세스 관리](#)
- [AWS 장애 인젝션 서비스의 인프라 보안](#)
- [인터페이스 VPC AWS 엔드포인트 \(\) 를 사용하여 FIS에 액세스 AWS PrivateLink](#)

AWS 폴트 인젝션 서비스의 데이터 보호

AWS [공동 책임 모델](#) [공동 책임 모델](#) AWS Fault Service의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 는 모든 것을 실행하는 글로벌 인프라를 보호하는 역할을 AWS 클라우드합니다. 이 인프라에서 호스팅되는 콘텐츠에 대한 제어를 유지하는 것은 사용자의 책임입니다. 사용하는 AWS 서비스의 보안 구성과 관리 작업에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 [데이터 프라이버시 FAQ](#)를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 [AWS 공동 책임 모델 및 GDPR](#) 블로그 게시물을 참조하세요.

데이터 보호를 위해 AWS 계정 자격 증명을 보호하고 AWS IAM Identity Center OR AWS Identity and Access Management (IAM) 을 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이 방식을 사용하면 각 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 리소스와 통신할 수 있습니다. AWS TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다. AWS CloudTrail
- 포함된 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용하십시오 AWS 서비스.
- Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하여 Amazon S3에 저장된 민감한 데이터를 검색하고 보호합니다.
- 명령줄 인터페이스 또는 API를 AWS 통해 액세스할 때 FIPS 140-2로 검증된 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용하십시오. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 [Federal Information Processing Standard\(FIPS\) 140-2](#)를 참조하십시오.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 양식 필드에 입력하지 않는 것이 좋습니다. 여기에는 콘솔, API 또는 SDK를 AWS 서비스 사용하여 AWS FIS 또는 기타 작업을 수행하는 경우가 포함됩니다. AWS CLI AWS 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함시켜서는 안 됩니다.

저장된 데이터 암호화

AWS FIS는 저장된 데이터를 항상 암호화합니다. AWS FIS의 데이터는 투명한 서버 측 암호화를 사용하여 유휴 상태에서 암호화됩니다. 이를 사용하면 중요한 데이터 보호와 관련된 운영 부담 및 복잡성을 줄일 수 있습니다. 유휴 시 암호화를 사용하면 암호화 규정 준수 및 규제 요구 사항이 필요한, 보안에 민감한 애플리케이션을 구축할 수 있습니다.

전송 중 암호화

AWS FIS는 서비스와 다른 통합 서비스 간에 전송되는 데이터를 암호화합니다. AWS FIS와 통합 서비스 간에 전달되는 모든 데이터는 전송 계층 보안 (TLS) 을 사용하여 암호화됩니다. 기타 통합 AWS 서비스에 대한 자세한 내용은 [지원되는 AWS 서비스](#) 을 참조하십시오.

AWS 장애 주입 서비스의 ID 및 액세스 관리

AWS Identity and Access Management (IAM) 은 관리자가 리소스에 대한 액세스를 안전하게 제어할 수 있는 AWS 서비스 있도록 도와줍니다. AWS IAM 관리자는 FIS 리소스를 사용할 수 있는 인증 (로그인) 및 권한 부여 (권한 보유) 를 받을 수 있는 사용자를 제어합니다. IAM은 추가 비용 없이 사용할 수 있습니다.

내용

- [고객](#)
- [자격 증명을 통한 인증](#)
- [정책을 사용한 액세스 관리](#)
- [AWS 장애 인젝션 서비스가 IAM과 함께 작동하는 방식](#)
- [AWS 장애 주입 서비스 정책 예제](#)
- [장애 인젝션 서비스에 서비스 연결 역할을 사용하십시오. AWS](#)
- [AWS 장애 주입 서비스에 대한 AWS 관리형 정책](#)

고객

FIS에서 수행하는 작업에 따라 사용 방법 AWS Identity and Access Management (IAM) 이 다릅니다. AWS

서비스 사용자 - AWS FIS 서비스를 사용하여 작업을 수행하는 경우 관리자가 필요한 자격 증명과 권한을 제공합니다. 더 많은 AWS FIS 기능을 사용하여 작업을 수행함에 따라 추가 권한이 필요할 수 있습니다. 액세스 권한 관리 방식을 이해하면 적절한 권한을 관리자에게 요청할 수 있습니다.

서비스 관리자 — 회사에서 FIS 리소스를 담당하는 경우 AWS FIS에 대한 전체 액세스 권한이 있을 것입니다. AWS 서비스 사용자가 액세스해야 하는 AWS FIS 기능 및 리소스를 결정하는 것은 여러분의 몫입니다. 그런 다음, IAM 관리자에게 요청을 제출하여 서비스 사용자의 권한을 변경해야 합니다. 이 페이지의 정보를 검토하여 IAM의 기본 개념을 이해하십시오.

IAM 관리자 — IAM 관리자라면 FIS 액세스를 관리하기 위한 정책을 작성하는 방법에 대해 자세히 알고 싶을 것입니다. AWS

자격 증명을 통한 인증

인증은 ID 자격 증명을 AWS 사용하여 로그인하는 방법입니다. IAM 사용자로 인증 (로그인 AWS) 하거나 IAM 역할을 맡아 인증 (로그인) 해야 합니다. AWS 계정 루트 사용자

ID 소스를 통해 제공된 자격 증명을 사용하여 페더레이션 ID로 로그인할 수 있습니다. AWS IAM Identity Center (IAM ID 센터) 사용자, 회사의 싱글 사인온 인증, Google 또는 Facebook 자격 증명이 페더레이션 ID의 예입니다. 연합형 ID로 로그인할 때 관리자가 이전에 IAM 역할을 사용하여 ID 연합을 설정했습니다. 페더레이션을 사용하여 액세스하는 경우 AWS 간접적으로 역할을 맡게 됩니다.

사용자 유형에 따라 AWS Management Console 또는 AWS 액세스 포털에 로그인할 수 있습니다. 로그인에 대한 자세한 내용은 AWS 로그인 사용 설명서의 [내 로그인 방법을](#) 참조하십시오. AWS 계정

AWS 프로그래밍 방식으로 액세스하는 경우 자격 증명을 사용하여 요청에 암호화 방식으로 서명할 수 있는 소프트웨어 개발 키트 (SDK)와 명령줄 인터페이스 (CLI)를 AWS 제공합니다. AWS 도구를 사용하지 않는 경우 요청에 직접 서명해야 합니다. 권장 방법을 사용하여 직접 요청에 서명하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 AWS [API 요청 서명을](#) 참조하십시오.

사용하는 인증 방법에 상관없이 추가 보안 정보를 제공해야 할 수도 있습니다. 예를 들어, AWS 계정의 보안을 강화하기 위해 다단계 인증 (MFA)을 사용할 것을 권장합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 [다중 인증](#) 및 IAM 사용 설명서의 [AWS에서 다중 인증\(MFA\) 사용](#)을 참조하십시오.

AWS 계정 루트 사용자

계정을 AWS 계정만들 때는 먼저 계정의 모든 AWS 서비스 리소스에 대한 완전한 액세스 권한을 가진 하나의 로그인 ID로 시작합니다. 이 ID를 AWS 계정 루트 사용자라고 하며, 계정을 만들 때 사용한 이메일 주소와 비밀번호로 로그인하여 액세스할 수 있습니다. 일상적인 작업에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 보안 인증 정보를 보호하고 루트 사용자만 수행할 수 있는 작업을 수행하는 데 사용합니다. 루트 사용자로 로그인해야 하는 태스크의 전체 목록은 IAM 사용 설명서의 [루트 사용자 보안 인증이 필요한 태스크](#)를 참조하세요.

연동 보안 인증

가장 좋은 방법은 관리자 액세스가 필요한 사용자를 비롯한 수동 AWS 서비스 사용자가 ID 공급자와의 페더레이션을 사용하여 임시 자격 증명을 사용하여 액세스하도록 하는 것입니다.

페더레이션 ID는 기업 사용자 디렉토리, 웹 ID 공급자, Identity Center 디렉터리의 사용자 또는 ID 소스를 통해 제공된 자격 증명을 사용하여 액세스하는 AWS 서비스 모든 사용자를 말합니다. AWS Directory Service 페더레이션 ID에 AWS 계정 액세스하면 이들이 역할을 맡고 역할은 임시 자격 증명을 제공합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center을 사용하는 것이 좋습니다. IAM Identity Center에서 사용자 및 그룹을 생성하거나 자체 ID 소스의 사용자 및 그룹 집합에 연결하고 동기화하여 모든 사용자 및 애플리케이션에서 사용할 수 있습니다. AWS 계정 IAM Identity Center에 대한 자세한

내용은 AWS IAM Identity Center 사용 설명서에서 [IAM Identity Center란 무엇입니까?](#)를 참조하십시오.

IAM 사용자 및 그룹

[IAM 사용자는 단일 사용자](#) 또는 애플리케이션에 대한 특정 권한을 AWS 계정 가진 사용자 내 자격 증명입니다. 가능하면 암호 및 액세스 키와 같은 장기 자격 증명이 있는 IAM 사용자를 생성하는 대신 임시 자격 증명을 사용하는 것이 좋습니다. 하지만 IAM 사용자의 장기 자격 증명이 필요한 특정 사용 사례가 있는 경우 액세스 키를 교체하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 [장기 보안 인증이 필요한 사용 사례의 경우 정기적으로 액세스 키 교체](#)를 참조하십시오.

[IAM 그룹](#)은 IAM 사용자 컬렉션을 지정하는 자격 증명입니다. 사용자는 그룹으로 로그인할 수 없습니다. 그룹을 사용하여 여러 사용자의 권한을 한 번에 지정할 수 있습니다. 그룹을 사용하면 대규모 사용자 집합의 권한을 더 쉽게 관리할 수 있습니다. 예를 들어, IAMAdmins라는 그룹이 있고 이 그룹에 IAM 리소스를 관리할 권한을 부여할 수 있습니다.

사용자는 역할과 다릅니다. 사용자는 한 사람 또는 애플리케이션과 고유하게 연결되지만, 역할은 해당 역할이 필요한 사람이라면 누구나 수입할 수 있습니다. 사용자는 영구적인 장기 보안 인증 정보를 가지고 있지만, 역할은 임시 보안 인증 정보만 제공합니다. 자세한 정보는 IAM 사용 설명서의 [IAM 사용자를 만들어야 하는 경우\(역할이 아님\)](#)를 참조하십시오.

IAM 역할

[IAM 역할](#)은 특정 권한을 가진 사용자 AWS 계정 내의 자격 증명입니다. IAM 사용자와 유사하지만, 특정 개인과 연결되지 않습니다. 역할을 AWS Management Console [전환하여](#) 에서 일시적으로 IAM 역할을 맡을 수 있습니다. AWS CLI 또는 AWS API 작업을 호출하거나 사용자 지정 URL을 사용하여 역할을 수입할 수 있습니다. 역할 사용 방법에 대한 자세한 정보는 IAM 사용 설명서의 [IAM 역할 사용](#)을 참조하십시오.

임시 보안 인증이 있는 IAM 역할은 다음과 같은 상황에서 유용합니다:

- 연합 사용자 액세스 - 연합 아이덴티티에 권한을 부여하려면 역할을 생성하고 해당 역할의 권한을 정의합니다. 연합 아이덴티티가 인증되면 역할이 연결되고 역할에 정의된 권한이 부여됩니다. 페더레이션 역할에 대한 자세한 내용은 IAM 사용 설명서의 [타사 자격 증명 공급자의 역할 생성](#)을 참조하십시오. IAM Identity Center를 사용하는 경우 권한 집합을 구성합니다. 인증 후 아이덴티티가 액세스할 수 있는 항목을 통제하기 위해 IAM Identity Center는 권한 세트를 IAM의 역할과 상관짓습니다. 권한 세트에 대한 자세한 내용은 AWS IAM Identity Center 사용 설명서의 [권한 세트](#)를 참조하십시오.
- 임시 IAM 사용자 권한 - IAM 사용자 또는 역할은 IAM 역할을 수입하여 특정 작업에 대한 다양한 권한을 임시로 받을 수 있습니다.

- 크로스 계정 액세스 - IAM 역할을 사용하여 다른 계정의 사용자(신뢰할 수 있는 보안 주체)가 내 계정의 리소스에 액세스하도록 허용할 수 있습니다. 역할은 계정 간 액세스를 부여하는 기본적인 방법입니다. 그러나 일부 AWS 서비스 경우에는 역할을 프록시로 사용하는 대신 정책을 리소스에 직접 연결할 수 있습니다. 교차 계정 액세스를 위한 역할과 리소스 기반 정책의 차이점을 알아보려면 IAM 사용 설명서의 [IAM 역할과 리소스 기반 정책의 차이](#)를 참조하십시오.
- 서비스 간 액세스 — 일부는 다른 AWS 서비스서비스의 기능을 AWS 서비스 사용합니다. 예컨대, 어떤 서비스에서 호출을 수행하면 일반적으로 해당 서비스는 Amazon EC2에서 애플리케이션을 실행하거나 Amazon S3에 객체를 저장합니다. 서비스는 호출하는 보안 주체의 권한을 사용하거나, 서비스 역할을 사용하거나, 또는 서비스 연결 역할을 사용하여 이 작업을 수행할 수 있습니다.
- 순방향 액세스 세션 (FAS) — IAM 사용자 또는 역할을 사용하여 작업을 수행하는 경우 보안 AWS 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 AWS 서비스서비스에 AWS 서비스 요청하기 위한 요청과 결합하여 사용합니다. FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 [전달 액세스 세션](#)을 참조하세요.
- 서비스 역할 - 서비스 역할은 서비스가 사용자를 대신하여 태스크를 수행하기 위해 맡는 [IAM 역할](#)입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 [역할을 생성하여 AWS 서비스에게 권한 위임](#)을 참조하십시오.
- 서비스 연결 역할 — 서비스 연결 역할은 에 연결된 서비스 역할의 한 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.
- Amazon EC2에서 실행되는 애플리케이션 — IAM 역할을 사용하여 EC2 인스턴스에서 실행되고 API 요청을 AWS CLI 하는 애플리케이션의 임시 자격 증명을 관리할 수 있습니다. AWS 이는 EC2 인스턴스 내에 액세스 키를 저장할 때 권장되는 방법입니다. EC2 인스턴스에 AWS 역할을 할당하고 모든 애플리케이션에서 사용할 수 있게 하려면 인스턴스에 연결된 인스턴스 프로필을 생성합니다. 인스턴스 프로파일에는 역할이 포함되어 있으며 EC2 인스턴스에서 실행되는 프로그램이 임시 보안 인증 정보를 얻을 수 있습니다. 자세한 정보는 IAM 사용 설명서의 [IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여](#)를 참조하십시오.

IAM 역할을 사용할지 또는 IAM 사용자를 사용할지를 알아보려면 [IAM 사용 설명서](#)의 IAM 역할(사용자 대신)을 생성하는 경우를 참조하십시오.

정책을 사용한 액세스 관리

정책을 생성하고 이를 AWS ID 또는 리소스에 AWS 연결하여 액세스를 제어할 수 있습니다. 정책은 ID 또는 리소스와 연결될 때 AWS 해당 권한을 정의하는 객체입니다. AWS 주도자 (사용자, 루트 사용자 또는 역할 세션) 가 요청할 때 이러한 정책을 평가합니다. 정책의 권한이 요청 허용 또는 거부 여부를 결정합니다. 대부분의 정책은 JSON 문서로 AWS 저장됩니다. JSON 정책 문서의 구조와 콘텐츠에 대한 자세한 정보는 IAM 사용 설명서의 [JSON 정책 개요](#)를 참조하십시오.

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

기본적으로, 사용자와 역할에는 어떠한 권한도 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 수임할 수 있습니다.

IAM 정책은 작업을 수행하기 위해 사용하는 방법과 상관없이 작업에 대한 권한을 정의합니다. 예를 들어, iam:GetRole태스크를 허용하는 정책이 있다고 가정합니다. 해당 정책을 사용하는 사용자는 AWS Management Console, AWS CLI, 또는 AWS API에서 역할 정보를 가져올 수 있습니다.

보안 인증 기반 정책

보안 인증 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 보안 인증에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 [IAM 정책 생성](#)을 참조하십시오.

자격 증명 기반 정책은 인라인 정책 또는 관리형 정책으로 한층 더 분류할 수 있습니다. 인라인 정책은 단일 사용자, 그룹 또는 역할에 직접 포함됩니다. 관리형 정책은 내 여러 사용자, 그룹 및 역할에 연결할 수 있는 독립형 정책입니다. AWS 계정관리형 정책에는 AWS 관리형 정책과 고객 관리형 정책이 포함됩니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 [관리형 정책과 인라인 정책의 선택](#)을 참조하십시오.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 [보안 주체를 지정](#)해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. IAM의 AWS 관리형 정책은 리소스 기반 정책에 사용할 수 없습니다.

액세스 제어 목록(ACL)

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 설명서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ACL을 지원하는 서비스의 예로는 아마존 S3와 아마존 VPC가 있습니다. AWS WAF ACL에 대해 자세히 알아보려면 Amazon Simple Storage Service 개발자 안내서의 [ACL\(액세스 제어 목록\) 개요](#)를 참조하십시오.

기타 정책 타입

AWS 일반적이지 않은 추가 정책 유형을 지원합니다. 이러한 정책 타입은 더 일반적인 정책 타입에 따라 사용자에게 부여되는 최대 권한을 설정할 수 있습니다.

- 권한 경계 – 권한 경계는 자격 증명 기반 정책에 따라 IAM 엔티티(IAM 사용자 또는 역할)에 부여할 수 있는 최대 권한을 설정하는 고급 기능입니다. 개체에 대한 권한 경계를 설정할 수 있습니다. 그 결과로 얻는 권한은 개체의 보안 인증 기반 정책과 그 권한 경계의 교집합입니다. Principal 필드에서 사용자나 역할을 보안 주체로 지정하는 리소스 기반 정책은 권한 경계를 통해 제한되지 않습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 권한 경계에 대한 자세한 정보는 IAM 사용 설명서의 [IAM 엔티티에 대한 권한 경계](#)를 참조하십시오.
- 서비스 제어 정책 (SCP) - SCP는 조직 또는 조직 단위 (OU) 에 대한 최대 권한을 지정하는 JSON 정책입니다. AWS Organizations AWS Organizations 사업체가 소유한 여러 AWS 계정 개를 그룹화하고 중앙에서 관리하는 서비스입니다. 조직에서 모든 특성을 활성화할 경우 서비스 제어 정책 (SCP)을 임의의 또는 모든 계정에 적용할 수 있습니다. SCP는 구성원 계정의 엔티티 (각 엔티티 포함) 에 대한 권한을 제한합니다. AWS 계정 루트 사용자조직 및 SCP에 대한 자세한 정보는 AWS Organizations 사용 설명서의 [SCP 작동 방식](#)을 참조하십시오.
- 세션 정책 – 세션 정책은 역할 또는 페더레이션 사용자에게 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 결과적으로 얻는 세션의 권한은 사용자 또는 역할의 자격 증명 기반 정책의 교집합과 세션 정책입니다. 또한 권한을 리소스 기반 정책에서 가져올 수도 있습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 자세한 정보는 IAM 사용 설명서의 [세션 정책](#)을 참조하십시오.

여러 정책 타입

여러 정책 타입이 요청에 적용되는 경우 결과 권한은 이해하기가 더 복잡합니다. 여러 정책 유형이 관련된 경우 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 [정책 평가 로직](#)을 참조하십시오.

AWS 장애 인젝션 서비스가 IAM과 함께 작동하는 방식

IAM을 사용하여 FIS 액세스를 관리하기 전에 AWS FIS에서 사용할 수 있는 IAM 기능에 대해 알아보십시오. AWS

장애 주입 서비스와 함께 사용할 수 있는 IAM 기능 AWS

IAM 특성	AWS FIS 지원
ID 기반 정책	예
리소스 기반 정책	아니요
정책 작업	예
정책 리소스	예
정책 조건 키(서비스별)	예
ACL	아니요
ABAC(정책의 태그)	예
임시 보안 인증	예
보안 주체 권한	예
서비스 역할	예
서비스 연결 역할	예

AWS FIS 및 기타 AWS 서비스가 대부분의 IAM 기능과 어떻게 작동하는지 자세히 알아보려면 IAM 사용 설명서의 [IAM과 함께 작동하는 AWS 서비스를](#) 참조하십시오.

AWS FIS의 ID 기반 정책

ID 기반 정책 지원

예

자격 증명기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 [IAM 정책 생성](#)을 참조하십시오.

IAM ID 기반 정책을 사용하면 허용되거나 거부되는 태스크와 리소스 뿐만 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. 자격 증명 기반 정책에서는 보안 주체가 연결된 사용자 또는 역할에 적용되므로 보안 주체를 지정할 수 없습니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 [IAM JSON 정책 요소 참조](#)를 참조하십시오.

FIS의 ID 기반 정책 예제 AWS

AWS FIS ID 기반 정책의 예를 보려면 [AWS 장애 주입 서비스 정책 예제](#)를 참조하십시오.

FIS 내의 리소스 기반 정책 AWS

리소스 기반 정책 지원

아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 문서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 [보안 주체를 지정](#)해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

크로스 계정 액세스를 활성화하려는 경우 전체 계정이나 다른 계정의 IAM 개체를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 리소스 기반 정책에 크로스 계정 보안 주체를 추가하는 것은 트러스트 관계 설정의 절반밖에 되지 않는다는 것을 유념하십시오. 보안 주체와 리소스가 다른 AWS 계정경우 신뢰할 수 있는 계정의 IAM 관리자는 보안 주체 개체 (사용자 또는 역할)에게 리소스에 액세스할 수 있는 권한도 부여해야 합니다. 개체에 자격 증명 기반 정책을 연결하여 권한을 부여합니다. 하지만 리소스 기반 정책이 동일 계정의 보안 주체에 액세스를 부여하는 경우 추가 자격 증명 기반 정책이 필요하지 않습니다. 자세한 정보는 IAM 사용 설명서의 [IAM 역할과 리소스 기반 정책의 차이](#)를 참조하십시오.

FIS를 위한 AWS 정책 조치

정책 작업 지원

예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 Action요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 태스크를 설명합니다. 정책 작업은 일반적으로 관련 AWS API 작업과 이름이 같습니다. 일치하는 API 작업이 없는 권한 전용 작업 같은 몇 가지 예외도 있습니다. 정책에서 여러 작업이 필요한 몇 가지 작업도 있습니다. 이러한 추가 작업을 종속 작업이라고 합니다.

연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하십시오.

AWS FIS 조치 목록을 보려면 서비스 권한 부여 참조의 AWS [장애 주입 서비스에서 정의한 작업을](#) 참조하십시오.

AWS FIS의 정책 조치는 조치 앞에 다음 접두사를 사용합니다.

```
fis
```

단일 문에서 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
  "fis:action1",
  "fis:action2"
]
```

와일드카드(*)를 사용하여 여러 작업을 지정할 수 있습니다. 예를 들어, List(이)라는 단어로 시작하는 모든 작업을 지정하려면 다음 작업을 포함합니다.

```
"Action": "fis:List*"
```

FIS 정책 리소스 AWS

정책 리소스 지원

예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

Resource JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 보고서에는 Resource 또는 NotResource 요소가 반드시 추가되어야 합니다. 모범 사례에 따라 [Amazon 리소스 이름\(ARN\)](#)을 사용하여 리소스를 지정합니다. 리소스 수준 권한이라고 하는 특정 리소스 유형을 지원하는 작업에 대해 이 작업을 수행할 수 있습니다.

작업 나열과 같이 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"

```

일부 AWS FIS API 작업은 여러 리소스를 지원합니다. 단일 문에서 여러 리소스를 지정하려면 ARN을 쉼표로 구분합니다.

```
"Resource": [
  "resource1",
  "resource2"
]

```

AWS FIS 리소스 유형 및 해당 ARN 목록을 보려면 서비스 권한 부여 참조의 [AWS 장애 주입 서비스에서 정의한 리소스 유형](#)을 참조하십시오. 각 리소스의 ARN을 지정할 수 있는 작업에 대해 알아보려면 [AWS 장애 삽입 서비스에서 정의한 작업](#)을 참조하십시오.

FIS의 정책 조건 키 AWS

서비스별 정책 조건 키 지원

예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

Condition 요소(또는 Condition블록)를 사용하면 정책이 발효되는 조건을 지정할 수 있습니다. Condition 요소는 옵션입니다. 같거나 적음 같은 [조건 연산자](#)를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다.

한 문에서 여러 Condition요소를 지정하거나 단일 Condition요소에서 여러 키를 지정하는 경우 AWS 는 논리적 AND태스크를 사용하여 평가합니다. 단일 조건 키에 여러 값을 지정하는 경우는 논리

적 OR 연산을 사용하여 조건을 AWS 평가합니다. 명령문의 권한을 부여하기 전에 모든 조건을 충족해야 합니다.

조건을 지정할 때 자리 표시자 변수를 사용할 수도 있습니다. 예를 들어, IAM 사용자에게 IAM 사용자 이름으로 태그가 지정된 경우에만 리소스에 액세스할 수 있는 권한을 부여할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 [IAM 정책 요소: 변수 및 태그](#)를 참조하십시오.

AWS 글로벌 조건 키 및 서비스별 조건 키를 지원합니다. 모든 AWS 글로벌 조건 키를 보려면 IAM 사용 [AWS 설명서의 글로벌 조건 컨텍스트 키](#)를 참조하십시오.

AWS FIS 조건 키 목록을 보려면 [서비스 권한 부여 참조의 AWS 장애 주입 서비스를 위한 조건 키](#)를 참조하십시오. 조건 키를 사용할 수 있는 작업 및 리소스에 대해 알아보려면 [AWS 오류 삽입 서비스에서 정의한 작업을](#) 참조하십시오.

AWS FIS ID 기반 정책의 예를 보려면 [AWS 장애 주입 서비스 정책 예제](#)를 참조하십시오.

FIS의 ACL AWS

ACL 지원	아니요
--------	-----

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 설명서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ABAC (FIS 포함) AWS

ABAC 지원(정책의 태그)	예
-----------------	---

ABAC(속성 기반 액세스 제어)는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. AWS에서는 이러한 속성을 태그라고 합니다. IAM 엔티티(사용자 또는 역할) 및 여러 AWS 리소스에 태그를 첨부할 수 있습니다. ABAC의 첫 번째 단계로 개체 및 리소스에 태그를 지정합니다. 그런 다음 보안 주체의 태그가 액세스하려는 리소스의 태그와 일치할 때 작업을 허용하도록 ABAC 정책을 설계합니다.

ABAC는 빠르게 성장하는 환경에서 유용하며 정책 관리가 번거로운 상황에 도움이 됩니다.

태그를 기반으로 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` 또는 `aws:TagKeys` 조건 키를 사용하여 정책의 [조건 요소](#)에 태그 정보를 제공합니다.

서비스가 모든 리소스 유형에 대해 세 가지 조건 키를 모두 지원하는 경우 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 유형에 대해서만 세 가지 조건 키를 모두 지원하는 경우 값은 부분입니다.

ABAC에 대한 자세한 정보는 IAM 사용 설명서의 [ABAC란 무엇인가요?](#)를 참조하십시오. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용 설명서의 [속성 기반 액세스 제어\(ABAC\) 사용](#)을 참조하십시오.

리소스의 태그를 기반으로 리소스에 대한 액세스를 제한하는 자격 증명 기반 정책의 예제는 [예: 태그를 사용하여 리소스 사용 제어](#)에서 확인할 수 있습니다.

FIS에서 임시 자격 증명 사용 AWS

임시 보안 인증 정보 지원

예

임시 자격 증명을 사용하여 로그인하면 일부 자격 증명에 AWS 서비스 작동하지 않습니다. 임시 자격 증명을 사용하는 방법을 AWS 서비스 비롯한 추가 정보는 [IAM 사용 설명서의 IAM과 AWS 서비스 연동되는](#) 내용을 참조하십시오.

사용자 이름과 암호를 제외한 다른 방법을 AWS Management Console 사용하여 로그인하면 임시 자격 증명을 사용하는 것입니다. 예를 들어 회사의 SSO (Single Sign-On) 링크를 AWS 사용하여 액세스하는 경우 이 프로세스에서 자동으로 임시 자격 증명을 생성합니다. 또한 콘솔에 사용자로 로그인한 다음 역할을 전환할 때 임시 보안 인증 정보를 자동으로 생성합니다. 역할 전환에 대한 자세한 정보는 IAM 사용 설명서의 [역할로 전환\(콘솔\)](#)을 참조하십시오.

또는 API를 사용하여 임시 자격 증명을 수동으로 생성할 수 있습니다 AWS CLI . AWS 그런 다음 해당 임시 자격 증명을 사용하여 액세스할 수 AWS 있습니다. AWS 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성할 것을 권장합니다. 자세한 정보는 [IAM의 임시 보안 자격 증명](#) 섹션을 참조하십시오.

FIS에 대한 서비스 간 주체 권한 AWS

전달 액세스 세션(FAS) 지원

예

IAM 사용자 또는 역할을 사용하여 작업을 수행하는 AWS 경우 보안 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 서비스에 AWS 서비스 요청하라는 요청과 결합하여 사용합니다. AWS 서비스 FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신

한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 [전달 액세스 세션](#)을 참조하세요.

AWS FIS에 대한 서비스 역할

서비스 역할 지원	예
-----------	---

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 [IAM 역할](#)입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 [역할을 생성하여 AWS 서비스에게 권한 위임](#)을 참조하십시오.

FIS의 서비스 연결 역할 AWS

서비스 링크 역할 지원	예
--------------	---

서비스 연결 역할은 에 연결된 서비스 역할 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수입할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

AWS FIS 서비스 연결 역할을 만들거나 관리하는 방법에 대한 자세한 내용은 을 참조하십시오. [장애 인젝션 서비스에 서비스 연결 역할을 사용하십시오. AWS](#)

AWS 장애 주입 서비스 정책 예제

기본적으로 사용자 및 역할에는 AWS FIS 리소스를 만들거나 수정할 권한이 없습니다. 또한 AWS Management Console, AWS Command Line Interface (AWS CLI) 또는 AWS API를 사용하여 작업을 수행할 수 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 맡을 수 있습니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 [IAM 정책 생성](#)을 참조하십시오.

각 리소스 유형의 ARN 형식을 비롯하여 AWS FIS에서 정의한 작업 및 리소스 유형에 대한 자세한 내용은 [서비스 권한 부여 참조의 AWS 장애 삽입 서비스를 위한 작업, 리소스 및 조건 키](#)를 참조하십시오.

내용

- [정책 모범 사례](#)
- [예: FIS 콘솔 사용 AWS](#)
- [예: 사용 가능한 AWS FIS 작업 목록](#)
- [예: 특정 작업을 위한 실험 템플릿 만들기](#)
- [예: 실험 시작](#)
- [예: 태그를 사용하여 리소스 사용 제어](#)
- [예: 특정 태그가 포함된 실험 템플릿 삭제](#)
- [예: 사용자가 자신의 권한을 볼 수 있도록 허용](#)
- [예: ec2:InjectApiError의 조건 키 사용](#)
- [예: aws:s3:bucket-pause-replication의 조건 키 사용](#)
- [예: aws:dynamodb:encrypted-global-table-pause-replication 실행 권한이 있는 실험 역할](#)

정책 모범 사례

ID 기반 정책은 누군가가 사용자 계정에서 AWS FIS 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부를 결정합니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. 자격 증명 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르십시오.

- AWS 관리형 정책으로 시작하여 최소 권한 권한으로 이동 — 사용자와 워크로드에 권한을 부여하려면 여러 일반적인 사용 사례에 권한을 부여하는 AWS 관리형 정책을 사용하세요. 해당 내용은 에서 사용할 수 있습니다. AWS 계정사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 더 줄이는 것이 좋습니다. 자세한 정보는 IAM 사용 설명서의 [AWS 관리형 정책](#) 또는 [직무에 대한 AWS 관리형 정책](#)을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서의 [IAM의 정책 및 권한](#)을 참조하세요.
- Use conditions in IAM policies to further restrict access(IAM 정책의 조건을 사용하여 액세스 추가 제한) – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어 SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 생성할 수 있습니다. 예를 AWS 서비스들어 특정 작업을 통해 서비스 작업을 사용하는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 AWS CloudFormation있습니다. 자세한 정보는 IAM 사용 설명서의 [IAM JSON 정책 요소: 조건](#)을 참조하십시오.

- IAM Access Analyzer를 통해 IAM 정책을 검증하여 안전하고 기능적인 권한 보장 – IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 신규 및 기존 정책을 검증합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 권장 사항을 제공하여 안전하고 기능적인 정책을 생성하도록 돕습니다. 자세한 정보는 IAM 사용 설명서의 [IAM Access Analyzer 정책 검증](#)을 참조하세요.
- 멀티 팩터 인증 (MFA) 필요 - IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 AWS 계정 MFA를 활성화하십시오. API 작업을 직접적으로 호출할 때 MFA가 필요하다면 정책에 MFA 조건을 추가합니다. 자세한 정보는 IAM 사용 설명서의 [MFA 보호 API 액세스 구성](#)을 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 [IAM의 보안 모범 사례](#)를 참조하십시오.

예: FIS 콘솔 사용 AWS

AWS 장애 주입 서비스 콘솔에 액세스하려면 최소 권한 집합이 있어야 합니다. 이러한 권한을 통해 자신의 AWS FIS 리소스에 대한 세부 정보를 나열하고 볼 수 AWS 계정 있어야 합니다. 최소 필수 권한보다 더 제한적인 ID 기반 정책을 만들면 콘솔이 해당 정책에 연결된 개체(사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 AWS API만 호출하는 사용자에게 최소 콘솔 권한을 허용할 필요는 없습니다. 그 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

다음 예제 정책은 FIS 콘솔을 사용하여 모든 AWS FIS 리소스를 나열하고 볼 수 있는 권한을 부여하지만 생성, 업데이트 또는 AWS 삭제는 허용하지 않습니다. 또한 실험 템플릿에서 지정할 수 있는 모든 AWS FIS 작업에 사용되는 사용 가능한 리소스를 볼 수 있는 권한을 부여합니다.

```
{
  "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"
      }
    }
  }
]
}

```

예: 사용 가능한 AWS FIS 작업 목록

다음 정책은 사용 가능한 AWS FIS 작업을 나열할 권한을 부여합니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "fis:ListActions"
      ],
      "Resource": "arn:aws:fis:*:*:action/*"
    }
  ]
}

```

```
]
}
```

예: 특정 작업을 위한 실험 템플릿 만들기

다음 정책은 `aws:ec2:stop-instances` 작업에 사용할 실험 템플릿을 만들 수 있는 권한을 부여합니다.

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

예: 실험 시작

다음 정책은 지정된 IAM 역할 및 실험 템플릿을 사용하여 실험을 시작할 수 있는 권한을 부여합니다. 또한 AWS FIS가 사용자를 대신하여 서비스 연결 역할을 생성할 수 있습니다. 자세한 정보는 [장애 인젝션 서비스에 서비스 연결 역할을 사용하십시오. AWS](#)를 참조하세요.

```
{
```



```

"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"
      }
    }
  }
]
}

```

예: 태그를 사용하여 리소스 사용 제어

다음 정책은 Purpose=Test 태그가 있는 실험 템플릿에서 실험을 실행할 권한을 부여합니다. 실험 템플릿을 생성 또는 수정하거나 지정된 태그가 없는 템플릿을 사용하여 실험을 실행할 수 있는 권한은 부여하지 않습니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "fis:StartExperiment",
      "Resource": "arn:aws:fis:*:*:experiment-template/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Purpose": "Test"
        }
      }
    }
  ]
}

```

```

    }
  }
}

```

예: 특정 태그가 포함된 실험 템플릿 삭제

다음 정책은 Purpose=Test 태그가 있는 실험 템플릿을 삭제할 권한을 부여합니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "fis:DeleteExperimentTemplate"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Purpose": "Test"
        }
      }
    }
  ]
}

```

예: 사용자가 자신의 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 자격 증명에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 API를 사용하여 프로그래밍 방식으로 이 작업을 완료할 수 있는 권한이 포함됩니다. AWS CLI `AWS`

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",

```

```

        "iam:ListGroupsWithUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
    ],
    "Resource": ["arn:aws:iam::*:user/${aws:username}"]
},
{
    "Sid": "NavigateInConsole",
    "Effect": "Allow",
    "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
    ],
    "Resource": "*"
}
]
}

```

예: **ec2:InjectApiError**의 조건 키 사용

다음 예제 정책은 `ec2:FisTargetArns` 조건 키를 사용하여 대상 리소스의 범위를 지정합니다. 이 정책은 AWS FIS 작업 `aws:ec2:api-insufficient-instance-capacity-error` 및 `aws:ec2:asg-insufficient-instance-capacity-error`를 허용합니다.

```

{
    "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": "*"
}
]
}

```

예: **aws:s3:bucket-pause-replication**의 조건 키 사용

다음 예제 정책은 S3:IsReplicationPauseRequest 조건 키를 사용하여 AWS FIS가 FIS 작업의 컨텍스트에서 허용하는 PutReplicationConfiguration GetReplicationConfiguration 경우에만 허용합니다. AWS **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": "*"
  }
]
}

```

예: **aws:dynamodb:encrypted-global-table-pause-replication** 실행 권한이 있는 실험 역할

다음 예제 정책은 단일 작업으로 실험을 실행하는 데 필요한 권한을 AWS FIS에 부여합니다.

aws:dynamodb:encrypted-global-table-pause-replication

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DynamoDB",
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:DescribeGlobalTable"
      ],
      "Resource": [
        "arn:aws:dynamodb:us-east-2:123456789012:table/MyEncryptedGlobalTable"
      ],
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/fis-enabled": "true"
        }
      }
    },
    {
      "Sid": "Tagging",
      "Effect": "Allow",
      "Action": [
        "tag:GetResources"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "KMS",
      "Effect": "Allow",
      "Action": [
        "kms:PutKeyPolicy",
        "kms:DescribeKey",
        "kms:GetKeyPolicy"
      ],
      "Resource": "arn:aws:kms:us-east-2:123456789012:key/MyGlobalTableEncryptionKey"
    },
    {
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/fis-enabled": "true"
        }
      }
    }
  ]
}
```

```

    }
  }
]
}

```

Note

AWS FIS는 고객 관리 AWS KMS 키에 대한 DynamoDB 액세스를 거부하는 `kms:PutKeyPolicy` 데 사용합니다. 그러면 복제가 중지됩니다. 이 작업으로 실험을 적극적으로 실행하는 경우에만 역할을 사용하는 것이 좋으며, 그렇지 않은 경우에는 삭제하는 것이 좋습니다. 역할을 삭제하면 `kms:PutKeyPolicy`에 대한 FIS 권한이 제거됩니다. 실험이 완료된 후 실험 템플릿 세부 정보에서 역할을 찾습니다. IAM 콘솔에서 IAM 역할에 대한 링크를 선택하고 삭제를 선택합니다. 역할을 삭제한 후 AWS KMS 콘솔로 이동하여 대상 DynamoDB 테이블에서 데이터를 보호하는 데 사용되는 AWS KMS 키를 찾습니다. AWS KMS 키 정책이 예상과 일치하는지 확인하십시오. AWS FIS 명세서 (예:) 가 더 이상 표시되지 않을 것입니다. `FIS_DDB_PAUSE_REPLICATION-EXP123456789012345_DO_NOT_MODIFY`

FIS `aws:dynamodb:encrypted-global-table-pause-replication` 작업은 대상 DynamoDB 글로벌 테이블의 데이터를 보호하는 데 사용되는 KMS 키에 대한 정책에 다음 사용 권한을 동적으로 추가합니다:

```

{
  "Sid": "DO_NOT_MODIFY_FIS_DDB_PAUSE_REPLICATION-EXP123456789012345",
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::123456789012:role/aws-service-role/replication.dynamodb.amazonaws.com/AWSServiceRoleForDynamoDBReplication"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:aws:dynamodb:tableName": [
        "transactions-global-table",

```

```

        "inventory-global-table"
    ]
}
}
}

```

이러한 권한은 기존 AWS KMS 키 정책 문서 끝에 추가됩니다. 위의 정책 문은 DynamoDB 서비스 연결 역할이 kms:EncryptionContext:aws:dynamodb:tableName 컨텍스트 키에 나열된 테이블에 데이터를 복제할 수 있는 권한을 제거합니다. 위 예제에서는 transaction-global-table, inventory-global-table과 같은 이름을 가진 DynamoDB 글로벌 테이블에 대한 복제가 중지됩니다.

장애 인젝션 서비스에 서비스 연결 역할을 사용하십시오. AWS

[AWS 장애 주입 서비스는 AWS Identity and Access Management \(IAM\) 서비스 연결 역할을 사용합니다.](#) 서비스 연결 역할은 FIS에 직접 연결되는 고유한 유형의 IAM 역할입니다. AWS 서비스 연결 역할은 AWS FIS에서 사전 정의하며, 서비스에서 사용자 대신 다른 AWS 서비스를 직접적으로 호출하기 위해 필요한 모든 권한을 포함합니다.

서비스 연결 역할을 사용하면 실험용 모니터링 및 리소스 선택을 관리하는 데 필요한 권한을 수동으로 추가할 필요가 없으므로 AWS FIS를 더 쉽게 설정할 수 있습니다. AWS FIS는 서비스 연결 역할의 권한을 정의하며, 달리 정의되지 않는 한 FIS만 AWS 역할을 맡을 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

서비스 연결 역할 외에도 실험 템플릿에서 대상으로 지정한 리소스를 수정할 수 있는 권한을 부여하는 IAM 역할도 지정해야 합니다. 자세한 정보는 [AWS FIS 실험을 위한 IAM 역할](#)을 참조하세요.

먼저 관련 리소스를 삭제한 후에만 서비스 링크 역할을 삭제할 수 있습니다. 이렇게 하면 리소스 액세스 권한을 실수로 제거할 수 없으므로 AWS FIS 리소스가 보호됩니다.

FIS의 서비스 연결 역할 권한 AWS

AWS FIS는 이름이 지정된 서비스 연결 역할을 사용하여 실험을 위한 모니터링 및 리소스 선택을 관리할 수 AWSServiceRoleForFIS있도록 합니다.

AWSServiceRoleForFIS서비스 연결 역할은 다음 서비스가 역할을 맡을 것으로 신뢰합니다.

- fis.amazonaws.com

AWSServiceRoleForFIS서비스 연결 역할은 AmazonFIS의 관리형 정책을 사용합니다.

ServiceRolePolicy 이 정책을 통해 AWS FIS는 실험을 위한 모니터링 및 리소스 선택을 관리할 수 있습니다. 자세한 내용은 관리형 정책 ServiceRolePolicy 참조의 [AWS AmazonFIS](#)를 참조하십시오.

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 링크 역할을 생성하고 편집하거나 삭제할 수 있도록 권한을 구성할 수 있습니다. AWSServiceRoleForFIS서비스 연결 역할을 성공적으로 생성하려면 AWS FIS를 사용하는 IAM ID에 필요한 권한이 있어야 합니다. 필수 권한을 부여하려면 다음 정책을 IAM ID에 연결하세요.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "fis.amazonaws.com"
        }
      }
    }
  ]
}
```

자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 권한](#)을 참조하십시오.

FIS의 서비스 연결 역할 생성 AWS

서비스 링크 역할은 수동으로 생성할 필요가 없습니다. AWS Management Console AWS CLI, 또는 AWS API에서 AWS FIS 실험을 시작하면 AWS FIS가 서비스 연결 역할을 자동으로 생성합니다.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다. AWS FIS 실험을 시작하면 FIS가 서비스 연결 AWS 역할을 다시 생성합니다.

FIS의 서비스 연결 역할 편집 AWS

AWS FIS에서는 서비스 연결 역할을 편집할 수 없습니다. AWSServiceRoleForFIS 서비스 링크 역할을 생성한 후에는 다양한 개체가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 편집](#)을 참조하십시오.

FIS의 서비스 연결 역할 삭제 AWS

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 따라서 적극적으로 모니터링하거나 유지하지 않는 미사용 엔터티가 없어야 합니다. 단, 서비스 링크 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

Note

리소스를 정리하려고 할 때 AWS FIS 서비스가 역할을 사용하는 경우 정리가 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

에서 AWS 사용하는 FIS 리소스를 정리하려면 `AWSServiceRoleForFIS`

현재 실행 중인 실험이 없는지 확인합니다. 필요한 경우 실험을 중단하세요. 자세한 정보는 [실험 중지](#)를 참조하세요.

IAM을 사용하여 수동으로 서비스 링크 역할을 삭제하려면

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 `AWSServiceRoleForFIS` 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서에서 [서비스 연결 역할 삭제](#)를 참조하세요.

FIS 서비스 연결 역할이 지원되는 AWS 지역

AWS FIS는 서비스를 사용할 수 있는 모든 지역에서 서비스 연결 역할을 사용할 수 있도록 지원합니다. 자세한 내용은 [AWS Fault Injection Service 엔드포인트 및 할당량](#)을 참조하세요.

AWS 장애 주입 서비스에 대한 AWS 관리형 정책

AWS 관리형 정책은 에서 생성하고 관리하는 독립형 정책입니다. AWS AWS 관리형 정책은 많은 일반 사용 사례에 대한 권한을 제공하도록 설계되었으므로 사용자, 그룹 및 역할에 권한을 할당하기 시작할 수 있습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있으므로 특정 사용 사례에 대해 최소 권한 권한을 부여하지 않을 수도 있다는 점에 유의하세요. 사용 사례에 고유한 [고객 관리형 정책](#)을 정의하여 권한을 줄이는 것이 좋습니다.

관리형 정책에 정의된 권한은 변경할 수 없습니다. AWS AWS 관리형 정책에 정의된 권한을 업데이트 하는 경우 AWS 해당 업데이트는 정책이 연결된 모든 주체 ID (사용자, 그룹, 역할) 에 영향을 미칩니다.

AWS 새 API 작업이 시작되거나 기존 서비스에 새 AWS 서비스 API 작업을 사용할 수 있게 되면 AWS 관리형 정책을 업데이트할 가능성이 가장 높습니다.

자세한 내용은 IAM 사용 설명서의 [AWS 관리형 정책](#)을 참조하십시오.

AWS 관리형 정책: AmazonFIS ServiceRolePolicy

이 정책은 AWS FIS가 실험용 모니터링 및 리소스 선택을 관리할 수 AWSServiceRoleForFIS있도록 이름이 지정된 서비스 연결 역할에 첨부됩니다. 자세한 정보는 [장애 인젝션 서비스에 서비스 연결 역할을 사용하십시오. AWS](#)을 참조하세요.

AWS 관리형 정책: AWSFaultInjectionSimulatorEC2Access

실험 역할에서 이 정책을 사용하여 Amazon [EC2용 AWS FIS 작업을 사용하는 실험을 실행할 권한을 AWS FIS에](#) 부여하십시오. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.

이 정책에 대한 권한을 보려면 AWS 관리형 정책 참조를 참조하십시오 [AWSFaultInjectionSimulatorEC2Access](#).

AWS 관리형 정책: AWSFaultInjectionSimulatorECSAccess

실험 역할에서 이 정책을 사용하여 [Amazon AWS ECS용 FIS 작업을 사용하는 실험을 실행할 권한을 AWS FIS에](#) 부여하십시오. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.

이 정책에 대한 권한을 보려면 AWS 관리형 정책 참조를 참조하십시오 [AWSFaultInjectionSimulatorECSAccess](#).

AWS 관리형 정책: AWSFaultInjectionSimulatorEKSAccess

실험 역할에서 이 정책을 사용하여 [Amazon AWS EKS용 FIS 작업을 사용하는 실험을 실행할 권한을 AWS FIS에](#) 부여하십시오. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.

이 정책에 대한 권한을 보려면 AWS 관리형 정책 참조를 참조하십시오 [AWSFaultInjectionSimulatorEKSAccess](#).

AWS 관리형 정책: AWSFaultInjectionSimulatorNetworkAccess

실험 역할에서 이 정책을 사용하여 AWS FIS [네트워킹 작업을 사용하는 실험을 실행할 권한을 AWS FIS에](#) 부여하십시오. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.

이 정책에 대한 권한을 보려면 AWS 관리형 정책 참조를 참조하십시오 [AWSFaultInjectionSimulatorNetworkAccess](#).

AWS 관리형 정책: AWSFaultInjectionSimulatorRDSAccess

실험 역할에서 이 정책을 사용하여 [Amazon AWS RDS용 FIS 작업을 사용하는 실험을 실행할 권한을 AWS FIS에](#) 부여하십시오. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.

이 정책에 대한 권한을 보려면 AWS 관리형 정책 참조를 참조하십시오 [AWSFaultInjectionSimulatorRDSAccess](#).

AWS 관리형 정책: AWSFaultInjectionSimulatorSSMAccess

실험 역할에서 이 정책을 사용하면 [Systems Manager용 AWS FIS 작업을 사용하는 실험을 실행할 수 있는 권한을 AWS FIS에](#) 부여할 수 있습니다. 자세한 정보는 [the section called “실험 역할”](#)을 참조하세요.

이 정책에 대한 권한을 보려면 AWS 관리형 정책 참조를 참조하십시오 [AWSFaultInjectionSimulatorSSMAccess](#).

AWSAWS 관리형 정책에 대한 FIS 업데이트

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후 AWS FIS의 AWS 관리형 정책 업데이트에 대한 세부 정보를 확인하십시오.

변경 사항	설명	날짜
AWSFaultInjectionSimulatorECSAccess - 기존 정책에 대한 업데이트	AWS FIS가 ECS 대상을 해결할 수 있는 권한이 추가되었습니다.	2024년 1월 25일
AWSFaultInjectionSimulatorNetworkAccess -기존 정책 업데이트	AWS FIS가 및 작업을 사용하여 실험을 실행할 수 있는 권한이 추가되었습니다. aws:network:route-table-disrupt-cross-region-connectivity aws:network:transit-gateway-disrupt-cross-region-connectivity	2024년 1월 25일
AWSFaultInjectionSimulatorEC2Access -기존 정책 업데이트	AWS FIS가 EC2 인스턴스를 해결할 수 있는 권한을 추가했습니다.	2023년 11월 13일
AWSFaultInjectionSimulatorEKSAccess -기존 정책 업데이트	AWS FIS가 EKS 대상을 해결할 수 있는 권한이 추가되었습니다.	2023년 11월 13일

변경 사항	설명	날짜
AWSFaultInjectionSimulatorRDSAccess -기존 정책 업데이트	AWS FIS가 RDS 대상을 해결할 수 있는 권한이 추가되었습니다.	2023년 11월 13일
AWSFaultInjectionSimulatorEC2Access -기존 정책 업데이트	AWS FIS가 EC2 인스턴스에서 SSM 문서를 실행하고 EC2 인스턴스를 종료할 수 있는 권한을 추가했습니다.	2023년 6월 2일
AWSFaultInjectionSimulatorSSMAccess -기존 정책 업데이트	AWS FIS가 EC2 인스턴스에서 SSM 문서를 실행할 수 있는 권한을 추가했습니다.	2023년 6월 2일
AWSFaultInjectionSimulatorECSAccess -기존 정책 업데이트	AWS FIS가 새 작업을 사용하여 실험을 실행할 수 있는 권한을 추가했습니다. <code>aws:ecs:task</code>	2023년 6월 1일
AWSFaultInjectionSimulatorEKSAccess -기존 정책 업데이트	AWS FIS가 새 <code>aws:eks:pod</code> 작업을 사용하여 실험을 실행할 수 있는 권한이 추가되었습니다.	2023년 6월 1일
AWSFaultInjectionSimulatorEC2Access - 새 정책	FIS가 Amazon EC2용 AWS FIS 작업을 사용하는 AWS 실험을 실행할 수 있도록 허용하는 정책이 추가되었습니다.	2022년 10월 26일
AWSFaultInjectionSimulatorECSAccess - 새 정책	FIS가 Amazon AWS ECS용 FIS 작업을 사용하는 AWS 실험을 실행할 수 있도록 허용하는 정책이 추가되었습니다.	2022년 10월 26일
AWSFaultInjectionSimulatorEKSAccess - 새 정책	FIS가 Amazon EKS에 대한 AWS FIS 작업을 사용하는 AWS 실험을 실행할 수 있도록 허용하는 정책이 추가되었습니다.	2022년 10월 26일
AWSFaultInjectionSimulatorNetworkAccess - 새 정책	AWS FIS가 FIS 네트워킹 작업을 사용하는 AWS 실험을 실행할 수 있도록 허용하는 정책이 추가되었습니다.	2022년 10월 26일

변경 사항	설명	날짜
AWSFaultInjectionSimulatorRDSAccess - 새 정책	FIS가 Amazon RDS에서 AWS FIS 작업을 사용하는 AWS 실험을 실행할 수 있도록 허용하는 정책이 추가되었습니다.	2022년 10월 26일
AWSFaultInjectionSimulatorSMSAccess - 새 정책	FIS가 Systems Manager용 FIS 작업을 사용하는 AWS 실험을 실행할 수 있도록 허용하는 정책이 추가되었습니다.	2022년 10월 26일
AmazonFIS ServiceRolePolicy — 기존 정책에 대한 업데이트	AWS FIS가 서브넷을 설명할 수 있도록 권한을 추가했습니다.	2022년 10월 26일
AmazonFIS ServiceRolePolicy — 기존 정책에 대한 업데이트	AWS FIS가 EKS 클러스터를 설명할 수 있도록 권한을 추가했습니다.	2022년 7월 7일
AmazonFIS ServiceRolePolicy — 기존 정책에 대한 업데이트	AWS FIS가 클러스터의 작업을 나열하고 설명할 수 있는 권한이 추가되었습니다.	2022년 2월 7일
AmazonFIS ServiceRolePolicy — 기존 정책에 대한 업데이트	events:DescribeRule 작업에서 events:ManagedBy 조건을 제거했습니다.	2022년 1월 6일
AmazonFIS — 기존 정책에 대한 업데이트 ServiceRolePolicy	AWS FIS가 중지 상태에서 사용된 CloudWatch 경보에 대한 기록을 검색할 수 있는 권한이 추가되었습니다.	2021년 6월 30일
AWS FIS는 변경 사항을 추적하기 시작했습니다.	AWS FIS는 관리형 정책의 변경 사항을 추적하기 시작했습니다. AWS	2021년 3월 1일

AWS 장애 인젝션 서비스의 인프라 보안

AWS 장애 주입 서비스는 관리형 서비스로서 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스 및 인프라 AWS 보호 방법에 대한 자세한 내용은 [AWS 클라우드 보안을 참조하십시오](#). 인프라

보안 모범 사례를 사용하여 AWS 환경을 설계하려면 Security Pillar AWS Well-Architected Framework의 [인프라 보호](#)를 참조하십시오.

AWS 게시된 API 호출을 사용하여 네트워크를 통해 AWS FIS에 액세스할 수 있습니다. 고객은 다음을 지원해야 합니다.

- 전송 계층 보안(TLS). TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 주체와 관련된 비밀 액세스 키를 사용하여 서명해야 합니다. 또는 [AWS Security Token Service](#)(AWS STS)를 사용하여 임시 보안 인증 정보를 생성하여 요청에 서명할 수 있습니다.

인터페이스 VPC AWS 엔드포인트 () 를 사용하여 FIS에 액세스 AWS PrivateLink

인터페이스 VPC 엔드포인트를 생성하여 VPC와 AWS 장애 인젝션 서비스 간에 프라이빗 연결을 설정할 수 있습니다. VPC 엔드포인트는 인터넷 게이트웨이, NAT 디바이스 [AWS PrivateLink](#), VPN 연결 또는 Direct Connect 연결 없이 AWS FIS API에 비공개로 액세스할 수 있는 기술인 에 의해 구동됩니다. AWS VPC의 인스턴스는 AWS FIS API와 통신하는 데 퍼블릭 IP 주소가 필요하지 않습니다.

각 인터페이스 엔드포인트는 서브넷에서 하나 이상의 [탄력적 네트워크 인터페이스](#)로 표현됩니다.

자세한 내용은 AWS PrivateLink가이드의 [액세스를 AWS 서비스 참조하십시오](#).AWS PrivateLink

AWS FIS VPC 엔드포인트에 대한 고려 사항

AWS FIS용 인터페이스 VPC 엔드포인트를 설정하기 전에 가이드의 [인터페이스 VPC 엔드포인트 AWS 서비스 사용 액세스](#)를 검토하십시오.AWS PrivateLink

AWS FIS는 VPC에서 모든 API 작업에 대한 호출을 지원합니다.

FIS용 인터페이스 VPC 엔드포인트 생성 AWS

Amazon VPC 콘솔 또는 () 를 사용하여 AWS FIS 서비스에 대한 VPC 엔드포인트를 생성할 수 있습니다. AWS Command Line Interface AWS CLI자세한 정보는 AWS PrivateLink 가이드의 [VPC 엔드포인트 생성](#)을 참조하세요.

다음 서비스 이름을 사용하여 AWS FIS용 VPC 엔드포인트를 생성합니다.

```
com.amazonaws.region.fis
```

엔드포인트에 프라이빗 DNS를 활성화하면 해당 지역의 기본 DNS 이름 (예:) 을 사용하여 AWS FIS에 API 요청을 보낼 수 있습니다. `fis.us-east-1.amazonaws.com`

FIS를 위한 VPC 엔드포인트 정책 생성 AWS

FIS 액세스를 제어하는 엔드포인트 정책을 VPC 엔드포인트에 연결할 수 있습니다. AWS 이 정책은 다음 정보를 지정합니다.

- 작업을 수행할 수 있는 보안 주체.
- 수행할 수 있는 작업.
- 작업을 수행할 수 있는 리소스.

자세한 정보는 AWS PrivateLink 가이드의 [엔드포인트 정책을 사용하여 VPC 엔드포인트에 대한 액세스 제어](#)를 참조하세요.

예: 특정 AWS FIS 작업에 대한 VPC 엔드포인트 정책

다음 VPC 엔드포인트 정책은 모든 리소스에서 나열된 AWS FIS 작업에 대한 액세스 권한을 모든 보안 주체에게 부여합니다.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "fis:ListExperimentTemplates",
        "fis:StartExperiment",
        "fis:StopExperiment",
        "fis:GetExperiment"
      ],
      "Resource": "*",
      "Principal": "*"
    }
  ]
}
```

예: 특정 사용자로부터의 액세스를 거부하는 VPC 엔드포인트 정책 AWS 계정

다음 VPC 엔드포인트 정책은 모든 작업과 리소스에 AWS 계정 대한 지정된 액세스를 거부하지만 모든 작업과 리소스에 AWS 계정 대한 다른 모든 액세스는 부여합니다.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*",
      "Principal": "*"
    },
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Principal": {
        "AWS": [ "123456789012" ]
      }
    }
  ]
}
```

AWS FIS 리소스에 태그 지정

태그는 사용자 또는 AWS가 AWS 리소스에 할당하는 메타데이터 레이블입니다. 각 태그는 키와 값으로 구성됩니다. 사용자가 할당하는 태그에 대해 키와 값을 정의합니다. 예를 들어 키를 `purpose`로 정의하고 리소스 값을 `test`로 정의할 수 있습니다.

태그는 다음을 지원합니다.

- AWS 리소스를 식별하고 정리합니다. 많은 AWS 서비스가 태그 지정을 지원하므로 다른 서비스의 리소스에 동일한 태그를 할당하여 해당 리소스의 관련 여부를 나타낼 수 있습니다.
- AWS 리소스에 대한 액세스를 제어합니다. 자세한 내용은 IAM 사용 설명서의 [태그를 사용한 액세스 제어](#)를 참조하세요.

태그 지정 제한

AWS FIS 리소스의 태그에 다음과 같은 기본 제한이 적용됩니다.

- 리소스에 할당할 수 있는 최대 태그 수: 50
- 최대 키 길이: 유니코드 128자
- 최대 값 길이: 유니코드 256자
- 키 및 값에 사용할 수 있는 문자 - a-z, A-Z, 0-9, 공백, `_`, `.`, `/`, `=`, `+`, `-` 및 `@` 문자
- 키와 값은 대/소문자를 구분합니다
- 키 접두사로 `aws:`를 사용하지 마세요. AWS 사용을 위해 예약되어 있습니다.

태그 작업

다음 AWS Fault Injection Service(AWS FIS) 리소스는 태그 지정을 지원합니다.

- 작업
- 실험
- 실험 템플릿

콘솔을 사용하여 실험용 태그 및 실험 템플릿으로 작업할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- [실험에 태그 지정](#)
- [실험 템플릿에 태그 지정](#)

다음 AWS CLI 명령을 사용하여 작업, 실험 및 실험 템플릿에 대한 태그 작업을 수행할 수 있습니다.

- [tag-resource](#) – 리소스에 태그를 추가합니다.
- [untag-resource](#) – 리소스에서 태그를 제거합니다.
- [list-tags-for-resource](#)— 특정 리소스의 태그를 나열합니다.

AWS 장애 주입 서비스의 할당량 및 한도

Your AWS 계정 서비스에는 각 서비스에 대한 기본 할당량 (이전에는 한도라고 함) 이 있습니다. AWS 다르게 표시되지 않는 한 리전별로 각 할당량이 적용됩니다. 일부 할당량에 대한 증가를 요청할 수 있으며 모든 할당량에 대한 증가를 요청할 수 없습니다.

AWS [FIS 할당량을 보려면 Service Quotas 콘솔을 엽니다](#). 탐색 창에서 AWS 서비스를 선택하고 AWS Fault Injection Service를 선택합니다.

할당량 증가를 요청하려면 [Service Quotas 사용 설명서](#)의 할당량 증가 요청을 참조하세요.

FIS와 AWS 계정 관련된 할당량은 다음과 같습니다. AWS

명칭	기본값	조정 가능	설명
작업 기간 (시간)	지원되는 각 리전: 12	아니요	현재 리전에서 이 계정으로 하나의 작업을 실행할 수 있는 최대 시간.
실험 템플릿별 작업	지원되는 각 지역: 20	아니요	현재 리전에서 이 계정의 실험 템플릿에 만들 수 있는 작업 최대 수.
적극적인 실험	지원되는 각 리전: 5개	아니요	현재 리전에서 이 계정으로 동시에 실행할 수 있는 활성 실험 최대 수.
완료된 실험 데이터 보존 기간(일)	지원되는 각 리전: 120	아니요	AWS FIS가 현재 지역의 이 계정에 완료된 실험에 대한 데이터를 보관할 수 있는 최대 일수입니다.
실험 기간(시간)	지원되는 각 리전: 12	아니요	현재 리전에서 이 계정으로 하나의 실험을 실행할 수 있는 최대 시간.

명칭	기본값	조정 가능	설명
실험 템플릿	지원되는 각 리전: 500개	아니요	현재 리전의 이 계정에서 생성할 수 있는 실험 템플릿 최대 수.
aws:network: -지역 연결 내 관리형 프리픽스 목록의 최대 개수 route-table-disrupt-cross	지원되는 각 리전: 15	아니요	작업당 aws:network: -region 커넥티비티에서 허용하는 관리형 프리픽스 목록의 최대 개수입니다. route-table-disrupt-cross
AWS의 최대 라우팅 테이블 수:네트워크: -지역 연결 route-table-disrupt-cross	지원되는 각 리전: 10	아니요	작업당 aws:network: route-table-disrupt-cross -region 커넥티비티가 허용하는 최대 라우팅 테이블 수입니다.
AWS의 최대 경로 수:네트워크: -지역 연결 route-table-disrupt-cross	지원되는 각 리전: 200개	아니요	aws:network: route-table-disrupt-cross -region 커넥티비티가 액션당 허용하는 최대 경로 수입니다.
실험당 병렬 작업	지원되는 각 리전: 10	아니요	현재 리전에서 이 계정으로 실험에 동시에 실행할 수 있는 작업의 최대 수.
실험 템플릿별 중지 조건	지원되는 각 리전: 5개	아니요	현재 리전에서 이 계정의 실험 템플릿에 추가할 수 있는 중지 조건의 최대 수.

명칭	기본값	조정 가능	설명
aws:ec2의 타겟 Auto Scaling 그룹: -오류 asg-insufficient-instance-capacity	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 aws:ec2: asg-insufficient-instance-capacity -error가 실험당 대상으로 지정할 수 있는 Auto Scaling 그룹의 최대 수입니다.
aws:s3의 타겟 버킷: bucket-pause-replication	지원되는 각 리전: 20개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:s3:가 bucket-pause-replication 타겟팅할 수 있는 최대 S3 버킷 수입니다.
aws:ecs의 타겟 클러스터: drain-container-instances	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:ecs:가 drain-container-instances 타겟팅할 수 있는 최대 클러스터 수입니다.
aws:rds의 대상 클러스터: failover-db-cluster	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:rds:가 failover-db-cluster 타겟팅할 수 있는 최대 클러스터 수입니다.

명칭	기본값	조정 가능	설명
aws:rds의 대상 DB 인스턴스: reboot-db-instances	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 aws:rds가 대상으로 지정할 reboot-db-instances 수 있는 실험당 최대 DB 인스턴스 수입니다.
aws:ec2:reboot-instances의 대상 인스턴스	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:ec2:reboot-instances가 대상으로 지정할 수 있는 인스턴스의 최대 수.
aws:ec2:stop-instances의 대상 인스턴스	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:ec2:stop-instances가 대상으로 지정할 수 있는 인스턴스의 최대 수.
aws:ec2:terminate-instances의 대상 인스턴스	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:ec2:terminate-instances가 대상으로 지정할 수 있는 인스턴스의 최대 수.
aws:ssm:send-command 대상 인스턴스	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:ssm:send-command가 타깃팅할 수 있는 최대 인스턴스 수.

명칭	기본값	조정 가능	설명
aws:eks의 대상 노드 그룹: terminate-nodegroup-instances	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 aws:eks가 terminate-nodegroup-instances 타겟팅할 수 있는 실험당 최대 노드 그룹 수입니다.
aws:eks의 타겟 포드: pod-cpu-stress	지원되는 각 지역: 50	예	파라미터를 사용하여 대상을 식별할 때 실험당 aws:eks가 pod-cpu-stress 타겟팅할 수 있는 파드의 최대 개수입니다.
aws:eks:pod-delete 대상 포드	지원되는 각 리전: 50개	예	파라미터를 사용하여 대상을 식별할 때 실험당 aws:eks:eks:pod-delete가 대상으로 지정할 수 있는 포드 최대 수.
aws:eks의 타겟 포드: pod-io-stress	지원되는 각 지역: 50	예	파라미터를 사용하여 대상을 식별할 때 실험당 aws:eks가 pod-io-stress 타겟팅할 수 있는 파드의 최대 개수입니다.
aws:eks의 타겟 포드: pod-memory-stress	지원되는 각 지역: 50	예	파라미터를 사용하여 대상을 식별할 때 실험당 aws:eks가 pod-memory-stress 타겟팅할 수 있는 파드의 최대 개수입니다.

명칭	기본값	조정 가능	설명
aws:eks의 타겟 포트: pod-network-blackhole-port	지원되는 각 지역: 50	예	파라미터를 사용하여 대상을 식별할 때 실험당 aws:eks:가 pod-network-blackhole-port 타겟팅할 수 있는 파드의 최대 개수입니다.
aws:eks의 타겟 포트: pod-network-latency	지원되는 각 지역: 50	예	파라미터를 사용하여 대상을 식별할 때 실험당 aws:eks:가 pod-network-latency 타겟팅할 수 있는 파드의 최대 개수입니다.
aws:eks의 타겟 포트: pod-network-packet-loss	지원되는 각 지역: 50	예	파라미터를 사용하여 대상을 식별할 때 실험당 aws:eks:가 pod-network-packet-loss 타겟팅할 수 있는 파드의 최대 개수입니다.
AWS:엘라스틱캐시의 대상: ReplicationGroups interrupt-cluster-az-power	지원되는 각 리전: 5개	예	실험당 태그/파라미터를 사용하여 대상을 식별할 때 interrupt-cluster-az-power 타겟팅할 수 있는 ReplicationGroups 있는 aws:elasticache의 최대 개수입니다.

명칭	기본값	조정 가능	설명
aws:ec2의 타겟은 SpotInstances 다음과 같습니다. send-spot-instance-interruptions	지원되는 각 리전: 5개	예	실험당 태그를 사용하여 대상을 식별할 때 aws:ec2가 send-spot-instance-interruptions 타겟팅할 수 있는 최대 개수입니다. SpotInstances
aws:network:disrupt-connectivity 대상 서브넷	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:network:disrupt-connectivity가 대상으로 지정할 수 있는 서브넷의 최대 수.
aws:네트워크: -지역 연결의 대상 서브넷 route-table-disrupt-cross	지원되는 각 리전: 6개	예	실험당 태그를 사용하여 대상을 식별할 때 aws:network: route-table-disrupt-cross -region connectivity 대상이 될 수 있는 최대 서브넷 수입니다.
aws:ecs:stop-task 대상 태스크	지원되는 각 리전: 5개	예	태그를 사용하여 대상을 식별할 때 실험당 aws:ecs:ecs:stop-task가 대상으로 지정할 수 있는 작업의 최대 수.

명칭	기본값	조정 가능	설명
aws:ecs의 대상 태스크: task-cpu-stress	지원되는 각 리전: 5개	예	태그/파라미터를 사용하여 대상을 식별할 때 실험당 aws:ecs:가 task-cpu-stress 타겟팅할 수 있는 최대 작업 수입니다.
aws:ecs의 타겟 태스크: task-io-stress	지원되는 각 리전: 5개	예	태그/파라미터를 사용하여 대상을 식별할 때 실험당 aws:ecs:가 task-io-stress 타겟팅할 수 있는 최대 작업 수입니다.
aws:ecs의 타겟 태스크: task-kill-process	지원되는 각 리전: 5개	예	태그/파라미터를 사용하여 대상을 식별할 때 실험당 aws:ecs:가 task-kill-process 타겟팅할 수 있는 최대 작업 수입니다.
aws:ecs의 타겟 태스크: task-network-blackhole-port	지원되는 각 리전: 5개	예	태그/파라미터를 사용하여 대상을 식별할 때 실험당 aws:ecs:가 task-network-blackhole-port 타겟팅할 수 있는 최대 작업 수입니다.
aws:ecs의 타겟 태스크: task-network-latency	지원되는 각 리전: 5개	예	태그/파라미터를 사용하여 대상을 식별할 때 실험당 aws:ecs:가 task-network-latency 타겟팅할 수 있는 최대 작업 수입니다.

명칭	기본값	조정 가능	설명
aws:ecs의 타겟 태스크: task-network-packet-loss	지원되는 각 리전: 5개	예	태그/파라미터를 사용하여 대상을 식별할 때 실험당 aws:ecs:가 task-network-packet-loss 타겟팅할 수 있는 최대 작업 수입니다.
AWS:네트워크 대상: -지역 연결 TransitGateways transit-gateway-disrupt-cross	지원되는 각 리전: 5개	예	실험당 태그를 사용하여 대상을 식별할 때 aws:network: transit-gateway-disrupt-cross - region connectivity 기능이 타겟팅할 수 있는 최대 트랜짓 게이트웨이 수입니다.
실험 템플릿별 대상 계정 구성	지원되는 각 리전: 10개	예	현재 리전에서 이 계정의 실험 템플릿에 대해 만들 수 있는 최대 대상 계정 구성 수.

AWS FIS 사용에는 다음과 같은 추가 제한 사항이 적용됩니다.

명칭	제한 사항
조치 대상 aws:elasticache:interrupt-cluster-az-power	계정당 하루에 지역당 10개의 aws:elasticache:redis-replicationgroup 클러스터가 손상될 수 있도록 제한됩니다. AWS Support 센터 콘솔에서 지원 사례를 생성하여 증가를 요청할 수 있습니다.

문서 기록

다음 표에서는 AWS Fault Injection Service 사용 설명서의 중요한 설명서 업데이트에 대해 설명합니다.

변경 사항	설명	날짜
새로운 액션 모드 실험 옵션	액션 모드를 로 skip-all 설정하여 실험을 실행하기 전에 타겟 미리보기를 생성할 수 있습니다.	2024년 3월 13일
AWS 관리형 정책 업데이트	AWS FIS는 기존 관리형 정책을 업데이트했습니다.	2024년 1월 25일
새 작업 및 시나리오	이제 AWS FIS 시나리오 크로스 리전: 연결 및 AZ 가용성: 전원 중단을 사용할 수 있습니다.	2023년 11월 30일
새로운 작업	이제 aws:ec2:asg-insufficient-instance-capacity-error 작업을 사용할 수 있습니다.	2023년 11월 30일
새로운 작업	이제 aws:ec2:api-insufficient-instance-capacity-error 작업을 사용할 수 있습니다.	2023년 11월 30일
새로운 작업	이제 aws:network:route-table-disrupt-cross-region-connectivity 작업을 사용할 수 있습니다.	2023년 11월 30일
새로운 작업	이제 aws:network:transit-gateway-disrupt-cross-region-connectivity 작업을 사용할 수 있습니다.	2023년 11월 30일
새로운 작업	이제 aws:dynamodb:encrypted-global-table-pause-rep	2023년 11월 30일

	lication 작업을 사용할 수 있습니다.	
새로운 작업	이제 aws:s3:bucket-pause-replication 작업을 사용할 수 있습니다.	2023년 11월 30일
새로운 작업	이제 aws:elasticache:interrupt-cluster-az-power 작업을 사용할 수 있습니다.	2023년 11월 30일
새 실험 옵션	이제 계정 타겟팅 및 빈 타겟 해결에 AWS FIS 실험 옵션을 사용할 수 있습니다.	2023년 11월 27일
FIS 이름 변경 AWS	서비스 이름을 AWS 장애 주입 서비스로 업데이트했습니다.	2023년 11월 15일
AWS 관리형 정책 업데이트	AWS FIS는 기존 관리형 정책을 업데이트했습니다.	2023년 11월 13일
새 시나리오 라이브러리	이제 AWS FIS 시나리오 라이브러리 기능을 사용할 수 있습니다.	2023년 11월 7일
새 실험 스케줄러	이제 AWS FIS 실험 스케줄러 기능을 사용할 수 있습니다.	2023년 11월 7일
AWS 관리형 정책 업데이트	AWS FIS는 기존 관리형 정책을 업데이트했습니다.	2023년 6월 2일
새로운 작업	새 aws:ecs:task 및 aws:eks:pod 작업을 사용할 수 있습니다.	2023년 6월 1일
AWS 관리형 정책 업데이트	AWS FIS는 기존 관리형 정책을 업데이트했습니다.	2023년 6월 1일
사전 구성된 새 SSM 문서	다음과 같은 사전 구성된 SSM 문서인 -Run-Disk-Fill을 사용할 수 있습니다. AWSFIS	2023년 4월 28일

새로운 작업	aws:ebs:pause-volume-io 작업을 사용하여 대상 볼륨과 대상 볼륨이 연결된 인스턴스 간의 I/O를 일시 중지할 수 있습니다.	2023년 1월 27일
새로운 작업	aws:network:disrupt-connectivity 작업을 사용하여 대상 서브넷으로 향하는 특정 유형의 트래픽을 거부할 수 있습니다.	2022년 10월 26일
새로운 작업	aws:eks:inject-kubernetes-custom-resource 작업을 사용하여 단일 대상 클러스터에서 ChaosMesh 또는 Litmus 실험을 실행할 수 있습니다.	2022년 7월 7일
실험 로깅	실험 활동 로그를 Logs 또는 S3 버킷으로 전송하도록 CloudWatch 실험 템플릿을 구성할 수 있습니다.	2022년 2월 28일
새 알림	실험 상태가 변경되면 AWS FIS는 알림을 보냅니다. 이러한 알림은 Amazon을 통해 이벤트로 제공됩니다 EventBridge.	2022년 2월 24일
새로운 작업	aws:ecs:stop-task 작업을 사용하여 지정된 작업을 중지할 수 있습니다.	2022년 2월 9일
새로운 작업	aws:cloudwatch:assert-alarm-state 작업을 사용하여 지정된 경보가 지정된 경보 상태 중 하나에 해당하는지 확인할 수 있습니다.	2021년 11월 5일

사전 구성된 새 SSM 문서

사전 구성된 SSM 문서로는 - Run-IO-Stress, -Run-Network-Blackhold-Port, -Run-Network-Latency-Sources, AWSFIS -Run-Network-Packet-Loss 및 AWSFIS -Run-Network-Packet-Loss Sources 등의 SSM 문서를 사용할 수 있습니다. AWSFIS AWSFIS AWSFIS

2021년 11월 4일

새로운 작업

aws:ec2:send-spot-instance-interruptions 작업을 사용하여 대상 스팟 인스턴스에 스팟 인스턴스 중단 알림을 보낸 다음 대상 스팟 인스턴스를 중단할 수 있습니다.

2021년 10월 20일

새로운 작업

aws:ssm:start-automation-execution 작업을 사용하여 자동화 런북의 실행을 시작할 수 있습니다.

2021년 9월 17일

최초 릴리스

장애 AWS 주입 서비스 사용 설명서의 초기 릴리스.

2021년 3월 15일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.