Service Name

AWS Fault Injection Service

Service Overview

AWS Fault Injection Service (AWS FIS) is a fully managed service that enables customers to perform fault injection experiments on AWS workloads based on chaos engineering principles. The service creates disruptive events to test application resilience by intentionally injecting faults such as stopping instances, disrupting network connectivity, stressing CPU/memory, or causing API errors. AWS FIS provides templates, controls, and guardrails needed to run experiments safely in production environments, including automatic rollback capabilities and stop conditions based on CloudWatch alarms. The service helps uncover application issues that are difficult to find through traditional testing methods.

Key Use Cases

- Use case 1: Application resilience testing by injecting faults like instance failures, network disruptions, and API errors to validate recovery procedures and failover mechanisms
- Use case 2: Chaos engineering implementation to build confidence in system reliability by regularly testing applications under real-world failure conditions in production environments
- Use case 3: Disaster recovery validation by simulating regional failures, availability zone disruptions, and service outages to verify backup systems and recovery time objectives

Features

- **Preconfigured Actions**: Ready-to-use fault injection activities for EC2, ECS, EKS, RDS, Lambda, S3, and other AWS services including instance termination, network disruption, and API errors
- **Stop Conditions**: CloudWatch alarm-based safety mechanisms that automatically halt experiments when predefined thresholds are exceeded to prevent uncontrolled damage
- **Multi-Account Experiments**: Capability to run fault injection experiments across multiple AWS accounts from a centralized orchestrator account with cross-account IAM roles
- **Systems Manager Integration**: Integration with SSM to execute custom fault injection scripts and automation documents for complex multi-step experiments
- **Experiment Templates**: Reusable blueprints that define actions, targets, stop conditions, and experiment configurations for consistent testing scenarios

Value Proposition

AWS FIS provides a fully managed chaos engineering solution that eliminates the complexity of building custom fault injection tools. Unlike traditional testing approaches, FIS offers production-safe experimentation with built-in guardrails, automatic rollback capabilities, and fine-grained targeting using AWS resource tags:

- The service integrates seamlessly with existing AWS services and provides preconfigured fault scenarios, reducing setup time and expertise requirements
- Key differentiators include comprehensive safety controls, multi-account support for enterprise scenarios, deep AWS service integration, and pay-per-use pricing that scales with experiment duration rather than requiring upfront infrastructure investment

Service Integration

AWS FIS integrates deeply with core AWS services to provide comprehensive fault injection capabilities. Primary integrations include Amazon CloudWatch for monitoring and stop conditions, AWS IAM for access control and experiment permissions, and AWS Systems Manager for custom script execution:

- The service works with compute services (EC2, ECS, EKS, Lambda), storage services (EBS, S3), databases (RDS, DynamoDB), and networking components (VPC, Route Tables, Transit Gateway)
- Additional integrations include AWS Application Recovery Controller for zonal autoshift testing, AWS Direct Connect for BGP session disruption, and multi-account management through AWS Organizations for enterprise-scale chaos engineering programs

Onboarding and Offboarding

Customers begin by creating IAM roles with necessary permissions for FIS to perform actions on target resources. The getting started process involves identifying target workloads, defining steady-state behavior, and creating experiment templates through the AWS console, CLI, or APIs:

- AWS provides pre-built experiment templates and tutorials for common scenarios like EC2 instance termination and network connectivity disruption
- Best practices include starting with non-production environments, implementing proper monitoring with CloudWatch alarms as stop conditions, and gradually expanding experiment scope
- The service requires no infrastructure provisioning as it's fully managed, allowing immediate experimentation once proper permissions are configured

Getting Started Guide: https://docs.aws.amazon.com/fis/latest/userguide/getting-started-planning.html

Data Removal

AWS FIS automatically retains completed experiment data for 120 days in each region, after which it's automatically deleted. During offboarding, customers should delete experiment templates and ensure no active experiments are running. The service doesn't store persistent customer data beyond experiment logs and templates, so offboarding primarily involves removing IAM roles and cleaning up experiment artifacts. Customers can manually delete experiment templates and associated CloudWatch alarms if no longer needed.

Backup/Restore and Disaster Recovery

AWS FIS itself doesn't provide backup and disaster recovery capabilities as it's a testing service that operates on existing resources. However, the service is instrumental in validating backup and disaster recovery procedures by simulating failures and testing recovery mechanisms. FIS experiments help verify that backup systems, failover processes, and disaster recovery plans function correctly under stress conditions, ensuring RPO and RTO objectives are met.

Backup Capabilities

AWS FIS doesn't have traditional backup capabilities as it's a fault injection testing service rather than a data storage service. The service doesn't integrate directly with AWS Backup since it operates on existing resources to test their resilience rather than storing data that requires backup protection.

Disaster Recovery

AWS FIS doesn't support AWS Elastic Disaster Recovery integration for its own operations, but serves as a critical tool for testing disaster recovery scenarios. The service helps validate disaster recovery procedures by simulating regional failures, AZ disruptions, and service outages to ensure recovery mechanisms function properly.

Recovery Objectives

AWS FIS helps customers validate their RPO and RTO objectives by simulating real failure scenarios and measuring actual recovery times. Through controlled chaos experiments, customers can verify that backup restoration meets RPO requirements and failover procedures achieve target RTO windows. The service enables testing of automated recovery mechanisms and identification of gaps between planned and actual recovery performance.

Service Constraints

Service Quotas: https://docs.aws.amazon.com/fis/latest/userguide/fis-quotas.html

FAQ: https://aws.amazon.com/fis/faqs/

Technical Requirements

Service Documentation: https://docs.aws.amazon.com/fis/latest/userguide/what-is.html

User Guide: https://docs.aws.amazon.com/fis/latest/userguide/

API Reference: https://docs.aws.amazon.com/fis/latest/APIReference/

Pricing Overview

Please see the AWS UK G Cloud 15 Pricing Document accompanying this service in the Digital Marketplace.

Public Pricing: https://aws.amazon.com/fis/pricing/

Cost Optimization

AWS FIS offers unique cost optimization through its pay-per-use pricing model charged per action-minute rather than provisioned infrastructure. Customers pay $0.10 per action-minute in most regions, with additional charges only for multi-account experiments:

- Cost optimization strategies include running shorter experiments, targeting specific resources rather than broad sweeps, and using experiment templates to avoid repetitive setup costs
- The service eliminates infrastructure costs for chaos engineering tools while providing comprehensive testing capabilities
- Experiment reports cost $5 each, allowing customers to balance detailed reporting needs with cost considerations
- Regional pricing variations exist, with GovCloud regions priced at $0.12 per action-minute

Savings Considerations

Primary cost savings come from preventing costly production outages through proactive resilience testing, reducing mean time to recovery (MTTR) by validating automation and procedures. By identifying weaknesses before they cause customer-impacting incidents, organizations avoid revenue loss, SLA penalties, and emergency response costs:

- FIS eliminates the need to build and maintain custom chaos engineering infrastructure, reducing operational overhead and tooling costs
- The service's precise targeting capabilities minimize unnecessary resource consumption during testing
- Multi-account support reduces the need for multiple testing environments across different teams, consolidating chaos engineering activities and associated costs while improving organizational learning and best practice sharing