Service Name

AWS Fargate

Service Overview

AWS Fargate is a serverless compute engine for containers that enables organizations to run containerized applications without managing servers, clusters, or underlying infrastructure. Compatible with both Amazon Elastic Container Service (ECS) and Amazon Elastic Kubernetes Service (EKS), Fargate allows developers to focus on application development while AWS handles infrastructure provisioning, scaling, and security. Each task or pod runs in its own isolation boundary with dedicated resources, providing enhanced security. Fargate supports both Linux and Windows containers with platform versions for Amazon Linux 2, Bottlerocket, and Microsoft Windows 2019 Server editions.

Key Use Cases

- Microservices Architecture: Deploy and scale microservices without infrastructure management overhead
- Batch Processing Workloads: Run batch jobs and data processing tasks with automatic resource provisioning
- Development and Test Environments: Quickly spin up containerized environments for development and testing
- Web Applications and APIs: Host containerized web applications with seamless scaling based on demand
- Stateful Applications: Run long-running stateful workloads without managing underlying infrastructure

Features

- **Serverless Container Compute**: Run containers without managing servers or clusters, with automatic infrastructure provisioning
- **Task Isolation**: Each task has its own isolation boundary with dedicated CPU, memory, and network resources
- **Automatic Scaling**: Scale applications based on demand without manual intervention or capacity planning
- **Fargate Spot**: Run interrupt-tolerant workloads at up to 70% discount using spare AWS compute capacity
- **Platform Version Support**: Support for Amazon Linux 2, Bottlerocket, and Windows 2019 Server platforms
- **Load Balancer Integration**: Seamless integration with Application Load Balancer and Network Load Balancer

- **Per-Second Billing**: Pay only for resources consumed with per-second billing granularity

Value Proposition

AWS Fargate eliminates the operational overhead of managing container infrastructure, allowing teams to focus on application development rather than server management. Key advantages include reduced total cost of ownership through pay-per-use pricing, enhanced security through task-level isolation, and simplified operations with automatic scaling:

- Fargate provides faster time-to-market by removing infrastructure provisioning delays and offers better resource utilization compared to traditional container deployments
- The service includes built-in high availability, compliance certifications for regulated industries, and seamless integration with the AWS ecosystem, making it ideal for organizations seeking serverless container solutions

Service Integration

AWS Fargate integrates deeply with Amazon ECS and Amazon EKS as the primary orchestration platforms. Core integrations include Amazon VPC for networking, IAM for security and access control, Amazon CloudWatch for monitoring and logging, AWS X-Ray for distributed tracing, and Amazon ECR for container image storage:

- Load balancing is provided through Application Load Balancer and Network Load Balancer
- Storage integrations include Amazon EFS for persistent storage, though Amazon EBS volumes cannot be mounted to Fargate tasks
- Additional integrations encompass AWS Lambda for event-driven architectures, Amazon SNS and SQS for messaging, and AWS Cloud Map for service discovery

Onboarding and Offboarding

Getting started with AWS Fargate requires creating an AWS account and setting up an IAM execution role for task execution. Customers can begin by creating a Fargate-enabled ECS cluster or EKS cluster through the AWS Management Console, AWS CLI, or infrastructure as code tools:

- The process involves defining task definitions with CPU and memory requirements, configuring networking settings including VPC and security groups, and deploying containers using pre-built images from Amazon ECR or other registries
- AWS provides step-by-step tutorials and getting started guides for both ECS and EKS implementations, along with sample applications and CloudFormation templates to accelerate deployment

Getting Started Guide:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/getting-started-fargate.html

Data Removal

Offboarding from AWS Fargate involves stopping all running tasks and services, deleting task definitions, and removing associated resources like load balancers and security groups. Data stored in Fargate containers is ephemeral and automatically deleted when tasks stop. For persistent data, customers must delete associated Amazon EFS file systems, DynamoDB tables, or other storage services. The process includes cleaning up IAM roles, removing CloudWatch log groups, and deleting ECS clusters or EKS clusters used with Fargate.

Backup/Restore and Disaster Recovery

AWS Fargate does not provide built-in backup capabilities as it is a serverless compute service for stateless containers. Data persistence and backup strategies depend on external storage services like Amazon EFS, Amazon S3, or Amazon RDS. Disaster recovery is achieved through multi-region deployments, container image replication across regions, and infrastructure as code practices. Organizations must implement application-level backup strategies and use AWS services like AWS Backup for supported resources.

Backup Capabilities

AWS Fargate itself does not have native backup capabilities since containers are ephemeral. Data backup must be implemented at the application level using AWS storage services. While Fargate tasks can write to Amazon EFS file systems which support AWS Backup integration, the Fargate service does not directly integrate with AWS Backup for task-level backups.

Disaster Recovery

AWS Fargate supports disaster recovery through multi-region deployment patterns but does not directly integrate with AWS Elastic Disaster Recovery. Organizations implement DR by deploying Fargate tasks across multiple Availability Zones and regions, using container image replication, and maintaining infrastructure as code templates for rapid environment recreation during disaster scenarios.

Recovery Objectives

AWS Fargate helps achieve low RTO objectives through rapid task startup capabilities, typically launching new tasks within minutes. RPO objectives depend on application-level data replication strategies to external storage services. The service supports automatic failover through Application Load Balancer health checks and can quickly spin up replacement tasks in different Availability Zones, contributing to overall disaster recovery capabilities.

Service Constraints

Service Quotas: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/service-quotas-manage.html

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

Technical Requirements

Service Documentation: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/AWS_Fargate.html

User Guide: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/getting-started-fargate.html

API Reference: https://docs.aws.amazon.com/AmazonECS/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/fargate/pricing/

Cost Optimization

AWS Fargate offers several unique cost optimization features including Fargate Spot for up to 70% savings on interrupt-tolerant workloads, per-second billing with no minimum commitments, and Compute Savings Plans for consistent usage patterns. Right-sizing recommendations through AWS Compute Optimizer help identify overprovisioned tasks:

- Graviton2-powered Fargate instances provide better price-performance ratios
- The service eliminates infrastructure management costs and reduces operational overhead through serverless architecture, allowing organizations to optimize spending by paying only for actual resource consumption

Savings Considerations

Primary cost savings come from eliminating server management overhead, reducing operational staff requirements, and paying only for consumed resources rather than provisioned capacity. Fargate Spot provides significant discounts for fault-tolerant workloads, while automatic scaling prevents overprovisioning:

- Organizations save on licensing costs by using Linux containers instead of Windows, and Graviton-based instances offer better price-performance

- The serverless model eliminates idle resource costs common in traditional container deployments, and integration with AWS Compute Optimizer enables continuous cost optimization through rightsizing recommendations