Service Name

AWS Batch

Service Overview

AWS Batch is a fully managed batch computing service that helps users run batch computing workloads on the AWS Cloud. It eliminates the undifferentiated heavy lifting of configuring and managing required infrastructure, similar to traditional batch computing software. AWS Batch automatically provisions compute resources and optimizes workload distribution based on the quantity and scale of submitted workloads. The service runs containerized batch ML, simulation, and analytics workloads across various AWS compute offerings including Amazon ECS, Amazon EKS, AWS Fargate, and Spot or On-Demand Instances. AWS Batch provides queuing, scheduling, and priority management capabilities while delivering operational capabilities to optimize workloads for throughput, speed, resource efficiency, and cost.

Key Use Cases

- High Performance Computing (HPC): Running compute-intensive workloads for scientific research, engineering simulations, and complex calculations at scale
- Machine Learning and AI: Training ML models, hyperparameter tuning, and running inference workloads with automatic resource scaling and SageMaker integration
- Financial Services: Post-trade analytics for risk analysis, high-frequency trading simulations, and fraud surveillance using machine learning algorithms
- Life Sciences: Drug screening for molecular libraries, DNA sequencing analysis, and genomics research with large-scale computational requirements
- Digital Media: Video rendering, transcoding workflows, and media supply chain processing for content creation and distribution
- Scientific Research: Monte Carlo simulations, climate modeling, and data processing for research institutions and laboratories

Features

- **Fully Managed Service**: Automatically provisions, scales, and terminates compute resources based on job requirements without manual infrastructure management
- **Multi-Compute Environment Support**: Supports Amazon EC2, AWS Fargate, Amazon EKS, and Spot Instances for flexible compute resource allocation
- **Job Queuing and Scheduling**: Priority-based job scheduling with dependency management and automatic retry capabilities for failed jobs
- **Container Orchestration**: Runs containerized workloads using Docker containers with support for multi-container and multi-node parallel jobs

- **Resource-Aware Scheduling**: Manages consumable resources like license tokens and bandwidth limits to optimize job scheduling and resource utilization
- **SageMaker Integration**: Provides queuing and scheduling capabilities for Amazon SageMaker Training jobs with fair-share scheduling policies
- **Auto Scaling**: Dynamically scales compute resources up and down based on job queue demand and workload requirements
- **Cost Optimization**: Supports EC2 Spot Instances for up to 90% cost savings and integrates with AWS cost management tools

Value Proposition

AWS Batch provides significant advantages over traditional batch computing solutions by eliminating infrastructure management overhead and reducing operational complexity. Customers benefit from automatic scaling that adapts to workload demands without manual intervention, resulting in improved resource utilization and cost efficiency:

- The service integrates seamlessly with existing AWS services and supports multiple compute environments, allowing organizations to optimize performance and costs
- AWS Batch's managed approach reduces the need for specialized infrastructure teams, enabling data scientists and researchers to focus on their core work rather than infrastructure management
- The service also provides built-in fault tolerance, automatic retries, and comprehensive monitoring capabilities that enhance reliability and operational visibility

Service Integration

AWS Batch integrates deeply with core AWS services to provide comprehensive batch computing capabilities. It leverages Amazon EC2 for compute instances, AWS Fargate for serverless containers, and Amazon EKS for Kubernetes-based workloads:

- The service integrates with Amazon ECS for container orchestration and management
- For machine learning workloads, AWS Batch provides native integration with Amazon SageMaker for training job queuing and scheduling
- Storage integration includes Amazon S3 for data input/output, Amazon EFS for shared file systems, and Amazon EBS for persistent storage
- Monitoring and logging capabilities are provided through Amazon CloudWatch for metrics and logs, AWS CloudTrail for auditing, and AWS Trusted Advisor for cost optimization recommendations
- The service also integrates with AWS IAM for access control and AWS VPC for network isolation

Onboarding and Offboarding

Getting started with AWS Batch involves creating core components in sequence: compute environments, job queues, job definitions, and jobs. Customers first set up IAM roles and policies, then create a compute environment specifying the compute resources (EC2, Fargate, or EKS) and configuration:

- Next, they create job queues that reference the compute environment and define scheduling priorities
- Job definitions specify container properties, resource requirements, and execution parameters
- Finally, jobs are submitted to queues for execution
- AWS provides multiple tutorials including wizard-based setup for beginners, CLI-based tutorials for advanced users, and specific guidance for different compute environments
- The onboarding process includes setting up VPC networking, security groups, and IAM permissions
- AWS offers comprehensive documentation, tutorials, and best practices guides to help customers implement batch workloads effectively

Getting Started Guide:
https://docs.aws.amazon.com/batch/latest/userguide/Batch_GetStarted.html

Data Removal

AWS Batch offboarding involves deleting resources in reverse order of creation and removing associated data. Customers must first disable and delete job queues, then disable and delete compute environments. Job definitions and completed job records can be deregistered and removed. Any data stored in Amazon S3 buckets, EBS volumes, or EFS file systems used by batch jobs must be manually deleted. Container images stored in Amazon ECR should be removed if no longer needed. CloudWatch logs and metrics are automatically retained according to configured retention policies but can be manually deleted. IAM roles and policies created for AWS Batch can be removed after confirming no other services depend on them.

Backup/Restore and Disaster Recovery

AWS Batch relies on underlying AWS services for backup and disaster recovery capabilities. Job definitions and compute environment configurations are stored in AWS Batch service metadata and automatically replicated across availability zones. Input and output data stored in Amazon S3 can leverage S3 versioning, cross-region replication, and backup policies. EBS volumes used by compute instances can be backed up using EBS snapshots and AWS Backup service. Container images in Amazon ECR support cross-region replication for disaster recovery. AWS Batch workloads can be recreated in different regions using Infrastructure as Code tools like CloudFormation.

Backup Capabilities

AWS Batch does not provide native backup capabilities but integrates with AWS Backup for underlying resources. The service configuration including job definitions and compute environments is automatically maintained by AWS with high availability:

- Customers can use AWS Backup to protect EBS volumes, EFS file systems, and other storage resources used by batch jobs
- Container images can be backed up through Amazon ECR replication features
- Job execution logs in CloudWatch can be exported to S3 for long-term retention and backup

Disaster Recovery

AWS Batch supports disaster recovery through multi-region deployment capabilities and integration with AWS disaster recovery services. The service can be deployed across multiple AWS regions with job definitions and compute environments recreated using Infrastructure as Code:

- AWS Batch integrates with AWS Elastic Disaster Recovery for comprehensive disaster recovery planning
- Workloads can be designed to automatically failover to secondary regions using AWS services like Route 53 for DNS failover and Application Recovery Controller for automated recovery orchestration

Recovery Objectives

AWS Batch helps customers meet RPO and RTO objectives through automated scaling, multi-region deployment, and integration with AWS backup services. Recovery Point Objectives can be achieved by implementing regular backups of input data using S3 versioning and cross-region replication, plus EBS snapshots for persistent storage. Recovery Time Objectives are supported through rapid resource provisioning, pre-configured compute environments in multiple regions, and automated job restart capabilities. The service's ability to quickly scale compute resources and recreate batch processing environments helps minimize recovery time during disaster scenarios.

Service Constraints

Service Quotas:
https://docs.aws.amazon.com/batch/latest/userguide/service_limits.html

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

Technical Requirements

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

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

API Reference: https://docs.aws.amazon.com/batch/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/batch/pricing/

Cost Optimization

AWS Batch offers unique cost optimization through automatic integration with EC2 Spot Instances for up to 90% cost savings compared to On-Demand pricing. The service automatically manages Spot Instance bidding, replacement, and interruption handling:

- Customers can leverage mixed instance types and allocation strategies to optimize cost and performance
- AWS Batch supports Savings Plans and Reserved Instances for predictable workloads
- Dynamic scaling ensures customers only pay for compute resources when jobs are running, eliminating idle capacity costs
- The service integrates with AWS Cost Explorer and AWS Budgets for cost monitoring and alerts
- Fair-share scheduling and resource-aware scheduling help optimize resource utilization and prevent over-provisioning

Savings Considerations

Primary cost savings come from eliminating the need to manage and maintain dedicated batch processing infrastructure, reducing operational overhead and staffing requirements. Automatic scaling prevents over-provisioning of resources and eliminates idle compute costs during low-demand periods:

- Spot Instance integration provides substantial cost reductions for fault-tolerant workloads with up to 90% savings over On-Demand pricing
- The service's efficient job scheduling and resource allocation minimize wasted compute cycles and improve overall resource utilization
- Multi-tenancy and shared infrastructure reduce per-job costs compared to dedicated solutions
- Integration with AWS cost management tools enables continuous optimization and cost control through monitoring, budgets, and recommendations