Service Name

Amazon API Gateway

Service Overview

Amazon API Gateway is a fully managed service that enables developers to create, publish, maintain, monitor, and secure APIs at any scale. It serves as the front door for applications to access data, business logic, or functionality from backend services including AWS Lambda functions, EC2 instances, other AWS services, and HTTP endpoints. API Gateway supports RESTful APIs, HTTP APIs, and WebSocket APIs, handling critical tasks such as traffic management, authorization, throttling, monitoring, caching, and API version management while providing a unified access point for clients through public or private endpoints.

Key Use Cases

- API management and governance: Centralized platform for creating, deploying, and managing APIs with comprehensive lifecycle support
- Third-party integrations: Secure exposure of backend resources to external partners and developers with OAuth, throttling, and usage tiers
- Serverless applications: Integration with AWS Lambda functions to build serverless architectures with automatic scaling and pay-per-use pricing
- Microservices architecture: Acting as a unified access point for microservices deployed on Amazon ECS, EKS, or EC2 instances
- Mobile and web application backends: Providing HTTP endpoints for mobile apps and web applications with built-in CORS support
- IoT device communication: Enabling secure communication between IoT devices and backend services through WebSocket APIs
- Real-time applications: Supporting bidirectional communication for chat applications, real-time dashboards, and multiplayer games using WebSocket APIs

Features

- **Multiple API Types**: Support for REST APIs, HTTP APIs, and WebSocket APIs with different feature sets and pricing models
- **Integration Options**: Lambda proxy/custom integrations, HTTP proxy/custom integrations, AWS service integrations, mock integrations, and private integrations
- **Security and Authorization**: AWS IAM, Lambda authorizers, Amazon Cognito user pools, OAuth 2.0, OpenID Connect, API keys, and AWS WAF integration
- **Traffic Management**: Request throttling, usage plans, quotas, caching, and burst handling with token bucket algorithm

- **Endpoint Types**: Edge-optimized (CloudFront), Regional, and Private endpoints with VPC integration capabilities
- **Monitoring and Observability**: CloudWatch metrics, X-Ray tracing, access logging, and comprehensive API analytics
- **Developer Experience**: OpenAPI 3.0 support, SDK generation, developer portals, API documentation, and stage management
- **Data Transformation**: Request/response mapping, validation, binary media type support, and CORS configuration

Value Proposition

Amazon API Gateway offers compelling advantages as a fully managed service that eliminates the need for server provisioning and infrastructure management. It provides automatic scaling to handle any load without warm-up limitations, ensuring consistent performance from a few requests per day to thousands per second:

- The service offers comprehensive security features including native OAuth 2.0, IAM integration, and WAF protection, reducing development complexity
- API Gateway's pay-per-use pricing model with no upfront commitments makes it cost-effective, especially with the generous free tier offering one million API calls monthly
- The service integrates seamlessly with the broader AWS ecosystem, particularly Lambda for serverless architectures, while supporting hybrid deployments through private integrations and VPC connectivity

Service Integration

API Gateway integrates natively with numerous AWS services to enable comprehensive solutions. Primary integrations include AWS Lambda for serverless computing, Amazon EC2 and ECS for containerized workloads, and Application/Network Load Balancers for scalable backend services:

- Security integrations encompass AWS IAM for access control, Amazon Cognito for user authentication, and AWS WAF for API protection
- Monitoring capabilities leverage Amazon CloudWatch for metrics and logging, AWS X-Ray for distributed tracing, and AWS Config for compliance tracking
- Storage and database integrations include Amazon S3, DynamoDB, RDS, and other AWS services through direct service proxies
- Infrastructure management utilizes AWS CloudFormation for deployment automation, Route 53 for DNS management, and CloudFront for global content delivery through edge-optimized endpoints

Onboarding and Offboarding

Getting started with API Gateway involves creating an AWS account and accessing the service through the AWS Management Console, AWS CLI, or SDKs. The quickest path is using the HTTP API quick create feature, which automatically configures a Lambda integration, default route, and stage in minutes:

- Users can also import existing OpenAPI 3.0 definitions or build APIs incrementally using the console's guided workflow
- The service offers comprehensive tutorials for different integration types including Lambda functions, HTTP endpoints, and AWS services
- API Gateway provides CloudFormation templates for infrastructure as code deployments and supports automated CI/CD pipelines
- The generous free tier allows users to experiment with one million API calls monthly, making it risk-free to evaluate the service before production deployment

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

Data Removal

API Gateway offboarding involves deleting APIs, stages, deployments, and associated resources through the console, CLI, or APIs. Deleting an API removes all associated resources including methods, integrations, models, and documentation. Custom domain names, VPC links, and usage plans require separate deletion. API keys and client certificates are automatically cleaned up when no longer referenced. CloudWatch logs and X-Ray traces persist beyond API deletion and require separate cleanup. CloudFormation stacks can automate complete resource cleanup. The service provides immediate deletion for most resources, with some propagation delays for edge-optimized APIs due to CloudFront distribution updates.

Backup/Restore and Disaster Recovery

API Gateway automatically maintains service resilience through AWS's multi-AZ architecture but doesn't provide traditional backup capabilities for API configurations. Disaster recovery relies on API definition exports using OpenAPI specifications, CloudFormation templates, or SDK-based configuration exports. These can be version-controlled and stored in S3, Git repositories, or other systems for restoration. Cross-region deployment requires recreating APIs in target regions using exported definitions. The service supports blue-green deployments through stages and canary releases for safe updates. Regional failures require manual failover to pre-deployed APIs in other regions using Route 53 health checks.

Backup Capabilities

API Gateway does not integrate with AWS Backup service and lacks built-in backup functionality for API configurations. However, API definitions can be exported as OpenAPI specifications, CloudFormation templates, or programmatically through APIs for version control and restoration:

- These exports capture API structure, methods, integrations, models, and documentation but not runtime data like logs or metrics
- Users must implement custom backup strategies using these export mechanisms combined with infrastructure as code practices for comprehensive API configuration preservation and recovery capabilities

Disaster Recovery

API Gateway does not integrate with AWS Elastic Disaster Recovery service as it's a managed API proxy service without persistent storage requirements. DR strategies focus on multi-region API deployment using exported configurations and automated failover mechanisms:

- The service inherits AWS's regional resilience but requires manual intervention for cross-region failover
- Application Recovery Controller (ARC) can automate traffic routing between regions
- DR planning involves maintaining synchronized API definitions across regions, implementing health checks, and establishing automated deployment pipelines for rapid recovery

Recovery Objectives

API Gateway supports various RPO and RTO objectives through architectural design patterns. For low RTO requirements, multi-region active-active deployments with Route 53 health checks enable sub-minute failover. Warm standby approaches using pre-deployed APIs in secondary regions can achieve RTOs of 5-15 minutes. Cold backup and restore strategies using CloudFormation or OpenAPI exports typically require 30-60 minutes for full recovery. RPO is primarily determined by configuration change frequency since API Gateway doesn't store application data, with exports providing point-in-time configuration recovery capabilities.

Service Constraints

Service Quotas:
https://docs.aws.amazon.com/apigateway/latest/developerguide/limits.html

FAQ: https://aws.amazon.com/api-gateway/faqs/

Technical Requirements

Service Documentation: https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html

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

API Reference: https://docs.aws.amazon.com/apigateway/latest/api/

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/api-gateway/pricing/

Cost Optimization

API Gateway offers several cost optimization strategies starting with choosing HTTP APIs over REST APIs for simpler use cases, as HTTP APIs cost significantly less per million requests. Implementing response caching reduces backend calls and associated Lambda or compute costs:

- Request throttling and usage plans prevent unexpected cost spikes from traffic bursts
- Direct integration with AWS services eliminates intermediate Lambda functions for simple operations
- Regional endpoint selection over edge-optimized reduces CloudFront charges when serving localized traffic
- For private integrations, the new direct Application Load Balancer integration eliminates Network Load Balancer costs
- Efficient API design with fewer resources and consolidated endpoints reduces complexity and potential charges
- Monitoring usage patterns helps identify opportunities for caching, throttling adjustments, and architecture optimization

Savings Considerations

Primary cost savings come from API Gateway's serverless model eliminating server management overhead and capacity planning complexities. The pay-per-use pricing means customers only pay for actual API calls, avoiding fixed infrastructure costs:

- The generous free tier provides one million API calls monthly for 12 months, supporting development and small production workloads

- Caching capabilities can dramatically reduce backend compute costs by serving repeated requests from cache
- Direct AWS service integrations bypass Lambda costs for simple operations like S3 object retrieval or DynamoDB queries
- Efficient throttling prevents cost overruns during traffic spikes while maintaining service availability
- Consolidating multiple APIs reduces management overhead and potential duplicate charges
- Regional endpoint deployment eliminates unnecessary CloudFront costs for local traffic patterns, while still maintaining AWS's reliability and security benefits