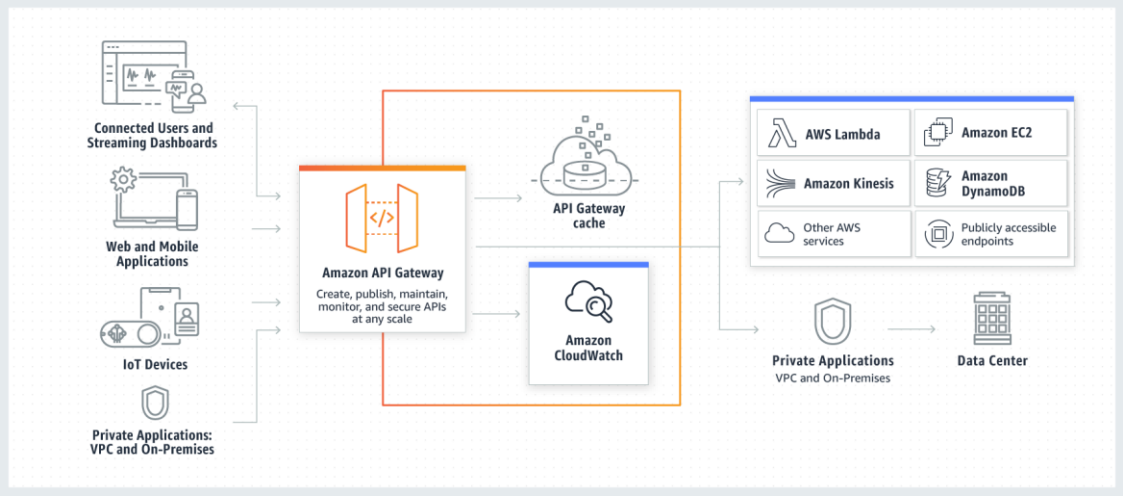
Red River

Amazon API Gateway

Amazon API Gateway

As businesses expand their customer base geographically, it is essential that customers have the same user experience regardless of their location. Red River helps customers create, publish, maintain, monitor, and secure their Application Programming Interfaces (APIs) at any scale. Amazon API Gateway is a fully managed service that makes it easy for developers to integrate applications within the AWS cloud. APIs act as the "front door" for applications to access data, business logic, or functionality from your backend services. Using API Gateway, Customers can create RESTful APIs and WebSocket APIs that enable real-time two-way communication applications. API Gateway supports containerized and serverless workloads, as well as web applications.



Features of Amazon API Gateway

1) Metering – Helps the customer define plans that meter and restrict third-party developer access to your APIs. Customers can define a set of plans, configure throttling, and quota limits on a per API key basis. API Gateway automatically meters traffic to the customers APIs and lets you extract utilization data for each API key
2) Security – provides you with multiple tools to authorize access to your APIs and control service operation access. API Gateway allows you to leverage AWS administration and security tools, such as AWS Identity and Access Management (IAM) and Amazon Cognito, to authorize access to your APIs. API Gateway can verify signed API calls on your behalf using the same methodology AWS uses for its own APIs. Using custom authorizers written as AWS Lambda functions, API

Gateway can also help you verify incoming bearer tokens, removing authorization concerns from your backend code.

3) Resiliency – API Gateway helps you manage traffic with throttling so that backend operations can withstand traffic spikes. API Gateway also helps you improve the performance of your APIs and the latency your end users experience by caching the output of API calls to avoid calling your backend every time.
4) Operations Monitoring – After an API is published and in use, API Gateway provides you with a metrics dashboard to monitor calls to your services. The API Gateway dashboard, through integration with Amazon CloudWatch, provides you with backend performance metrics covering API calls, latency data and error rates. You can enable detailed metrics for each method in your APIs and receive error, access or debug logs in CloudWatch Logs.
5) Lifecycle Management – After an API has been published, you often need to build and test new versions that enhance or add new functionality. API Gateway lets you operate multiple API versions and multiple stages for each version simultaneously so that existing applications can continue to call previous versions after new API versions are published.

Benefits of Amazon API Gateway

1) Efficient API development - Run multiple versions of the same API simultaneously with API Gateway, allowing you to quickly iterate, test, and release new versions. You pay for calls made to your APIs and data transfer out and there are no minimum fees or upfront commitments.
2) Performance at any scale - Provide end users with the lowest possible latency for API requests and responses by taking advantage of our global network of edge locations using Amazon CloudFront. Throttle traffic and authorize API calls to ensure that backend operations withstand traffic spikes and backend systems are not unnecessarily called.
3) Cost savings at scale - API Gateway provides a tiered pricing model for API requests. With an API Requests price as low as $0.90 per million requests at the highest tier, you can decrease your costs as your API usage increases per region across your AWS accounts.
4) Easy monitoring - Monitor performance metrics and information on API calls, data latency, and error rates from the API Gateway dashboard, which allows you to visually monitor calls to your services using Amazon CloudWatch.
5) Flexible security controls - Authorize access to your APIs with AWS Identity and Access Management (IAM) and Amazon Cognito. If you use OAuth tokens, API Gateway offers native OIDC and OAuth2 support. To support custom authorization requirements, you can execute a Lambda authorizer from AWS Lambda.
6) RESTful API options - Create RESTful APIs using HTTP APIs or REST APIs. HTTP APIs are the best way to build APIs for a majority of use cases—they're up to 71% cheaper than REST APIs. If your use case requires API proxy functionality and management features in a single solution, you can use REST APIs.

Use Cases:

1) Create HTTP APIs
2) Create REST APIs
3) WebSocket APIs

Case Studies:

1. Large State Agency

Challenge:

Our client wanted to migrate their royalty and compliance application from an on-premises solution to the cloud and ensure a seamless user experience. The customer also had a need to retain and protect files uploaded by its users as well as meet NIST-compliance high availability and security standards.

Solution:

Amazon API Gateway is a crucial component in the overall cloud architecture both from a security and scalability perspective. From a security perspective, its frontends the Network Load Balancer (NLB), which manages middle-tier backend services. It has associated WAF (Web Access Firewall) which protects from cross-site scripting and SQL injection types of attacks. Further, the Amazon API Gateway endpoint can only be accessed by CloudFront origin that hosts the corresponding User Interface (UI) application. This reduces the access footprint to only the UI application. Finally, each REST service requested is checked for validity using AWS Lambda OKTA authorizers to check for validity of the authorize token which is part of the request payload.

Outcome:

1. Delivered an end-to-end secured workload
2. Highly scalable solution
3. Improved application security

2. Public Agency SaaS Solution Provider

Challenge

Our client wanted to migrate their application from the commercial cloud to the AWS GovCloud for improved security and compliance as well as new market to sell their solution to.

Solution:

The application is an N-tier application consisting of a User Interface (static JavaScript application hosted on AWS S3 with web hosting enabled), middle tier node-based application, and PostgreSQL database.

Outcome:

1. Improved application security to better meet the requirements of their customers.
2. Application will scale as the client grows
3. User experience will be the same regardless of their geography.