Amazon API Gateway

Overview

Eren Akbaba

Solutions Architect, AWS
eakbaba@amazon.com

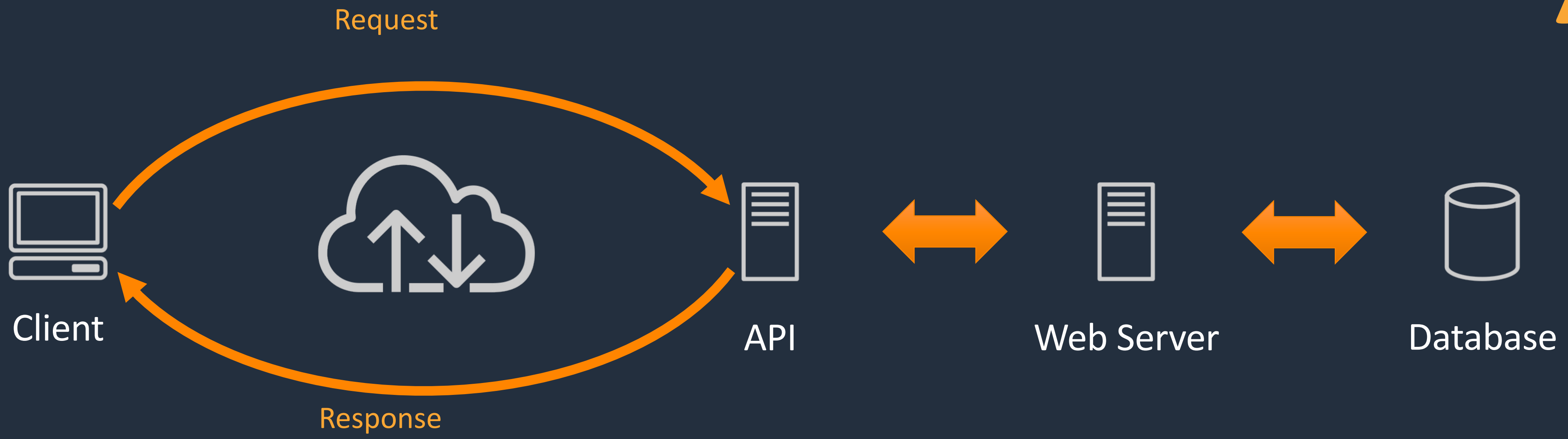
Agenda

- What is API Gateway
- API Types
- Integrations
- Protecting APIs
- Validation and Transformation
- Stages and Versioning
- Custom Domain Names
- Observability
- Other Features
- Pricing
- Best Practices

What is API Gateway?

Application Programming Interface (API)

“In building applications, an API simplifies programming by abstracting the underlying implementation and only exposing objects or actions the developer needs.”



Web-based companies and services offer APIs for developers to use, such as:

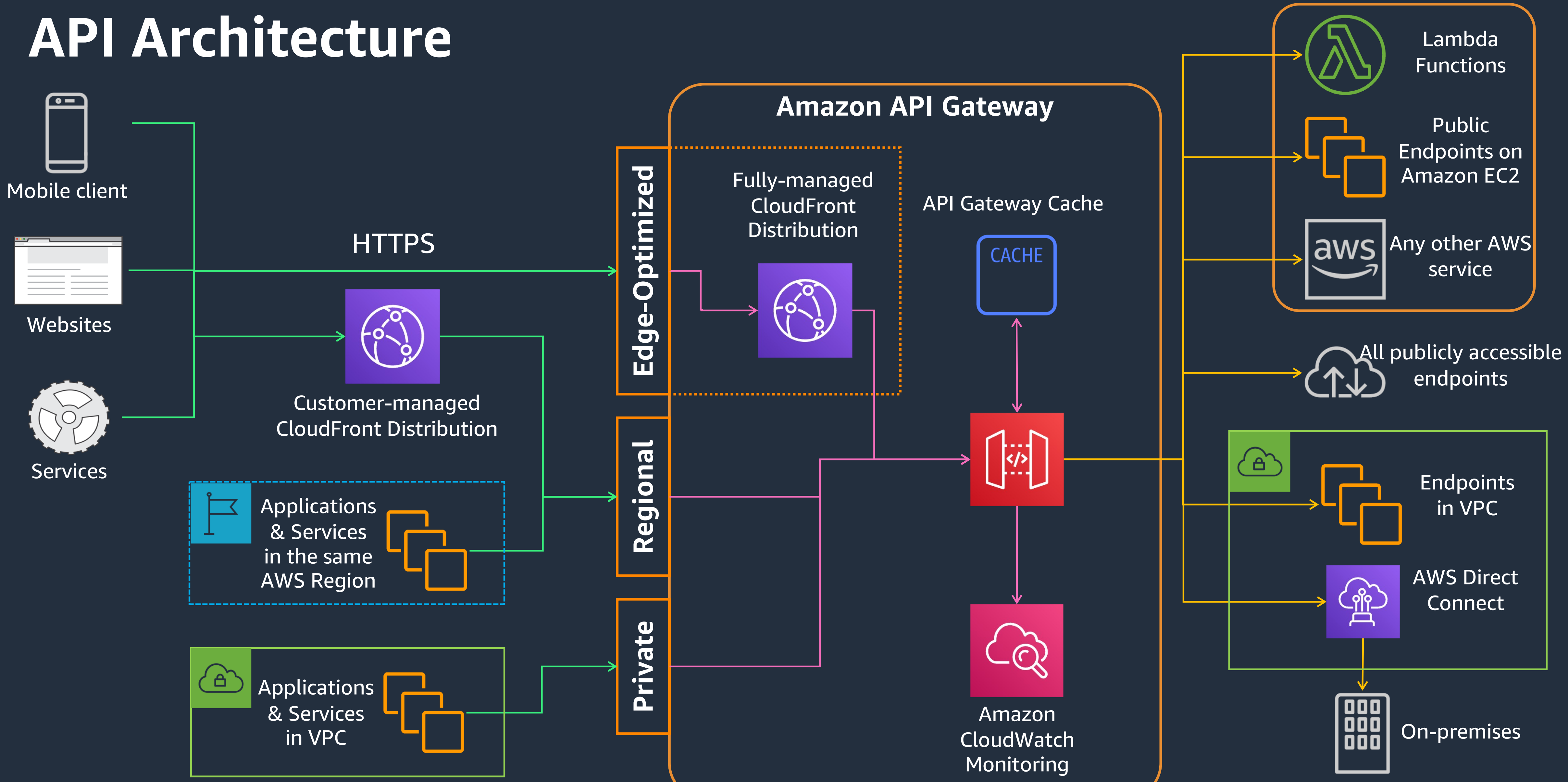
- Social Networks – Facebook, Twitter, etc
- Payment Processing – Amazon Pay, PayPal, etc

https://en.wikipedia.org/wiki/Application_programming_interface

Amazon API Gateway

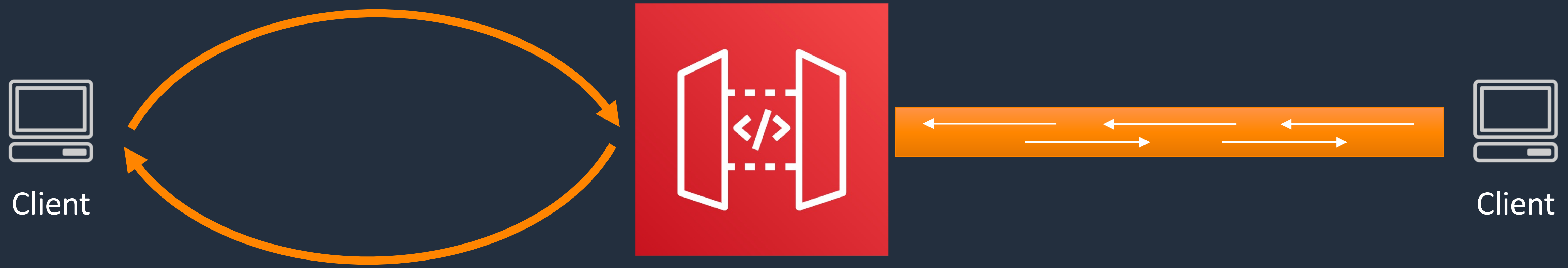
Amazon API Gateway is a fully managed (serverless) service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale.

API Architecture


Mobile client
Websites
Services
HTTPS
Customer-managed CloudFront Distribution
Applications & Services in the same AWS Region
Applications & Services in VPC
Edge-Optimized
Regional
Private
Amazon API Gateway
Fully-managed CloudFront Distribution
API Gateway Cache
CACHE
Amazon CloudWatch Monitoring
Lambda Functions
Public Endpoints on Amazon EC2
aws
Any other AWS service
All publicly accessible endpoints
Endpoints in VPC
AWS Direct Connect
On-premises

API Types

Supported Protocols



- Request / Response
- HTTP Methods like GET, POST, etc
- Short-lived communication
- Stateless

- Serverless WebSocket
- 2 way communication channel
- Long-lived communication
- Stateful

RESTful APIs

- Two flavors: REST API (v1) and HTTP API (v2)
- REST API is more feature rich, feature parity in HTTP API will take some time.
- HTTP API is built from the ground up:
  - Faster – up to 60% faster
  - Lower cost – up to 71% less expensive
  - Easier to use

Endpoint types

REST

Edge-Optimized

- Utilizes CloudFront to reduce TLS connection overhead (reduces roundtrip time)
- Designed for a globally distributed set of clients

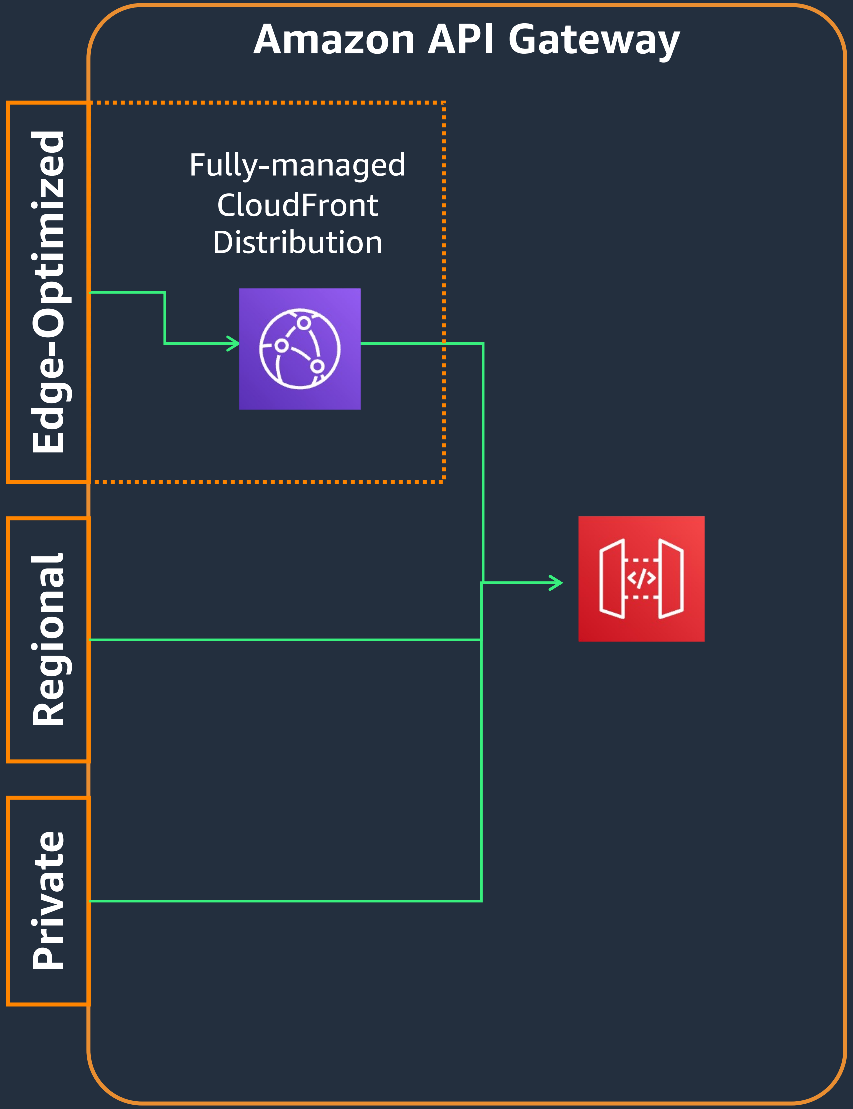
Regional

- Recommended API type for general use cases
- Designed for building APIs for clients in the same region

REST

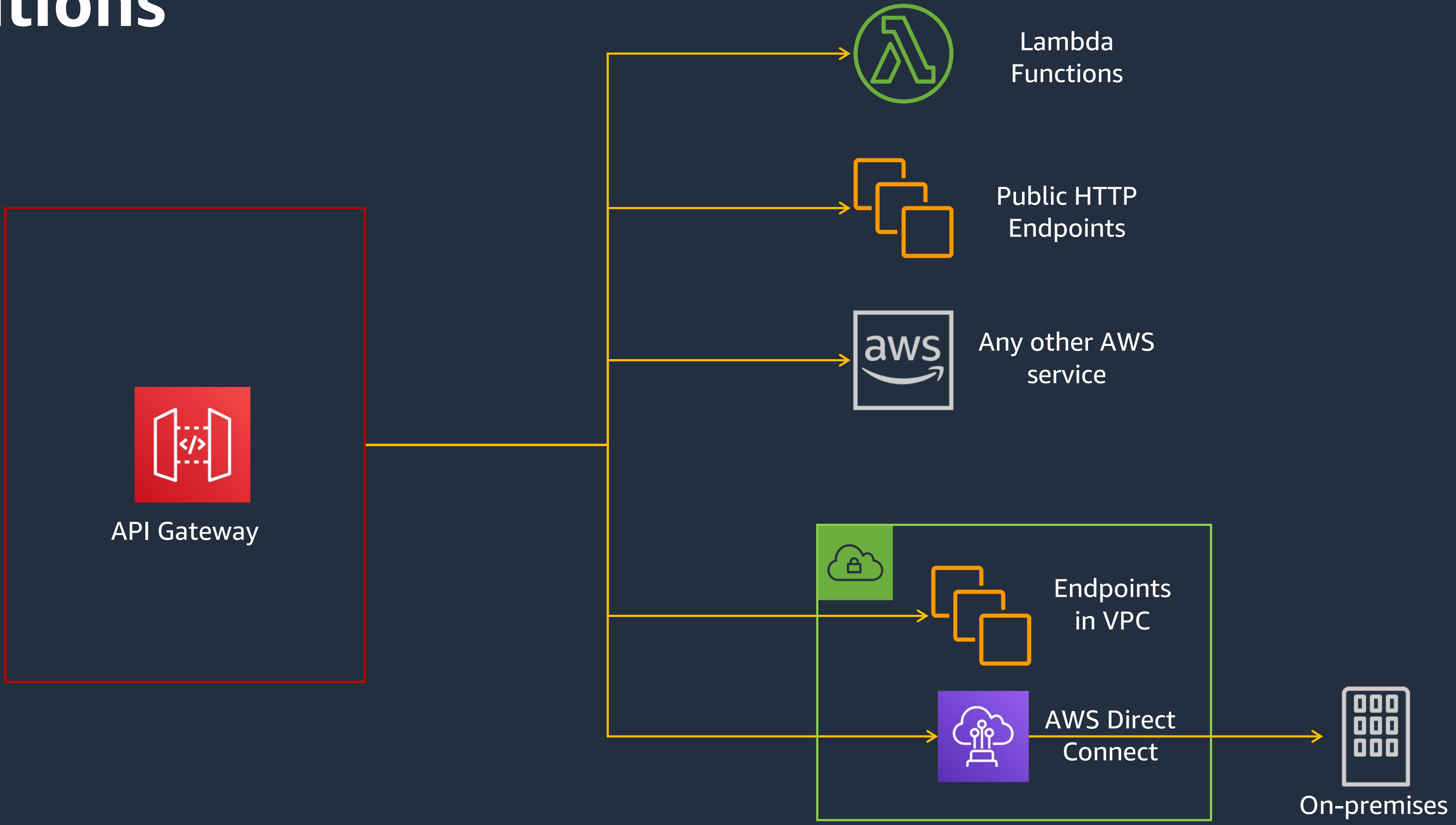
Private

- Only accessible from within VPC (and networks connected to VPC)
- Designed for building APIs used internally or by private microservices

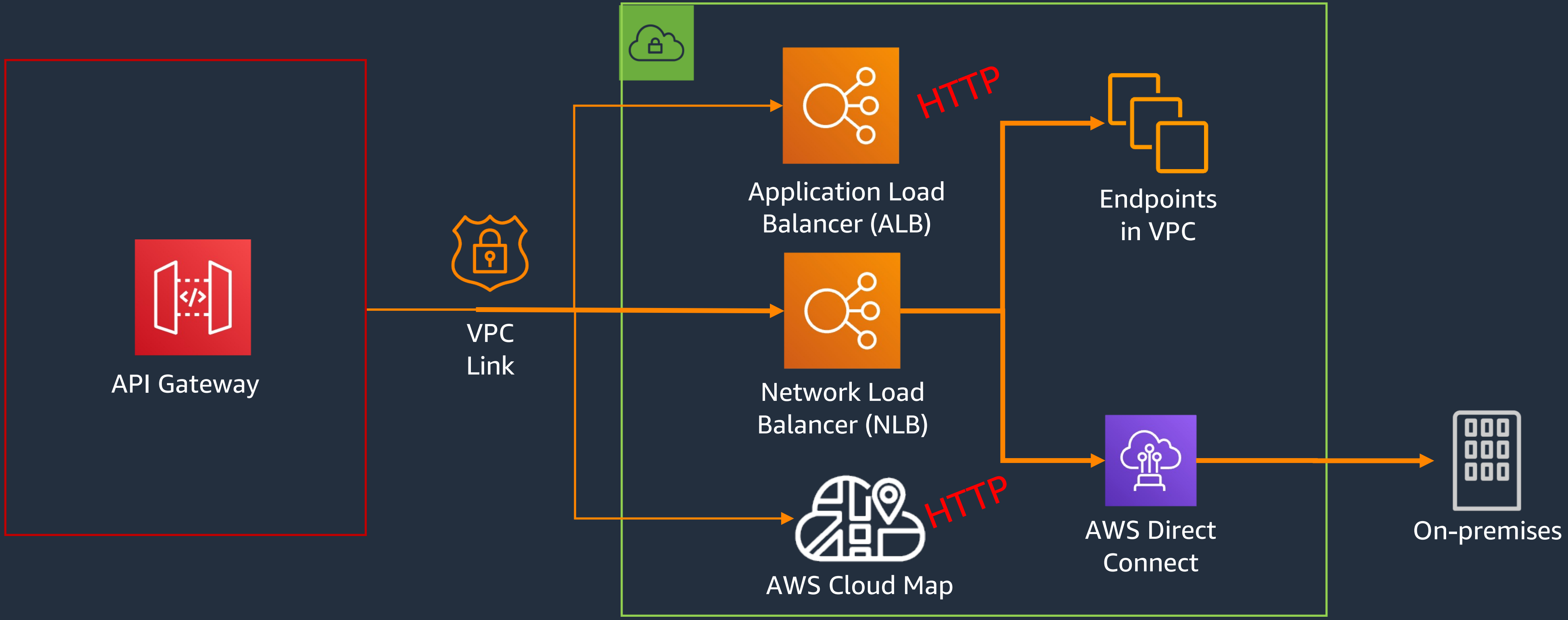


Integrations

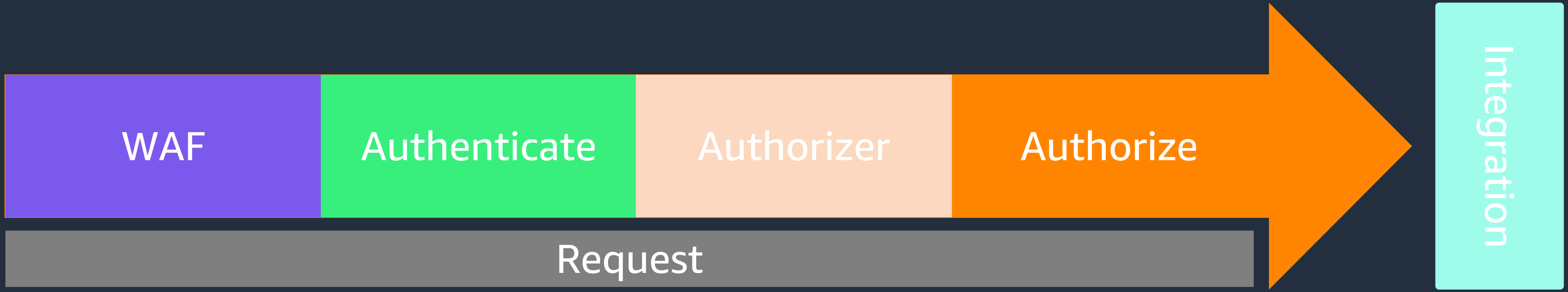
Integrations



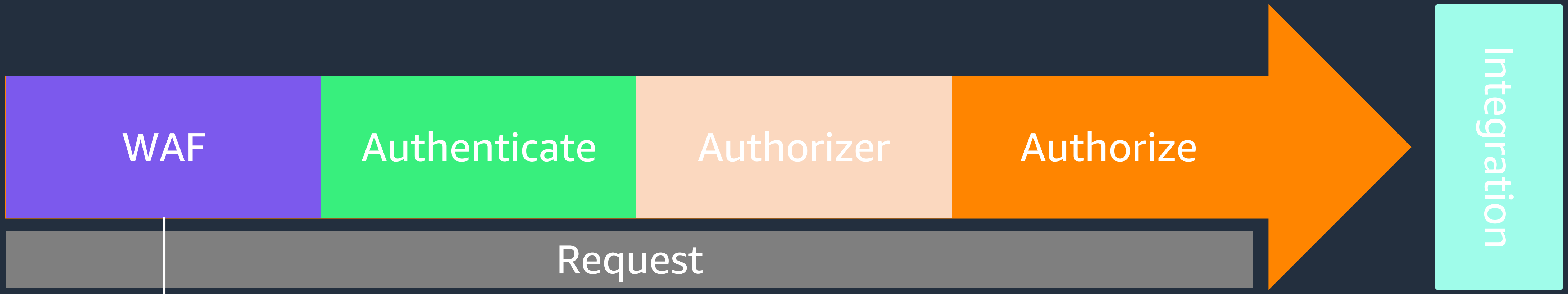
Private Integrations – VPC Link



The request cycle

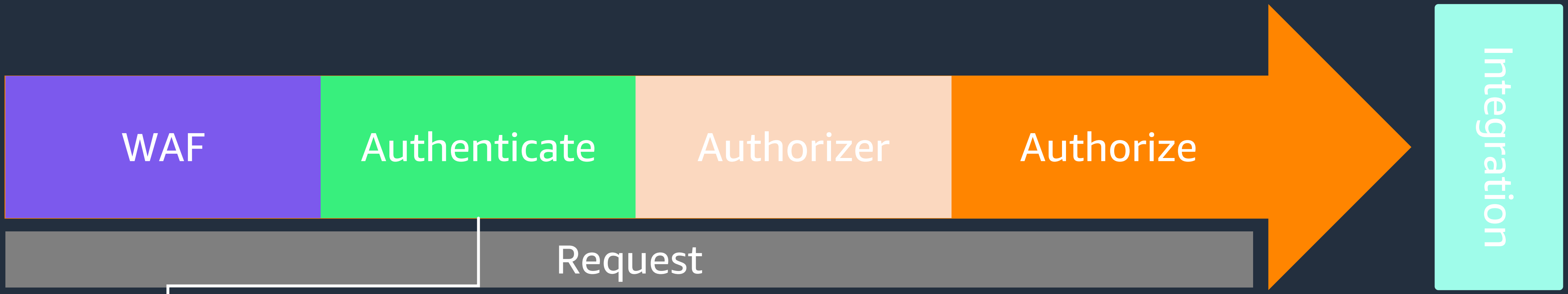


The request cycle



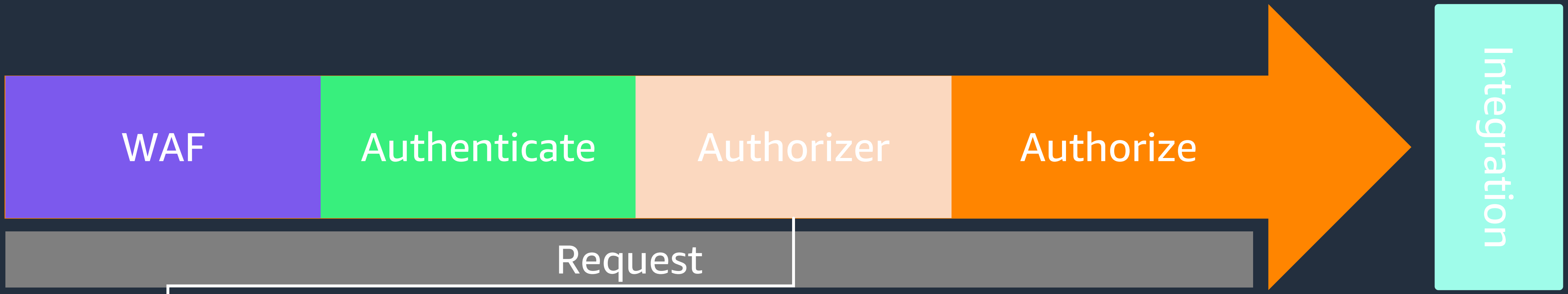
Only appears when an AWS WAF web access control list (ACL) is configured for enhanced security. During this phase, AWS WAF rules are evaluated and a decision is made on whether to continue or cancel the request.

The request cycle



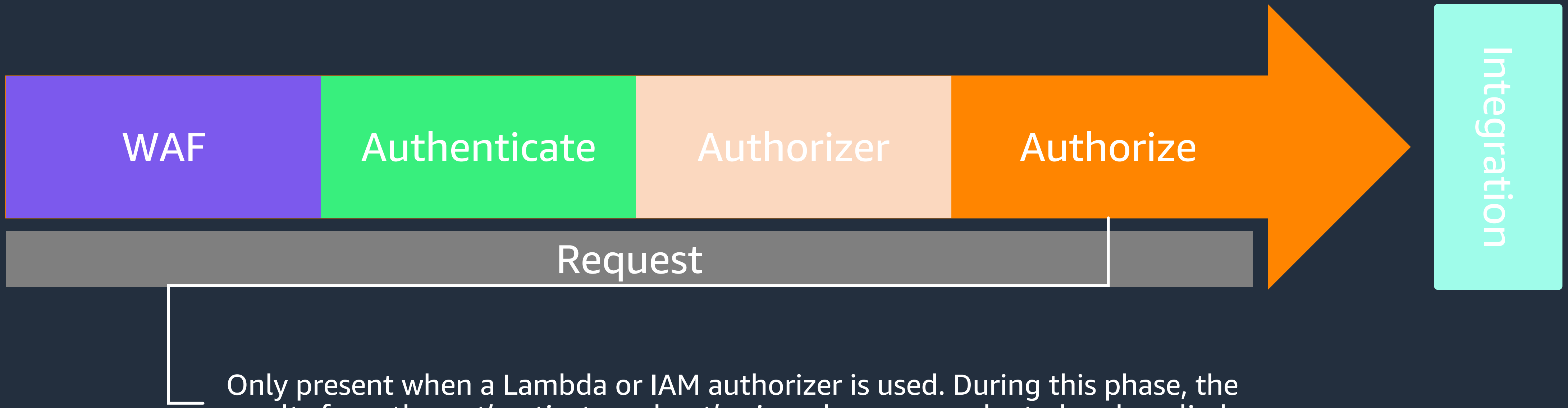
Only present when AWS Identity and Access Management (IAM) authorizers are used. During this phase, the credentials of the signed request are verified. Access is granted or denied based on the client’s right to assume the access role.

The request cycle



Only present when a Lambda, JWT, or Amazon Cognito authorizer is used. During this phase, the authorizer logic is processed to verify the user's right to access the resource.

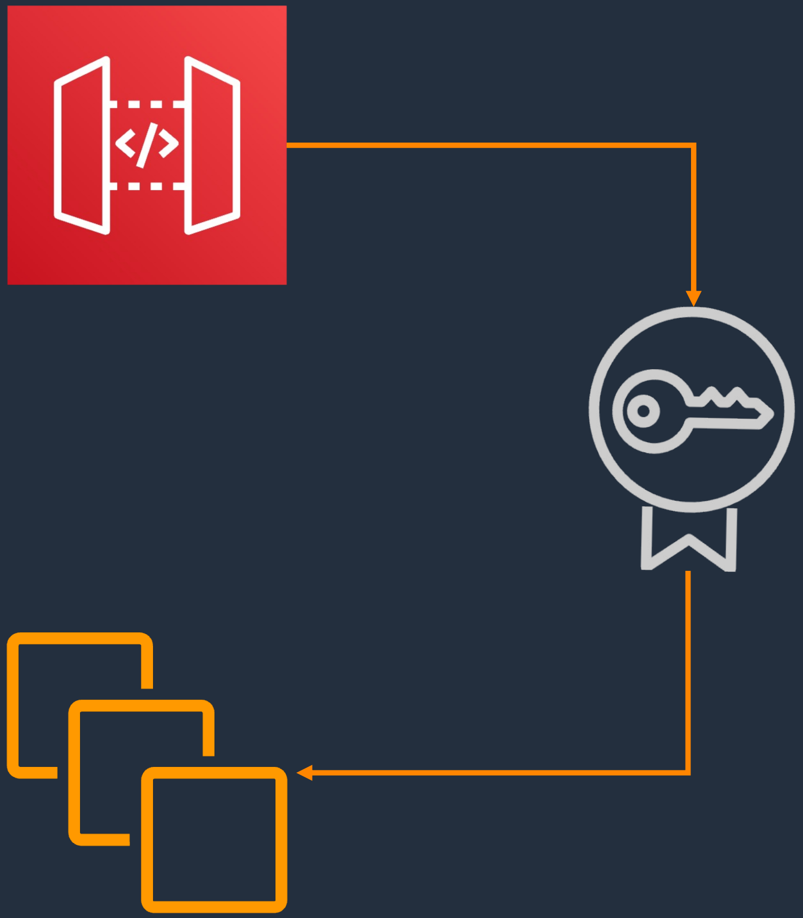
The request cycle



Only present when a Lambda or IAM authorizer is used. During this phase, the results from the *authenticate* and *authorizer* phase are evaluated and applied.

REST

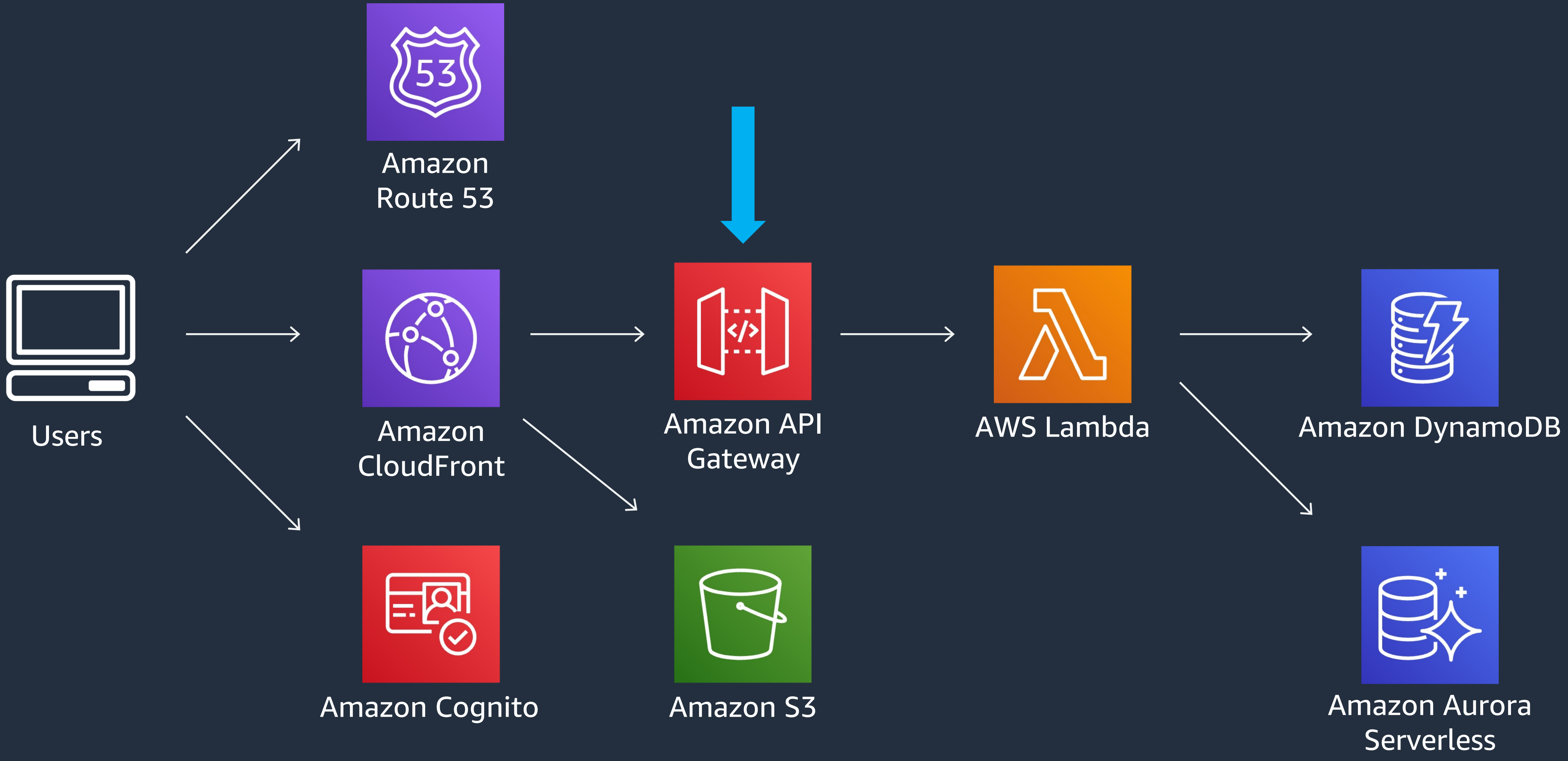
Client Certificates



- Generate client-side SSL certificate using the API Gateway
- Allow backend to verify request coming from API Gateway using public key
- Expires after 365 days

Protecting APIs

API Security with Amazon API Gateway



Types of authorization

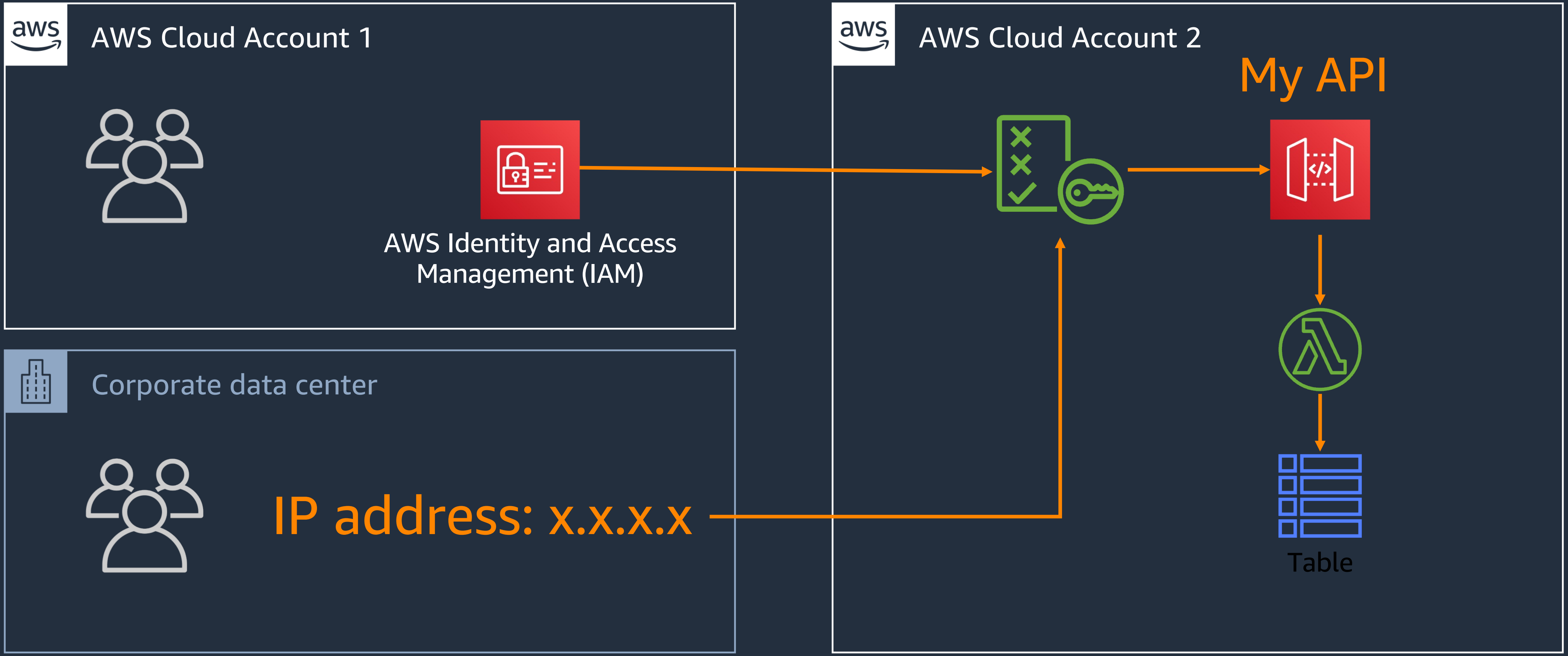
- REST Resource Policy
- Mutual TLS
- REST WAF

+

- REST Cognito User Pools
- HTTP JWT
- IAM
- Lambda Authorizer

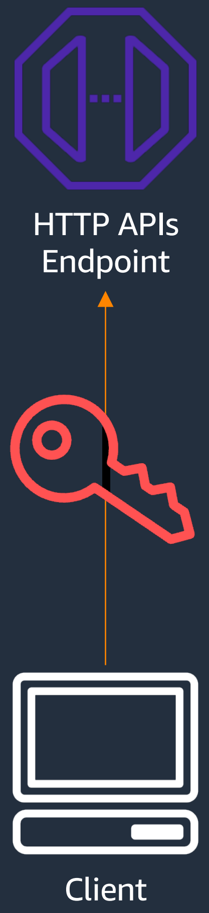
REST

Resource Policies



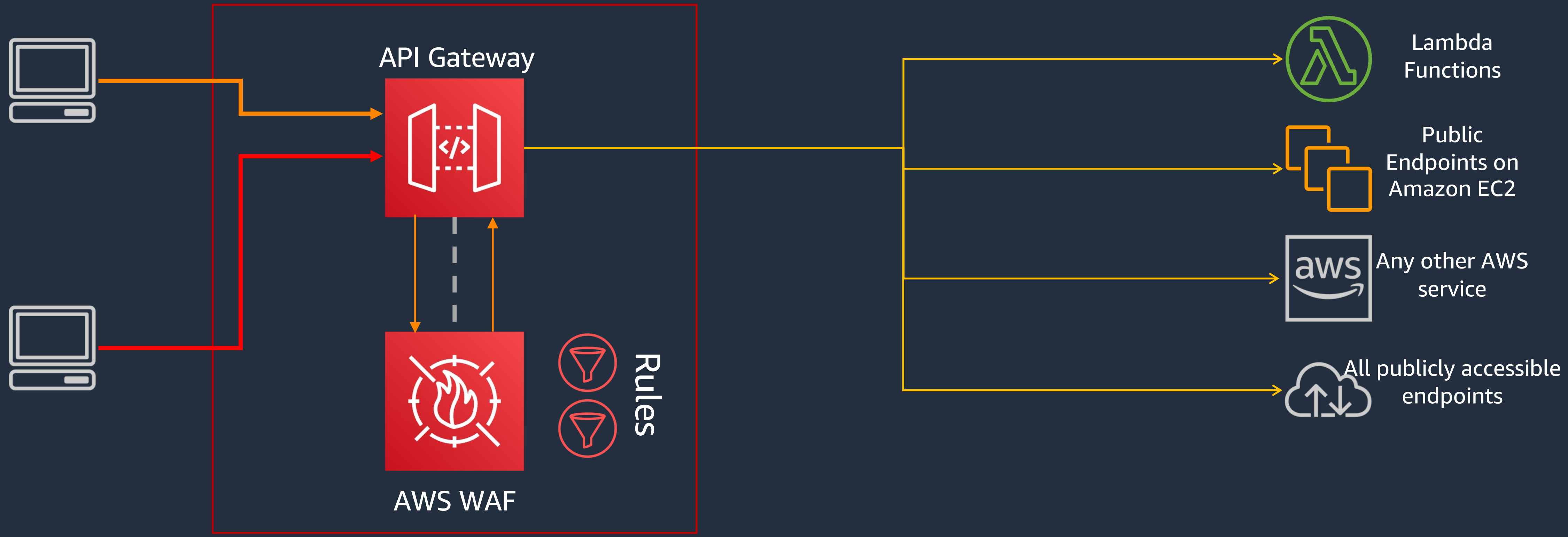
Mutual TLS

- Requires two-way authentication between client and server
- Client must present X.509 certificates to verify identity
- Often used in Internet of Things (IoT) and business to business applications
- Only supported on Custom Domains



REST

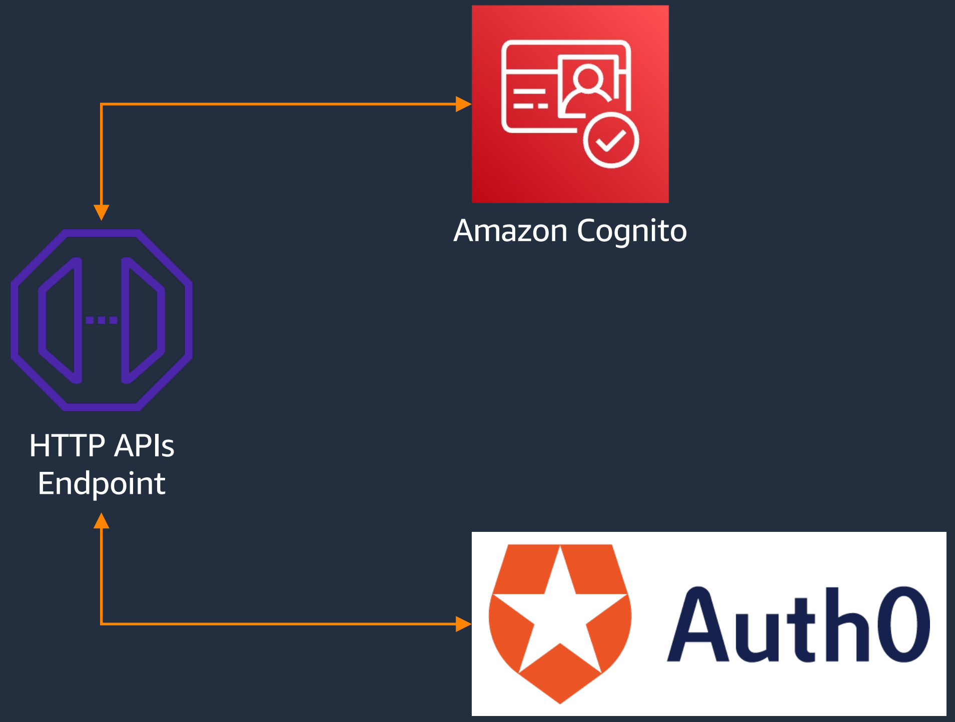
Web Application Firewall (WAF)


API Gateway
AWS WAF
Rules
Lambda Functions
Public Endpoints on Amazon EC2
aws
Any other AWS service
All publicly accessible endpoints

HTTP REST

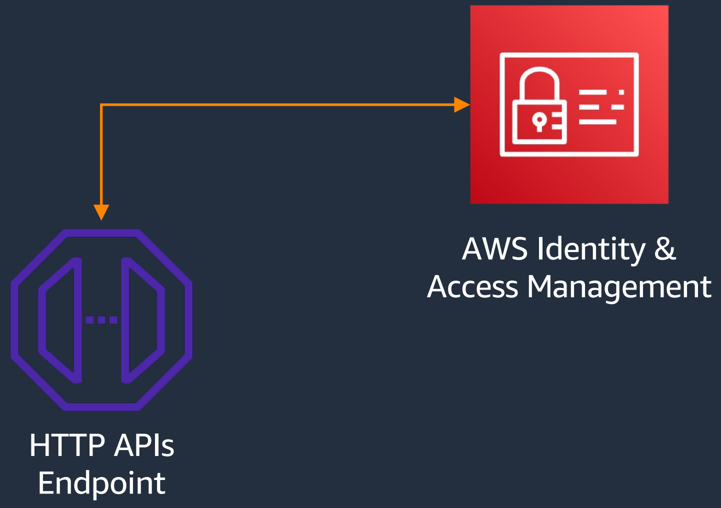
JWT/Cognito authorizer

- OAuth2 compliant (part of OpenID Connect - OIDC)
- Allows or denies access based on token validity and optional scopes
- Any required scopes for the route are validated in the token



IAM authorizer

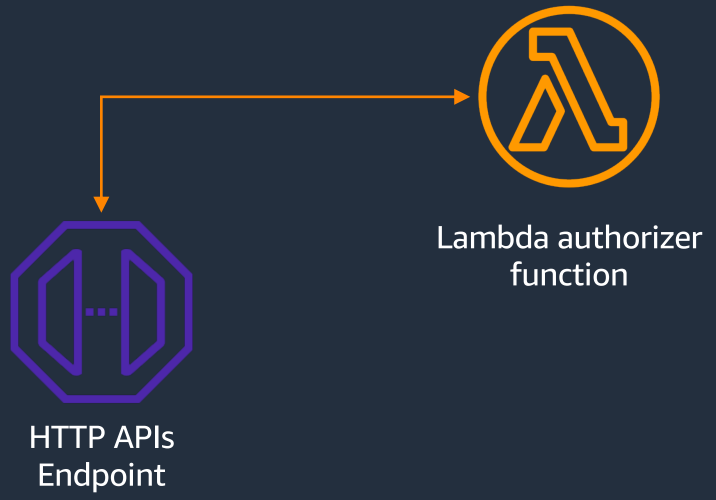
- Clients must use Signature Version 4 to sign their requests with AWS credentials
- Authorization token is decoded
- User is verified against the Identity & Access Management (IAM) service
- User must have *execute-api* access on the route to proceed



Lambda authorizer

- Your custom logic to validate the request
- 2 payload options
  - Payload 1: must return an IAM policy that allows or denies access to your API route
  - Payload 2: can return IAM policy or Boolean
- Authorization can be cached

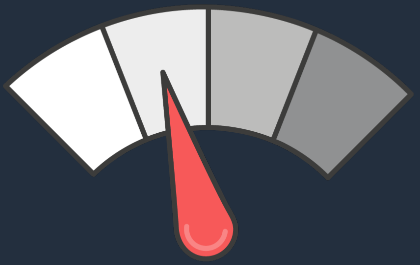
HTTP



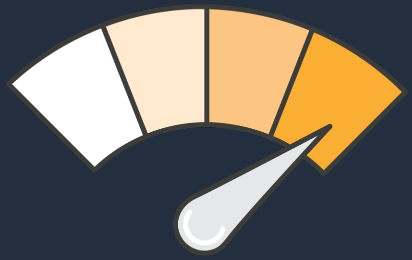
Throttling and REST usage plans

- Protect backend systems
- Prevents one customer from consuming all your backend system's capacity
- Let's you decide how to allocate capacity among your API consumers with quotas and request rates.

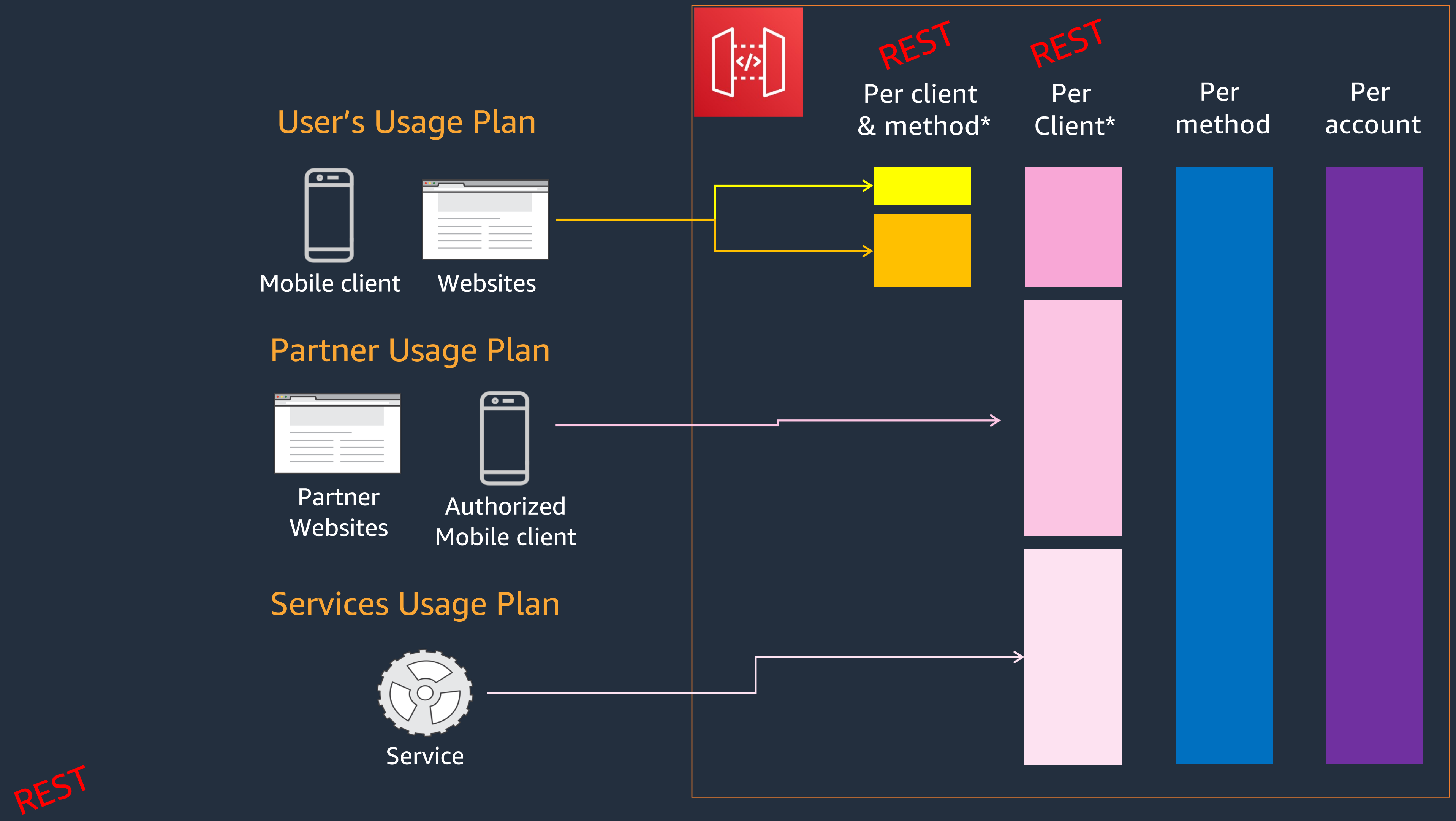
Professional plan users:
10 RPS, up to 100 calls / day



Enterprise plan users:
500 RPS, no limit on calls / day



Four levels of Throttling



* Requires API Keys and Usage Plans

REST

Usage Plans and API Keys

API Keys

- Alphanumeric string values that you distribute to clients (per user/client)
- Generated by API Gateway or you can imported from a CSV file
- Use API keys together with Usage Plans or Lambda authorizers to control access to your APIs

Usage Plans

- Specifies who can access API stages and methods
- How much and how fast they can access the resource
  - Rate limit
  - Quota limit
- Uses API keys to identify API clients

REST

Validation and Transformation

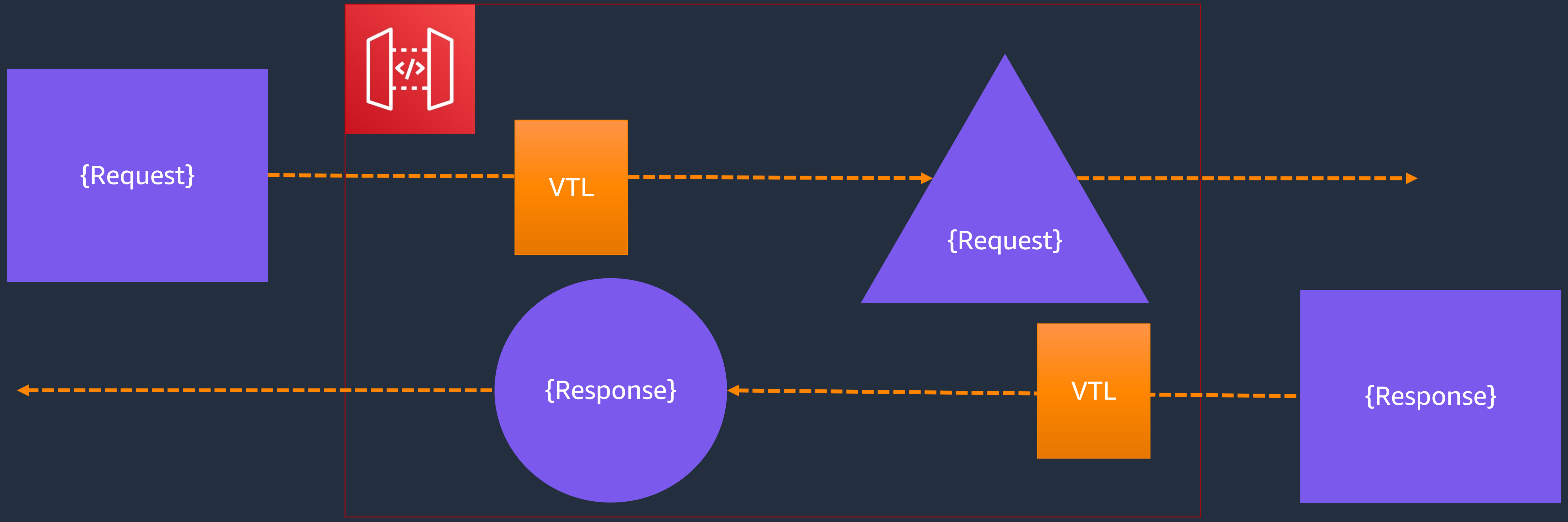
REST

Validation

- The required request parameters in the URI, query string, and headers of an incoming request are included and non-blank.
- The applicable request payload adheres to the configured JSON schema request model of the method.

REST

Transformation - Mapping



Stages & Versions

Stages

API Gateway enables you to set stage variables, allowing the same API to point to different backends.

Your APIs are versioned and can be rolled back.

- APIs are deployed to staging environments.
  - You choose what to name them.
- For example, these environments:
  - Dev (e.g., example.com/dev)
  - Beta (e.g., example.com/beta)
  - Prod (e.g., example.com/prod)

Stages

API Gateway

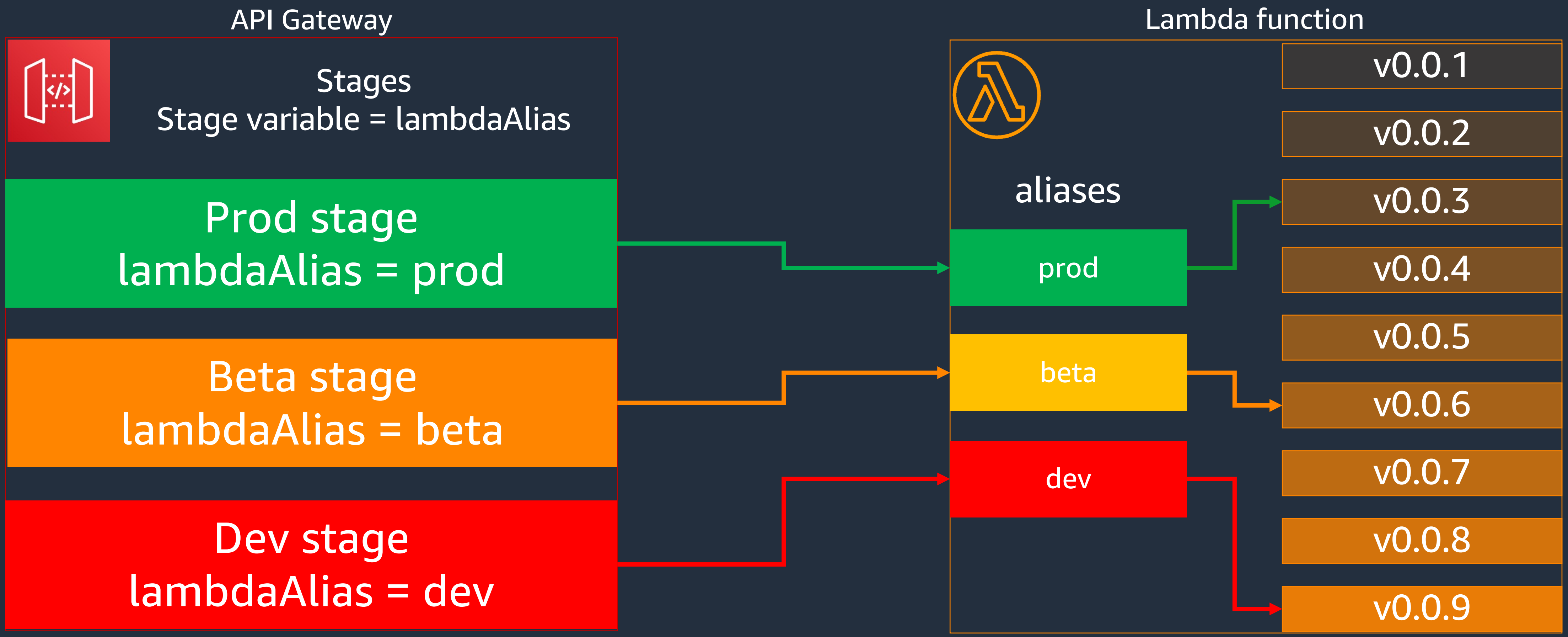
Stages
Stage variable = lambdaAlias

Prod stage
lambdaAlias = prod

Beta stage
lambdaAlias = beta

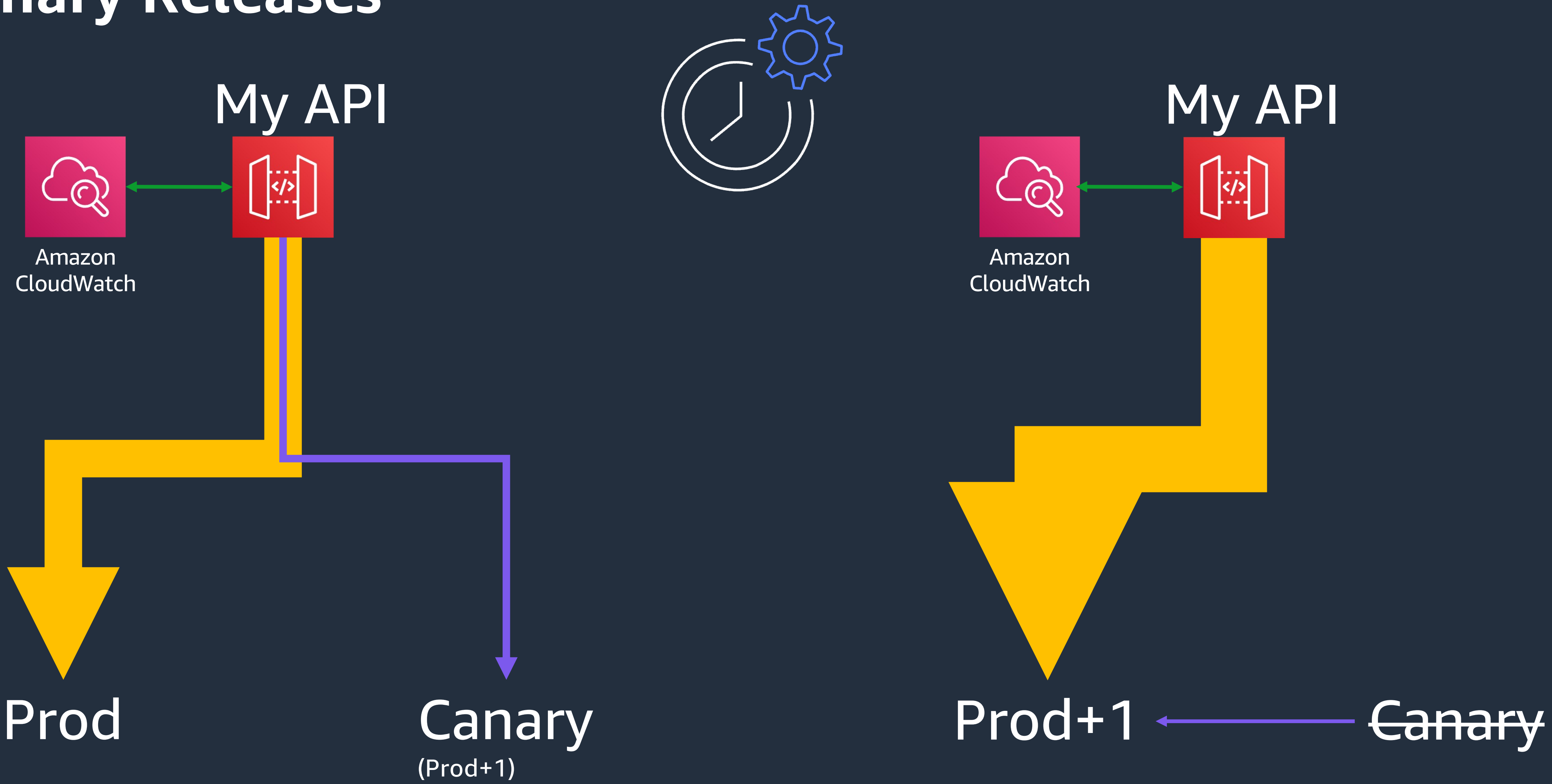
Dev stage
lambdaAlias = dev

Stages


API Gateway
Stages
Stage variable = lambdaAlias
Prod stage
lambdaAlias = prod
Beta stage
lambdaAlias = beta
Dev stage
lambdaAlias = dev
Lambda function
aliases
prod
beta
dev
v0.0.1
v0.0.2
v0.0.3
v0.0.4
v0.0.5
v0.0.6
v0.0.7
v0.0.8
v0.0.9

REST

Canary Releases

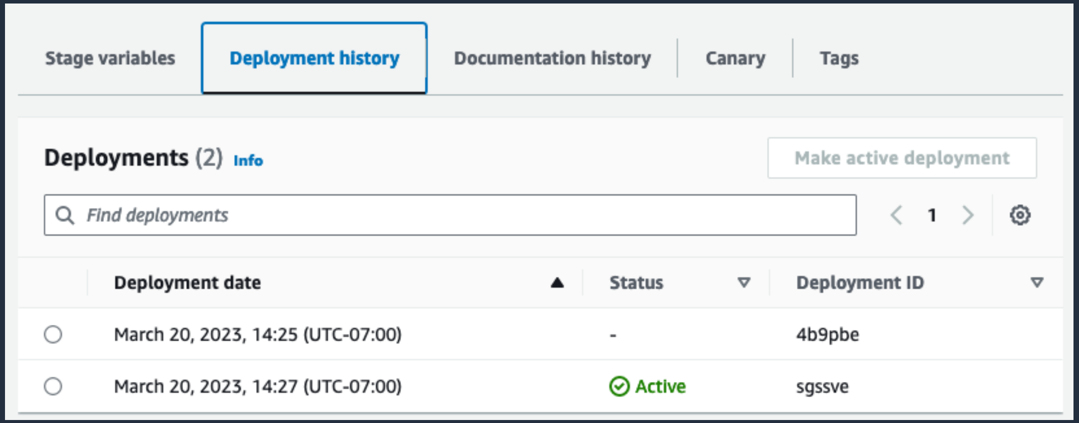


REST

API Versioning

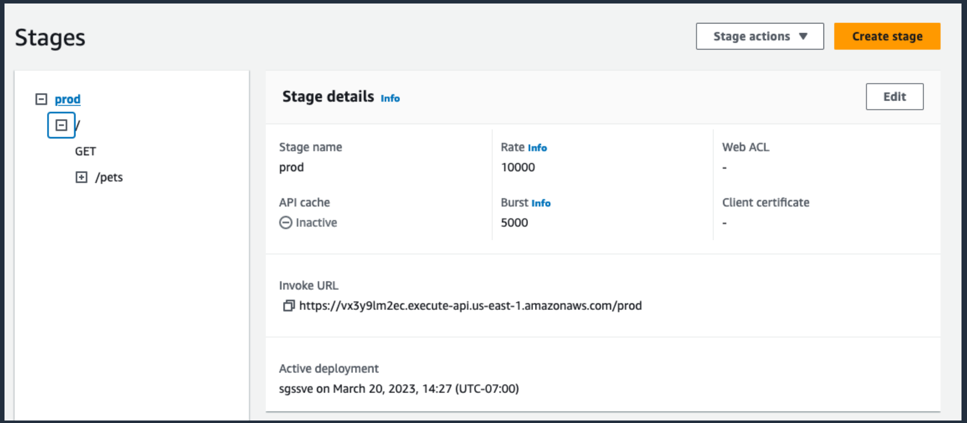
We support versioning inside the API Console allowing to easily roll back to a previous API version and deploy it.

- Easily roll back to different snapshots of the same API (“Deployments”).
- Each Stage points to a Deployment. You can update the Stage to point to a previous Deployment.
- Permits Canary Deployments.



Deployments (2) Info

	Deployment date	Status	Deployment ID
○	March 20, 2023, 14:25 (UTC-07:00)	-	4b9pbe
○	March 20, 2023, 14:27 (UTC-07:00)	Active	sgssve



Stage details Info

Stage name: prod	Rate Info: 10000	Web ACL: -
API cache: Inactive	Burst Info: 5000	Client certificate: -

Invoke URL
https://vx3y9lm2ec.execute-api.us-east-1.amazonaws.com/prod

Active deployment
sgssve on March 20, 2023, 14:27 (UTC-07:00)

Custom Domain Names

Custom Domains

$$\text{https://}\underbrace{\textit{12345}}_{\text{API ID}}\text{.execute-api.}\underbrace{\textit{us-east-1}}_{\text{Region}}\text{.amazonaws.com/}\underbrace{\textit{prod}}_{\text{Stage}}\text{/}\underbrace{\textit{catalog}}_{\text{Resource}}$$

Custom Domains

~~https://*12345*.execute-api.*us-east-1*.amazonaws.com/*prod*/*catalog*~~

Base path mapping	API name	Stage
api-one	API1	prod
api-two/v2	API2	dev

https://*mydomain.com/api-one/products*

https://*mydomain.com/api-two/v2/catalog*

- Supports HTTP, REST, and WebSocket APIs
- SSL Certs managed through ACM
- Supports multiple APIs through multi level base path mapping

Observability

Metrics

Built-in

REST

API Calls Count, Latency, 4XXs, 5XXs, Integration Latency, Cache Hit Count, Cache Miss Count

HTTP

API Calls Count, Latency, 4XXs, 5XXs, Integration Latency, DataProcessed

WebSocket

Connect Count, Message Count, Integration Error, Client Error, Execution Error, Integration Latency

Custom

Create Custom Metrics via Metric Filter out of logs

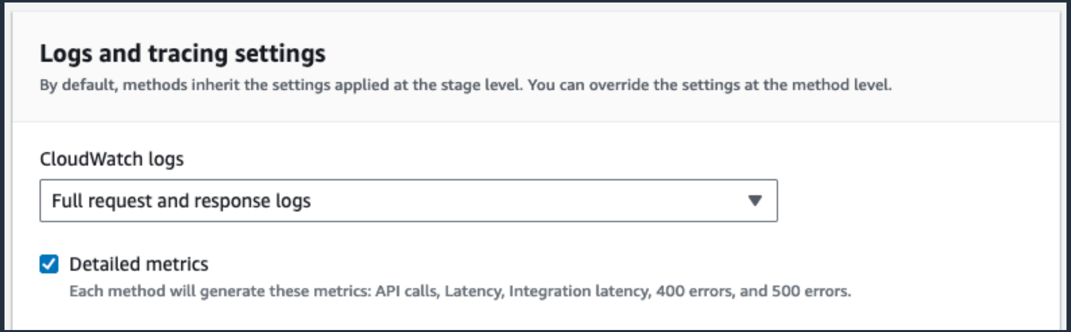
Logging

Execution Logs

REST

WS

- Two levels of logging, ERROR and INFO
- Optionally log method request/body content
- Set globally in stage, or override per method



Logs and tracing settings

By default, methods inherit the settings applied at the stage level. You can override the settings at the method level.

CloudWatch logs: Full request and response logs

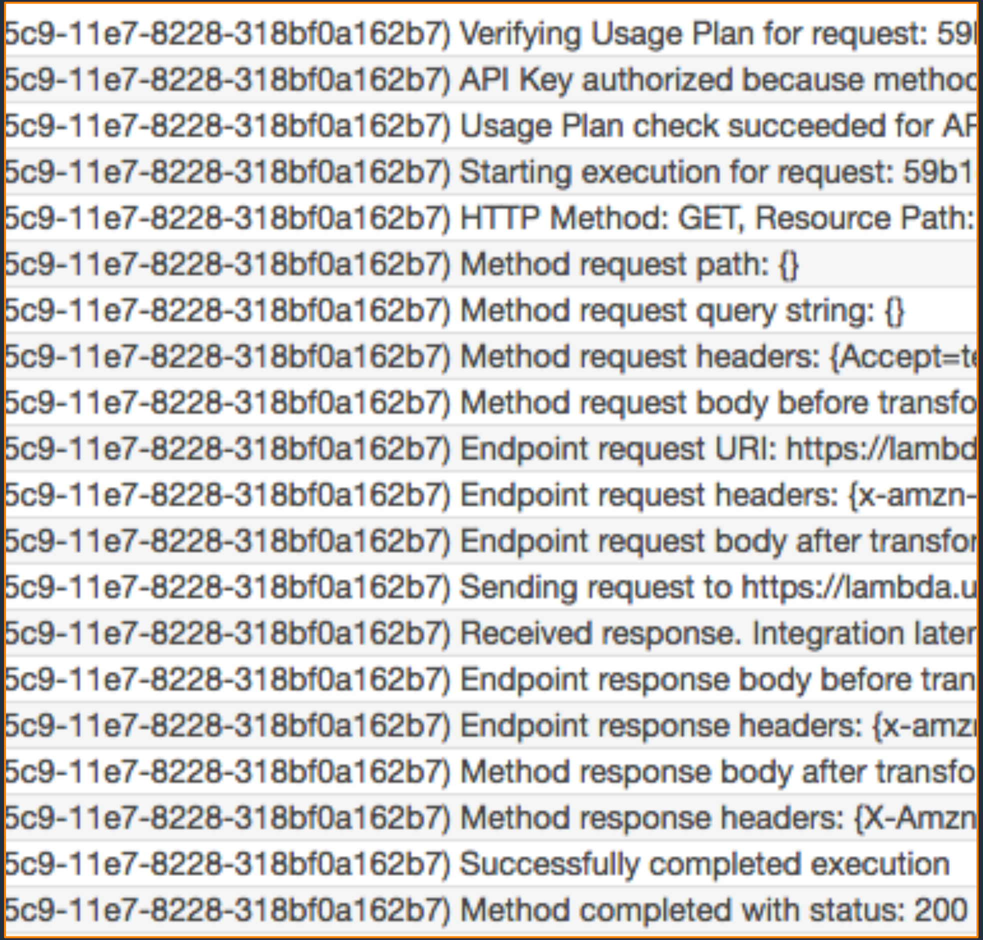
[✓] Detailed metrics

Each method will generate these metrics: API calls, Latency, Integration latency, 400 errors, and 500 errors.

Access Logs

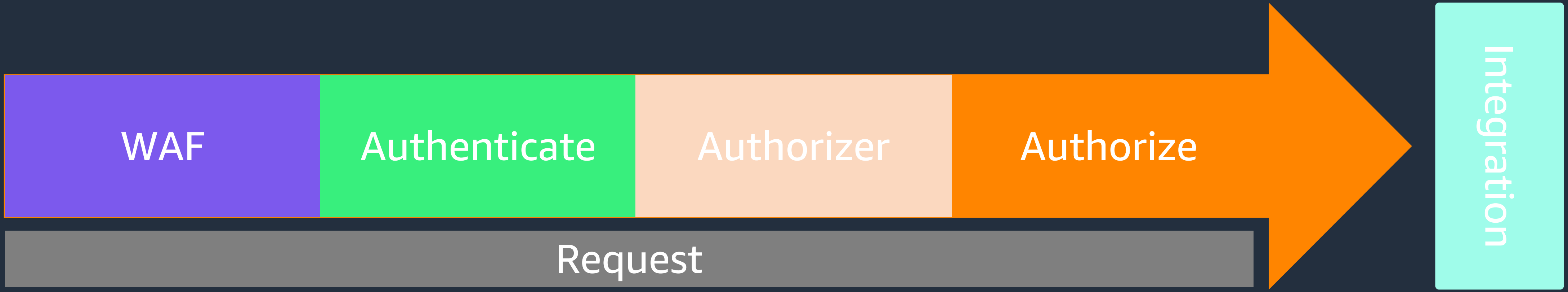
- Customizable format for machine parsable logs
- CloudWatch Logs OR Kinesis Firehose

REST



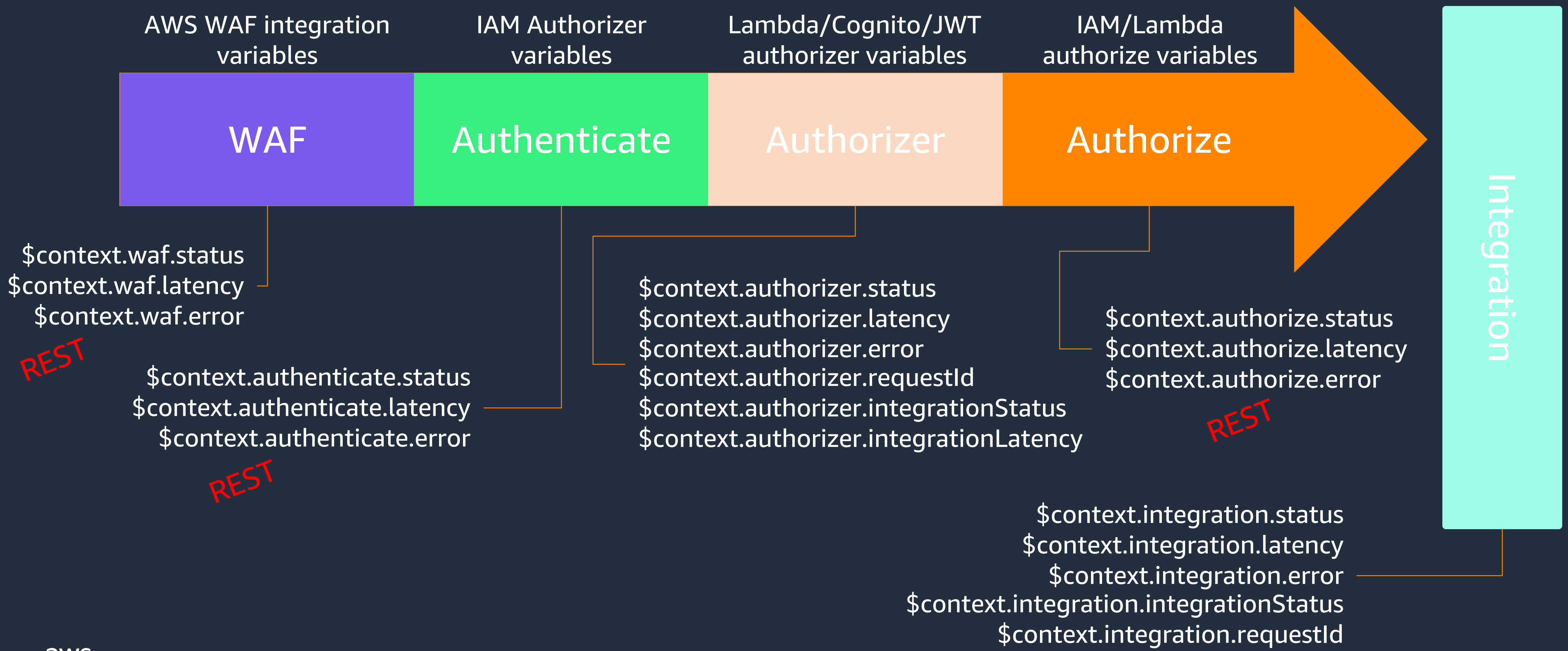
```
5c9-11e7-8228-318bf0a162b7) Verifying Usage Plan for request: 59l
5c9-11e7-8228-318bf0a162b7) API Key authorized because methoc
5c9-11e7-8228-318bf0a162b7) Usage Plan check succeeded for AF
5c9-11e7-8228-318bf0a162b7) Starting execution for request: 59b1
5c9-11e7-8228-318bf0a162b7) HTTP Method: GET, Resource Path:
5c9-11e7-8228-318bf0a162b7) Method request path: {}
5c9-11e7-8228-318bf0a162b7) Method request query string: {}
5c9-11e7-8228-318bf0a162b7) Method request headers: {Accept=te
5c9-11e7-8228-318bf0a162b7) Method request body before transfo
5c9-11e7-8228-318bf0a162b7) Endpoint request URI: https://lambd
5c9-11e7-8228-318bf0a162b7) Endpoint request headers: {x-amzn-
5c9-11e7-8228-318bf0a162b7) Endpoint request body after transfor
5c9-11e7-8228-318bf0a162b7) Sending request to https://lambda.u
5c9-11e7-8228-318bf0a162b7) Received response. Integration later
5c9-11e7-8228-318bf0a162b7) Endpoint response body before tran
5c9-11e7-8228-318bf0a162b7) Endpoint response headers: {x-amzr
5c9-11e7-8228-318bf0a162b7) Method response body after transfo
5c9-11e7-8228-318bf0a162b7) Method response headers: {X-Amzn
5c9-11e7-8228-318bf0a162b7) Successfully completed execution
5c9-11e7-8228-318bf0a162b7) Method completed with status: 200
```

Enhanced observability variables



status
$context.<phase>.latency
error

Enhanced observability variables



AWS WAF integration variables

$context.waf.status
$context.waf.latency
$context.waf.error

IAM Authorizer variables

$context.authenticate.status
$context.authenticate.latency
$context.authenticate.error

Lambda/Cognito/JWT authorizer variables

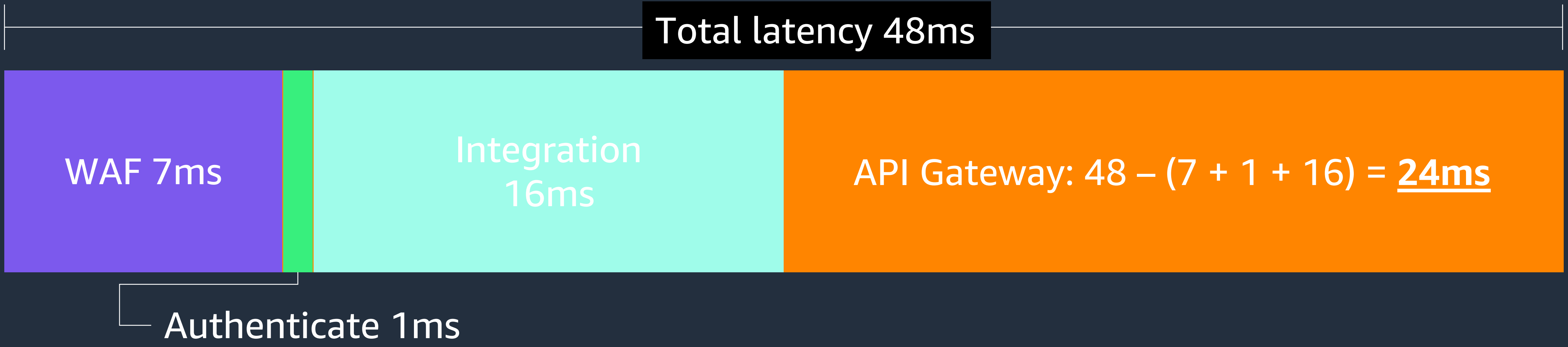
$context.authorizer.status
$context.authorizer.latency
$context.authorizer.error
$context.authorizer.requestId
$context.authorizer.integrationStatus
$context.authorizer.integrationLatency

IAM/Lambda authorize variables

$context.authorize.status
$context.authorize.latency
$context.authorize.error

$context.integration.status
$context.integration.latency
$context.integration.error
$context.integration.integrationStatus
$context.integration.requestId

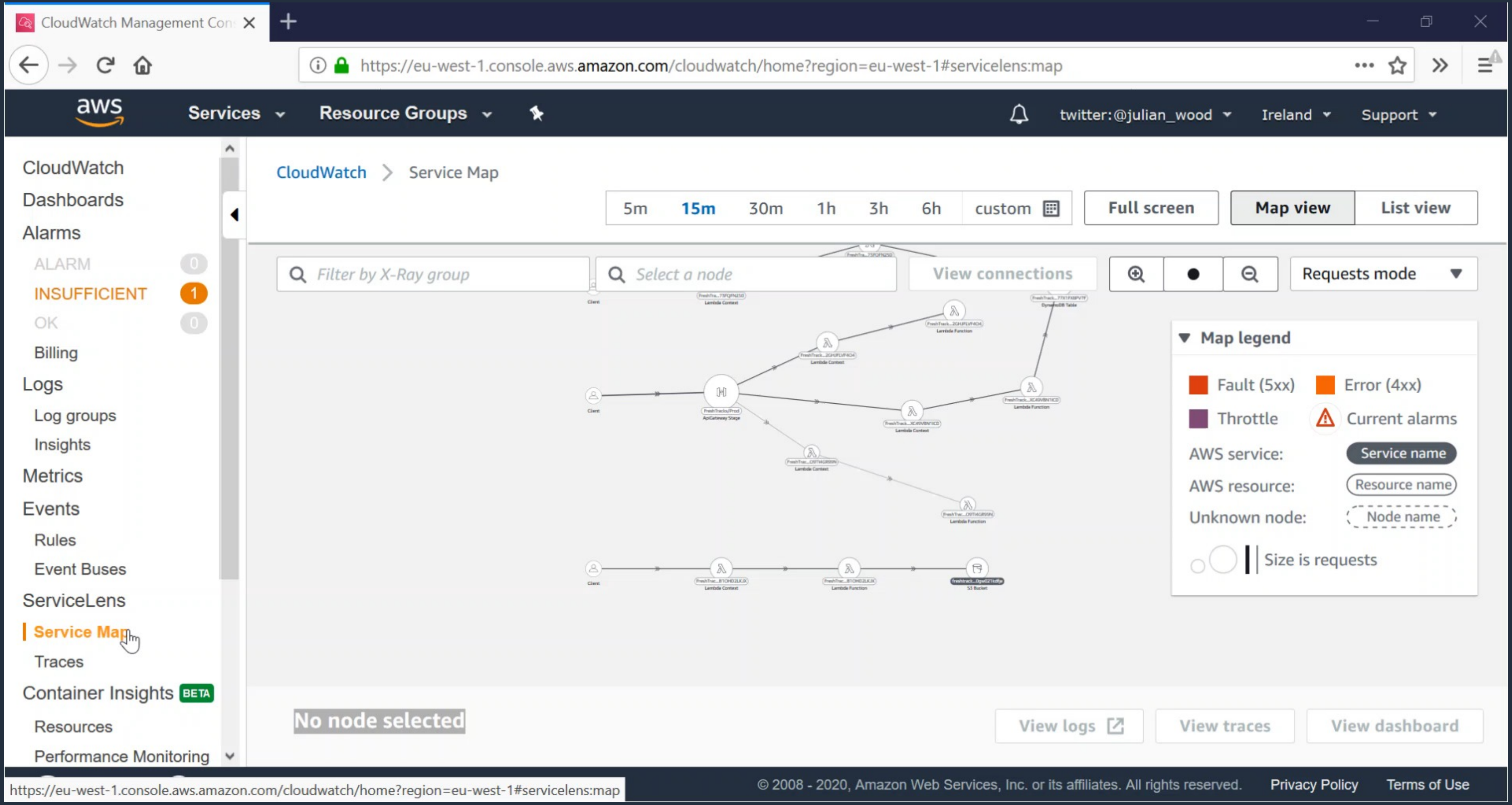
Troubleshooting latency



Example of an IAM Authorized Endpoint with an AWS WAF in front of it

REST

Tracing – X-Ray



CloudWatch Management Con

https://eu-west-1.console.aws.amazon.com/cloudwatch/home?region=eu-west-1#servicelens:map

aws Services Resource Groups twitter:@julian_wood Ireland Support

CloudWatch
Dashboards
Alarms
- ALARM 0
- INSUFFICIENT 1
- OK 0
- Billing

Logs
- Log groups
- Insights

Metrics
Events
- Rules
- Event Buses

ServiceLens
- Service Map
- Traces

Container Insights BETA
- Resources
- Performance Monitoring

CloudWatch > Service Map

5m 15m 30m 1h 3h 6h custom | Full screen | Map view | List view

Filter by X-Ray group | Select a node | View connections | Requests mode

Map legend
- 🟥 Fault (5xx)
- 🟧 Error (4xx)
- 🟪 Throttle
- ⚠ Current alarms
- AWS service: Service name
- AWS resource: Resource name
- Unknown node: Node name
- Size is requests

No node selected

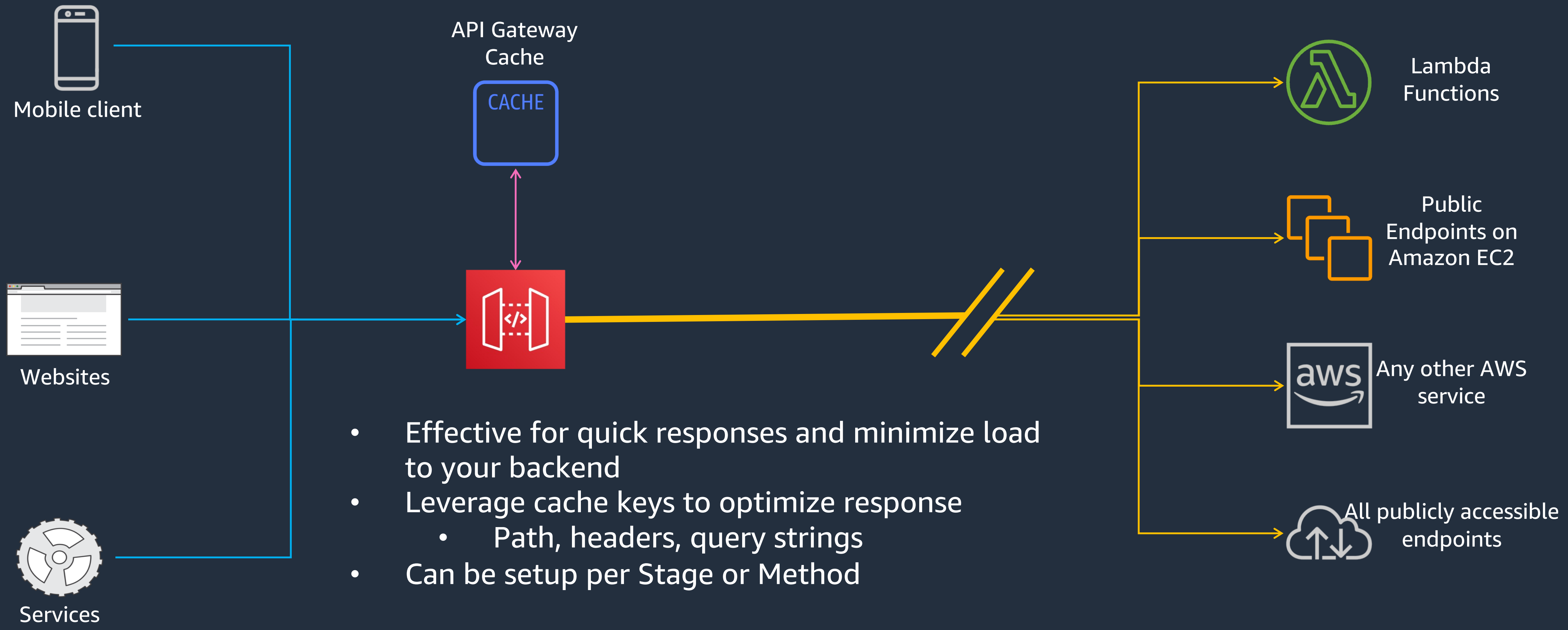
View logs | View traces | View dashboard

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

https://eu-west-1.console.aws.amazon.com/cloudwatch/home?region=eu-west-1#servicelens:map

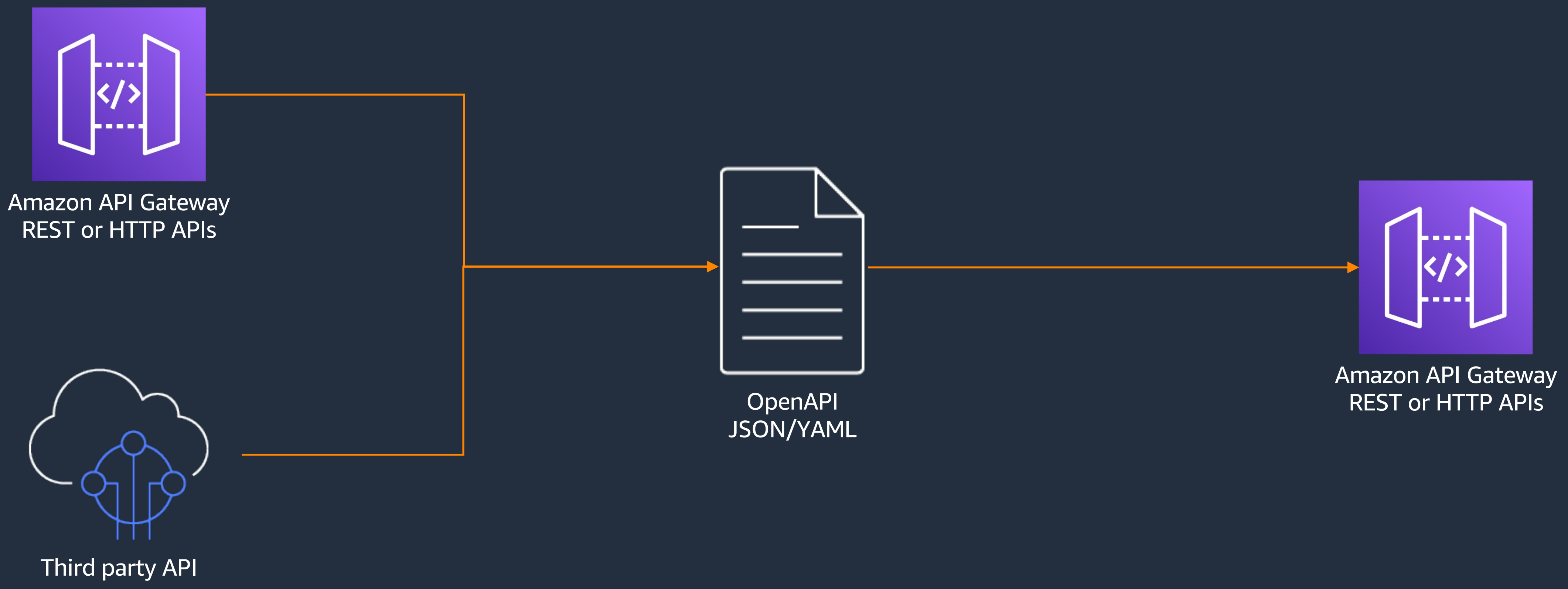
Other Features

Caching



- Effective for quick responses and minimize load to your backend
- Leverage cache keys to optimize response
  - Path, headers, query strings
- Can be setup per Stage or Method

OpenAPI for migrating APIs



Pricing

Pricing (eu-west-1)

- HTTP API requests: $1.11 / Million *(1M)**

- REST API requests : $3.5 / Million *(1M)* *
- REST API cache: $0.2/hour for 0.5GB $\rightarrow$ $3.80/hour for 237 GB *

- WebSocket requests: $1.14 / Million messages * *(1M)*
- WebSocket connection minutes: $0.285 / Million minutes *(750,000)*

* Tiered pricing

Free tier per month for 1 year

Important quotas

Important quotas

- Throughput: 10,000 Requests/second
- Max integration timeout: *30 seconds*
- Payload size: *10 MB*

- WebSocket connections: 500 connections/sec
- WebSocket connection duration: *2 hours*
- WebSocket message size: *128 KB*

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

Best practices

- Use HTTP APIs if existing features are sufficient
- Create an API per team/microservice and unify them with Custom Domain Name
- The console is for experimenting, use Infrastructure as Code:
    - SAM
    - CloudFormation
    - Terraform
    - Etc.

Useful resources

Security Overview of AWS API Gateway:

https://d1.awsstatic.com/whitepapers/api-gateway-security.pdf

Serverless Lens:

https://docs.aws.amazon.com/wellarchitected/latest/serverless-applications-lens/welcome.html

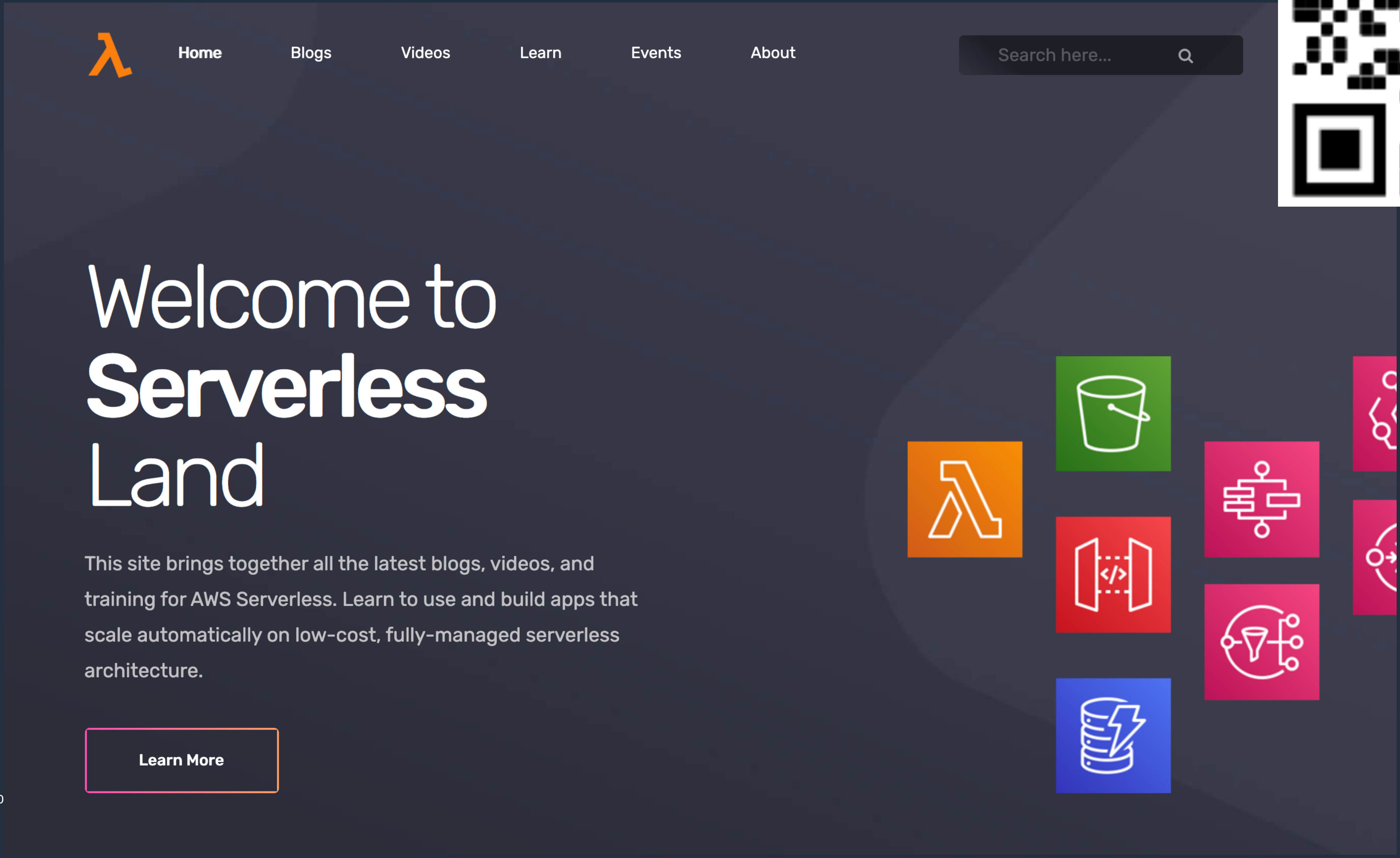
Choosing between HTTP and REST APIs

https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api-vs-rest.html

Best Practices for Designing Amazon API Gateway Private APIs and Private Integration

https://docs.aws.amazon.com/whitepapers/latest/best-practices-api-gateway-private-apis-integration/best-practices-api-gateway-private-apis-integration.html

Serverlessland.com



Welcome to **Serverless** Land

This site brings together all the latest blogs, videos, and training for AWS Serverless. Learn to use and build apps that scale automatically on low-cost, fully-managed serverless architecture.