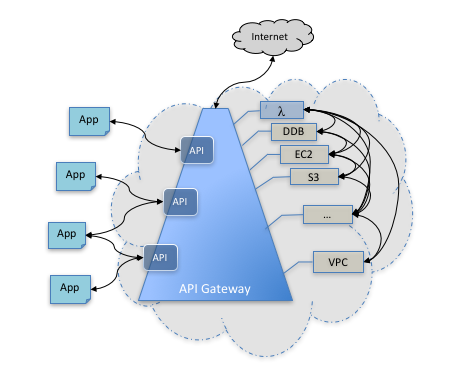
Amazon API Gateway is an AWS service that enables developers to create, publish, maintain, monitor, and secure APIs at any scale. You can create APIs that access AWS or other web services, as well as data stored in the [AWS Cloud](https://aws.amazon.com/what-is-cloud-computing/).

API Gateway can be considered a [backplane](https://en.wikipedia.org/wiki/Backplane) in the cloud to connect [AWS services](https://aws.amazon.com/) and other public or private websites. It provides consistent RESTful application programming interfaces (APIs) for mobile and web applications to access AWS services.



Permissions to invoke a method are granted using IAM roles and policies or API Gateway Lambda authorizers (formerly known as custom authorizers).

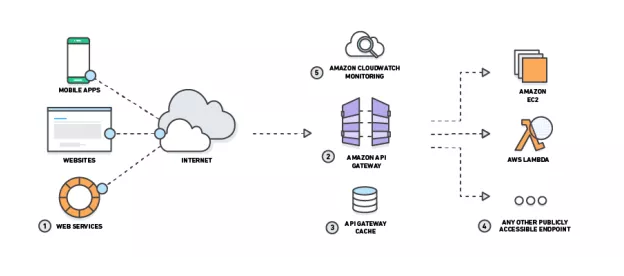
**Private API**

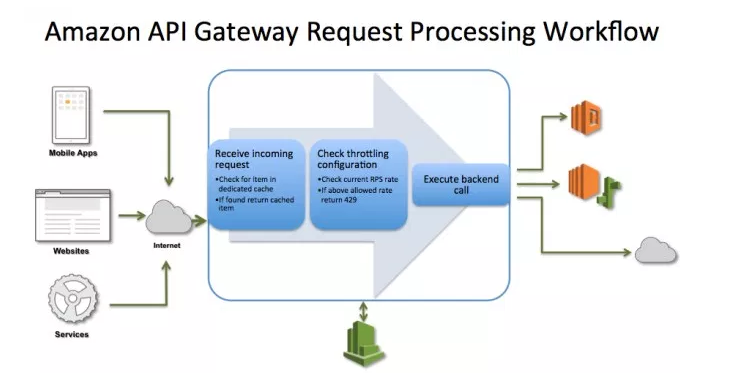
An API that is exposed through interface VPC endpoints and isolated from the public internet

**API deployment**

A point-in-time snapshot of your API Gateway API resources and methods. To be available for clients to use, the deployment must be associated with one or more API stages.

* API Gateway endpoints are always public to the Internet and does not run within an VPC. Proxy requests to backend operations also need to be publicly accessible on the Internet.





API Gateway can communicate to multiple backends

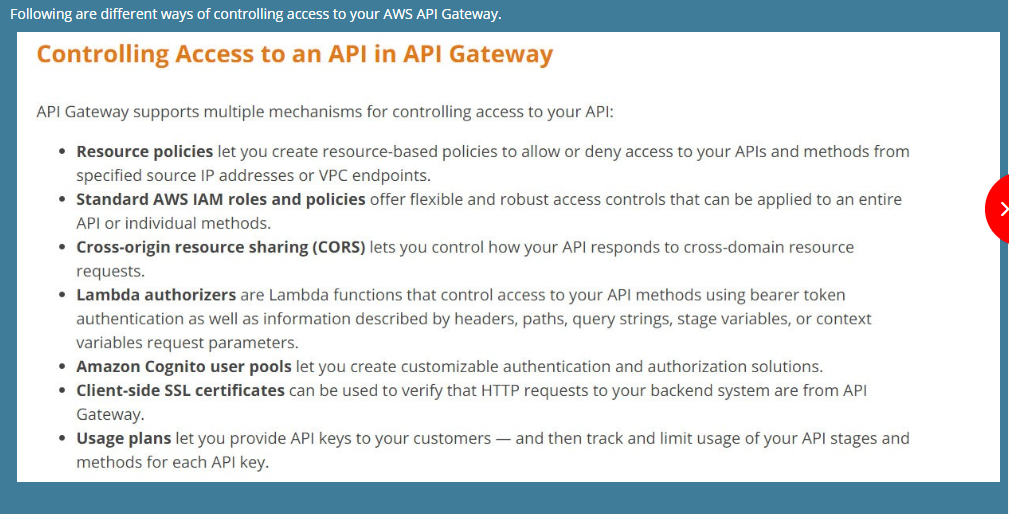
* Lambda functions
* AWS Step functions state machines
* HTTP endpoints exposed through Elastic Beanstalk, ELB or EC2 servers
* **Non AWS hosted HTTP based operations accessible via public Internet**

**Benefits of API Gateway**

API Gateway helps you deliver robust, secure, and scalable mobile and web application backends.

API Gateway allows you to securely connect mobile and web applications to business logic hosted on AWS Lambda, APIs hosted on Amazon EC2, or other publicly addressable web services hosted inside or outside of AWS. With API Gateway, you can create and operate APIs for backend services. For example, you don't need to develop and maintain infrastructure to handle authorization and access control, traffic management, monitoring and analytics, version management, and software development kit (SDK) generation.

API Gateway is designed for web and mobile developers who want to provide secure, reliable access to backend APIs for access from mobile apps, web apps, and server apps that are built internally or by third-party ecosystem partners. The business logic behind the APIs can be provided by a publicly accessible endpoint that API Gateway proxies call, or it can be entirely run as a Lambda function.



# Enable CORS for an API Gateway Resource

When your API's resources receive requests from a domain other than the API's own domain, you must enable cross-origin resource sharing (CORS) for selected methods on the resource. This amounts to having your API respond to the OPTIONSpreflight request with at least the following CORS-required response headers:

Prevents access from certain domain only..

Yes, all of the APIs created with Amazon API Gateway expose HTTPS endpoints only. Amazon API Gateway does not support unencrypted (HTTP) endpoints.

Amazon API Gateway can execute AWS Lambda functions in your account, start AWS Step Functions state machines, or call HTTP endpoints hosted on AWS Elastic Beanstalk, Amazon EC2, and also non-AWS hosted HTTP based operations that are accessible via the public Internet.&nbsp;API Gateway also allows you to specify a mapping template to generate static content to be returned, helping you mock your APIs before the backend is ready. You can also integrate API Gateway with other AWS services directly – for example, you could expose an API method in API Gateway that sends data directly to Amazon Kinesis.

Usage plans  - API restricting

help you declare plans for third-party developers that restrict access only to certain APIs, define throttling and request quota limits, and associate them with API keys. You can also extract utilization data on an per-API key basis to analyze API usage and generate billing documents.

**What is a stage?**

In Amazon API Gateway, stages are similar to tags. They define the path through which the deployment is accessible. For example, you can define a development stage and deploy your cars API to it. The resource will be accessible at <https://www.myapi.com/dev/cars>. You can also set up custom domain names to point directly to a stage, so that you don’t have to use the additional path parameter. For example, if you pointed myapi.com directly to the development stage, you could access your cars resource at <https://www.myapi.com/cars>. Stages can be configured using variables that can be accessed from your API configuration or mapping templates.

**Q: What is a Resource Policy?**

A Resource Policy is a JSON policy document that you attach to an API to control whether a specified principal (typically an IAM user or role) can invoke the API. You can use a Resource Policy to enable users from a different AWS account to securely access your API or to allow the API to be invoked only from specified source IP address ranges or CIDR blocks.

**How does AWS Signature Version 4 work?**

You can use AWS credentials - **access and secret keys - to sign requests to your service** and authorize access like other AWS services.

**What is a Lambda authorizer?**

Lambda authorizers are AWS Lambda functions. With custom request authorizers, you will be able to authorize access to APIs using a bearer token auth strategy such as OAuth. When an API is called, API Gateway checks if a Lambda authorizer is configured, API Gateway then calls the Lambda function with the incoming authorization token. You can ***use Lambda to implement various authorization strategies (e.g. JWT verification, OAuth provider callout) that return IAM policies which are used to authorize the request.*** If the policy returned by the authorizer is valid, API Gateway will cache the policy associated with the incoming token for up to 1 hour.

**Q: If an API response is served by cached data, is it still considered an API call for billing purposes?**

Yes. API calls are counted equally for billing purposes whether the response is handled by your backend operations or the Amazon API Gateway caching operation.

Thanks & Regards,

Shantaram Vernekar

**From:** Shantaram Vernekar   
**Sent:** Saturday, July 28, 2018 5:03 PM  
**To:** Shantaram Vernekar <[shantaram\_vernekar@persistent.com](mailto:shantaram_vernekar@persistent.com)>; 'shantaram.vernekar@citi.com' <[shantaram.vernekar@citi.com](mailto:shantaram.vernekar@citi.com)>  
**Subject:** AWS - API Gateway

API Gateway

<http://jayendrapatil.com/aws-api-gateway/>

* Caching
* Throttling
* Security
  + Authorization using IAM and Cognito
  + Prevent denial of service attacks..
* Monitoring using cloudwatch
* Lifecycle management using version
* Metering ..statistics…

Thanks & Regards,

Shantaram Vernekar