KONG OPEN SOURCE

Kong is an API gateway and platform. That means it is a form of middleware between computing clients and your API-based applications. Kong easily and consistently extends the features of your APIs. Some of the popular features deployed through Kong include Authentication, Security, Traffic Control, Serverless, Analytics & Monitoring, Request /Response Transformations and logging.

WHY KONG:

Compared to other API gateways and platforms, Kong has many important advantages that are not found in the market today.
Choose Kong to ensure your API gateway platform is:

- Radically Extensible
- Blazingly Fast
- Open Source
- Platform Agnostic
- Cloud Native
- RESTful

KEY FEATURES:

- Runs on any Infrastructure – Kong runs anywhere. You can deploy Kong in the cloud or on-premise environments, including single or multi data center setups and for public, private or invite only API's.
- Provides Authentication, Load Balancing i.e., dynamically balances load across backend services.
- Scalable- Kong easily scales horizontally by simply adding more machines, meaning your platform can handle virtually any load while keeping latency low.
- Modular-Kong can be extended by adding new plugins which are easily configured through a Restful Admin API.
- Configuration and administration tasks split between REST API and CLI.

- Extensible via 36 available plugins. Of those 36, 30 are Open Source and 6 Commercial.
- Enterprise Plugins offered by Kong are Paid.
- Supports any Language or framework.
- Supports all microservice use cases, patterns and designs.

Advantages

Kong has several advantages compared to other API management platforms and tools. Some of them are:

- **Open-Source**: No black box. For enterprise or free usage, Kong is entirely open-source, always.
- **Based on Nginx**: Kong is embedded in Nginx and benefits from its amazing proxying performances.
- **Customizable**: Write plugins to cover all your architecture use-cases.
- **Data Ownership**: Kong and its underlying datastore run on your servers.
- **Easy to scale**: All Kong nodes are stateless. Spawning new nodes in your cluster is very easy.
- **Integrations**: Many plugins integrate with popular third-party services in the microservices world.

Use Cases:

Because Kong is open-source and highly customizable, you'll have full control over your architecture. It's perfectly suited for managing *internal* microservice traffic as well as *partners* or *public* entities.

Internal

Internal API traffic that connects microservices built inside your organization can originate from teams in different geographic regions but belong to the same entity. Internal microservices and APIs can be deployed either on bare metal or a cloud provider.

Partners

Partner's usage is when your software has to communicate with a mission critical third party microservice to offer its key features (and vice-versa). For example, a credit card company relies on at least one bank to offer its service. In the past most of those connections would happen via an ESB, nowadays all you need is an API Gateway.

Public

Public is when you offer your API with a self-serve onboarding process. Any developer can get a key and access/consume your services. An example of this model is Facebook’s Graph API. Common features are authentication, rate limiting, and billing tiers if it’s a paid API.

HOW DOES KONG WORK:

A typical Kong setup is made of two main components:

- Kong’s Server, based on the widely adopted **NGINX** HTTP server, which is a reverse proxy processing your clients’ requests to your upstream services.
- Kong’s datastore, in which the configuration is stored to allow you to horizontally scale Kong nodes. Apache Cassandra and PostgreSQL can be used to fulfill this role.

Kong needs to have both these components set up and operational.

KONG SERVER

The Kong Server, built on top of **NGINX**, is the server that will actually process the API requests and execute the configured plugins to provide additional functionalities to the underlying APIs before proxying the request upstream.

Kong listens on several ports that must allow external traffic and are by default:

- `8000` for proxying. This is where Kong listens for HTTP traffic.
- `8443` for proxying HTTPS traffic.

Additionally, those ports are used internally and should be firewalled in production usage:

- `8001` provides Kong's **Admin API** that you can use to operate Kong.
- `8444` provides Kong's **Admin API** over HTTPS.

You can use the **Admin API** to configure Kong, create new users, enable or disable plugins, and a handful of other operations. Since you will be using this RESTful API to operate Kong, it is also extremely easy to integrate Kong with existing systems.

KONG DATASTORE:

Kong uses an external datastore to store its configuration such as registered APIs, Consumers and Plugins. Plugins themselves can store every bit of information they need to be persisted, for example rate-limiting data or Consumer credentials.

Kong maintains a cache of this data so that there is no need for a database roundtrip while proxying requests, which would critically impact performance. This cache is invalidated by the inter-node communication when calls to the Admin API are made. As such, it is discouraged to manipulate Kong's datastore directly, since your nodes cache won't be properly invalidated.

This architecture allows Kong to scale horizontally by simply adding new nodes that will connect to the same datastore and maintain their own cache.

Which datastores are supported?

Apache Cassandra:

Apache Cassandra is a popular, solid and reliable datastore used at major companies like Netflix and Facebook. It excels at securely storing data in both single-datacenter or multi-datacenter setups by providing good performance and a fail-tolerant architecture. Kong can use Cassandra as its primary datastore if you are aiming at a distributed, high-availability Kong setup. It is reasonably easy to configure a multi-region infrastructure with a Cassandra datastore.

PostgreSQL 9.5+

PostgreSQL is an established SQL database for use with Kong.

It is a good candidate for single instance or centralized setups due to its relative simplicity and strong performance. Many cloud providers can host and scale PostgreSQL instances, most notably Amazon RDS.

Whether using Cassandra or PostgreSQL, Kong maintain its own cache. As a result, Kong gateway and plugin performance is sub-millisecond for most use-cases.

Routing & Oauth2 using Kong:

To Add an OAuth 2.0 authentication layer with the Authorization Code Grant, Client Credentials, Implicit Grant follow the below steps.

Step1: Install kong and set data store (postgresql/cassandra). Kong can be installed into various operating environments like docker, aws linux, kubernetes and macOS etc. Follow the link

Step2: Follow the guide to quick start with Kong

Step3: Configure a service. Follow the guide to do so. Below are the sub steps involved

- Add your services using Admin API
- Add route for the service
- Forward requests through Kong

Step4: Follow the guide to authenticate your API's. Below are the sub steps involved.

- Enable Oauth2 plugin on the service.

- Create a consumer
- Create an Application
- Migrate Access Tokens
- Upstream Headers

And now you can authenticate your API’s using Kong