Kong

- **Introduction**
  Kong is an open source API gateway. 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. It helps in managing API traffic with different plugins such as rate limiting, authentication etc.

- **Terminologies**
  client: Refers to the downstream client making requests to Kong's proxy port.
  upstream service: Refers to your own API/service sitting behind Kong, to which client requests are forwarded.
  Service: Service entities, as the name implies, are abstractions of each of your own upstream services. Examples of Services would be a data transformation microservice, a billing API, etc.
  Route: This refers to the Kong Routes entity. Routes are entry points into Kong, and defining rules for a request to be matched, and routed to a given Service.
  Plugin: This refers to Kong "plugins", which are pieces of business logic that run in the proxying lifecycle. Plugins can be configured through the Admin API - either globally (all incoming traffic) or on specific Routes and Services
- **Architecture of Kong**

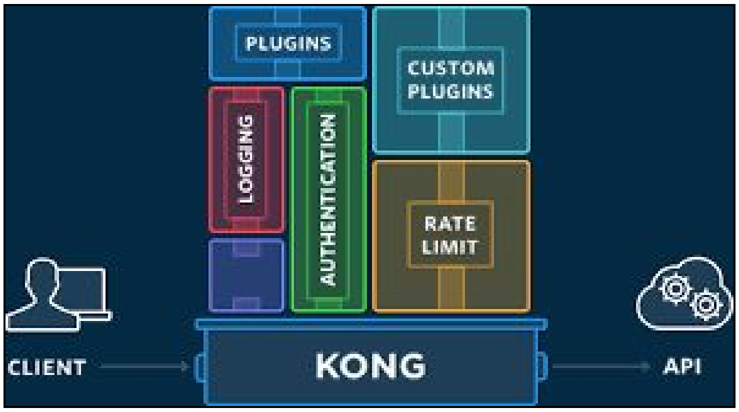

Fig. 13.D.1 : Kong architecture

- **Advantages of using Kong**
  1. Radically Extensible: Ready-to-deploy plugins add powerful functionality to your APIs and applications
  2. Open Source: Open and developer-friendly, tens of thousands have embraced the Kong microservice API gateway.
  3. RESTful Interface: Kong operates through a simple and easy-to-use RESTful API. In other words the transformation we obtain after we use Kong is:

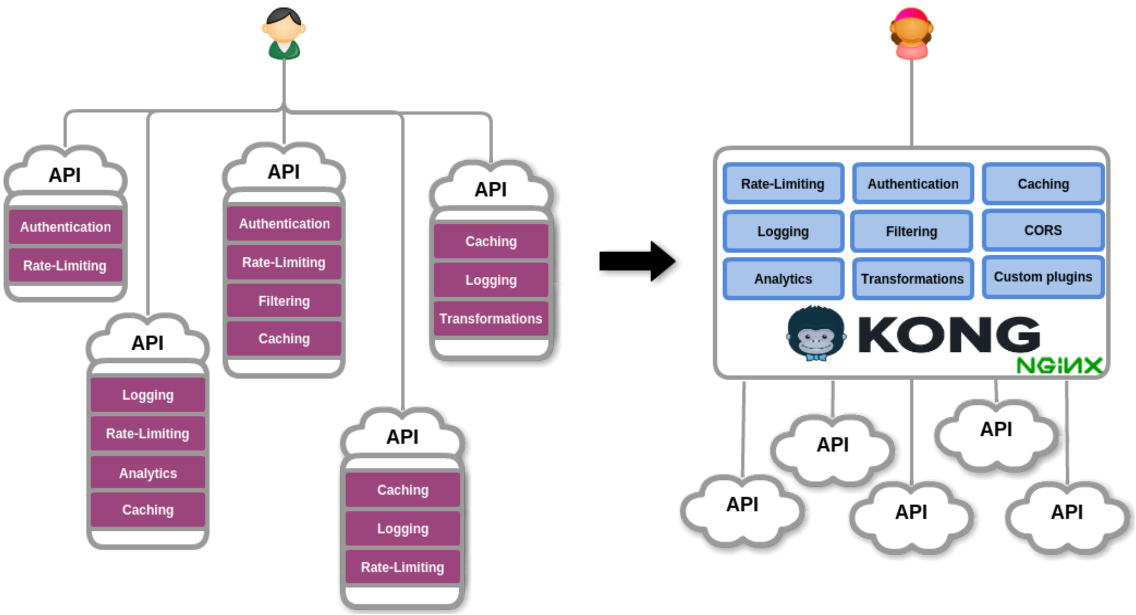

Fig. 13.D.2 : Kong transforming APIs

- **Functionality of Kong**

  A typical kong setup is made of two components:

1) Kong's server (based on NGINX HTTP server)

The Kong Server 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. The default ports through which kong listens are:
8000: for proxying
8443: for proxying HTTPS content
8001: for admin API
8444: for admin API over HTTPS.

Kongs Admin API can be used to create new users, configure Kongs, enable/disable plugins etc. Since this admin API uses REST interface, it becomes easy to integrate kong with existing systems.

2) Kong's dataserver (PostgreSQL or Apache Cassandra)

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 when calls to the Admin API are made.

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

- **Kong Plugins**
  Plugins are one of the most important features of Kong. Many Kong API gateway features are provided by plugins. Authentication, rate-limiting, transformation, logging etc, are all implemented independently as plugins. Plugins can be installed and configured via the Admin API running alongside Kong.

- **Examples to Add API**
  Kong listens at port 8001 for admin API requests, so by sending the following request it is possible to add an API to local instance of kong.
  POST : http://10.129.103.85/apis/
  Request Body (example):
    name: Swift
    upstream_url: http://10.196.0.56:8000/files/info/
    request_path: /swiftapis
    strip_request_path: true