**KONG GATEWAY**

# Định nghĩa

* Là 1 opensource API GATEWAY giúp quản lý, điều hướng và bảo mật các yêu cầu API giữa các microservice hoặc giữa user với hệ thống.
* Nó hoạt động như 1 lớp trung gian giữa backend và frontend hoặc bên thứ 3, cung cấp các tính năng nhằm tối ưu hóa việc quản lý và bảo mật API
* Phiên bản trước 2.x Kong không hoạt động với Postgres 12

# Tại sao lại sử dụng Kong

* Điểm mạnh
  + Hiệu năng cao: Kong được build bằng [Ngnix](#_Ngnix). Nhờ kiến trúc [event-driven](#_Event-driven) của Ngnix mà Kong có thể xử lý nhiều request cùng 1 lúc mà không ảnh hưởng đến hiệu suất và tốc độ
  + Tính mở rộng: có thể mở rộng để xử lí số lượng yêu cầu API lớn. Tính năng [load balancing](#_load_balancing) và [clustering](#_clustering) giúp duy trì tính sẵn sàng cao và khả năng mở rộng hệ thống
  + Khả năng mở rộng qua plugin: Có thể hỗ trợ nhiều tính năng xác thực (Oauth2, JWT, API Key, …), caching, logging, …
  + Dễ triển khai: Có thể deploy nó trên đa dạng các nền tảng như Docker, Kubernetes hoặc trên các server vật lý/ảo. Có thể tích hợp với các dịch vụ quản lý API khác (Amazon API Gateway, …) và các công cụ devops (Prometheus, Grafana)
  + Quản lý API tập trung: Kong quản lý API ở 1 chỗ duy nhất với khả năng theo dõi, kiểm soát luồng truy cập, bảo vệ API, điều hướng API tới đúng service
  + Hỗ trợ phân tán và microservices: Kong có khả năng service discovery, Kong tự động phát hiện và tích hợp các services backend trong hệ thống giúp quản lý dễ dàng hơn
* Điểm yếu:
  + Cấu hình và bảo trì phực tạp: Cấu hình các plugin, kiểm soát xác thực hay các tính bảo mật đòi hỏi người quản trị có kiến thức sâu về hệ thống
  + Tài nguyên yêu cầu cao: Kong có thể tiêu tốn 1 lượng lớn tài nguyên đáng kể (CPU, RAM) 🡺 Chi phí tăng cao
  + Hạn chế trong việc xử lý phức tạp: [1 số TH](#_TH_Kong_gặp) API Gateway xử lý phức tạp hơn có thể không phù hợp với Kong
  + Quản lý nhiều môi trường: Khi hệ thống có nhiều môi trường (dev, staging, production), việc quản lý và đồng bộ các cấu hình của Kong sẽ gặp khó khăn nếu ko có quy trình chuẩn và không có công cụ quản lý
  + Cộng đồng hỗ trợ và tài liệu: Kong có cộng đồng mạnh mẽ nhưng tài liệu không quá chi tiết và dễ tiếp cận
  + Giá cho bản nâng cấp: Phiên bản Kong trả phí có nhiều tính năng mạnh mẽ như GUI quản lý, phân tích chi tiết và tích hợp với các dịch vụ doanh nghiệp.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tiêu chí** | **Kong Gateway** | **Spring Cloud Gateway** | **Traefik** | **Apigee** |
| Mô hình triển khai | Triển khai tại chỗ hoặc đám mây |  | Triển khai tại chỗ hoặc đám mây | Đám mây hoặc tại chỗ |
| Hiệu suất | Hiệu suất cao, tối ưu cho tải lớn |  | Hiệu suất cao, hỗ trợ tải lớn | Hiệu suất tốt, có thể mở rộng linh hoạt |
| Khả năng mở rộng | Mở rộng dễ dàng với kiến trúc microservices |  | Mở rộng tự động, hỗ trợ Docker và Kubernetes | Mở rộng tốt với tính năng trên đám mây Google |
| Tính năng chính | Proxy API, quản lý traffic, plugin tùy chỉnh phong phú |  | Proxy API, hỗ trợ Docker, Kubernetes | Proxy API, phân tích, bảo mật, quản lý toàn diện |
| Khả năng bảo mật | Hỗ trợ OAuth2, ACL, rate limiting, JWT |  | Hỗ trợ TLS, OAuth2, rate limiting, JWT | Bảo mật API với OAuth2, SAML, rate limiting, JWT |
| Tích hợp DevOps | Hỗ trợ CI/CD, tích hợp dễ dàng với Kubernetes |  | Tích hợp CI/CD, Kubernetes | Hỗ trợ CI/CD, tích hợp tốt với Google Cloud |
| Giám sát và quản lý | Cung cấp dashboard, tích hợp Prometheus, Grafana |  | Hỗ trợ Prometheus, Grafana | Tích hợp với Google Cloud Monitoring, phân tích chi tiết |
| Mã nguồn mở | Mã nguồn mở |  | Mã nguồn mở | Không phải mã nguồn mở, thuộc Google |

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **Kong Gateway** | **Traefik** |
| Ngôn ngữ phát triển | Lua, OpenResty (Nginx + LuaJIT) | Go |
| Triển khai | Hỗ trợ Docker, Kubernetes, bare-metal. Yêu cầu cơ sở dữ liệu (hoặc DB-less mode) | Hỗ trợ Docker, Kubernetes. Không yêu cầu cơ sở dữ liệu |
| Service Discovery | Cần cấu hình hoặc sử dụng plugin như Consul, Kubernetes | Tự động phát hiện dịch vụ với Docker, Kubernetes, Consul |
| Hỗ trợ giao thức | HTTP/HTTPS, gRPC, TCP, WebSocket | HTTP/HTTPS, gRPC, TCP, WebSocket |
| Plugin & Middleware | Nhiều plugin mạnh mẽ cho bảo mật, quản lý API (OAuth2, JWT, rate limiting, v.v.) | Middleware cơ bản (Basic Auth, JWT), ít phong phú hơn Kong |
| Quản lý & Mở rộng | Có bản Kong Enterprise với nhiều tính năng quản lý, hỗ trợ giám sát, bảo mật API | Có Traefik Enterprise, cung cấp dashboard trực quan, tự động phát hiện dịch vụ |
| Hiệu suất | Hiệu suất cao nhờ Nginx, LuaJIT. Phù hợp với hệ thống lớn | Hiệu suất cao nhờ Go, tối ưu cho môi trường container và microservices |
| Bảo mật | Hỗ trợ nhiều plugin bảo mật: OAuth2, JWT, ACL | Middleware bảo mật cơ bản (Basic Auth, JWT) |
| Cộng đồng & Hỗ trợ | Cộng đồng lớn, hỗ trợ doanh nghiệp với bản Enterprise | Cộng đồng năng động, đặc biệt trong môi trường container. Bản Enterprise hỗ trợ thêm |
| Sử dụng phổ biến | Phù hợp cho các tổ chức lớn, yêu cầu quản lý API và bảo mật phức tạp | Phổ biến trong DevOps, Kubernetes, và container. Dễ triển khai, ít cấu hình phức tạp |
| Giá cả | Bản Community miễn phí, Enterprise tính phí theo yêu cầu doanh nghiệp | Bản OSS miễn phí, bản Enterprise có phí với tính năng mở rộng |

# Cách sử dụng

* docker network create kong-net
  + Tạo ra 1 docker để các containers có thể giao tiếp với nhau
* docker run -d --name kong-db --network=kong-net -p 5432:5432 -e "POSTGRES\_USER=kong" -e "POSTGRES\_DB=kong" -e "POSTGRES\_PASSWORD=password" postgres:13
  + Tạo và chạy database cho Kong, trong TH này sử dụng postgres phiên bản 13
* docker run --rm --network=kong-net -e "KONG\_DATABASE=postgres" -e "KONG\_PG\_HOST=kong-db" -e "KONG\_PG\_PASSWORD=password" -e "KONG\_PASSWORD=test" kong/kong-gateway:3.8.0.0 kong migrations bootstrap
  + Add Postgres vào Kong database
* docker run -d --name kong-gateway --network=kong-net -e "KONG\_DATABASE=postgres" -e "KONG\_PG\_HOST=kong-database" -e "KONG\_PG\_USER=kong" -e "KONG\_PG\_PASSWORD=kongpass" -e "KONG\_PROXY\_ACCESS\_LOG=/dev/stdout" -e "KONG\_ADMIN\_ACCESS\_LOG=/dev/stdout" -e "KONG\_PROXY\_ERROR\_LOG=/dev/stderr" -e "KONG\_ADMIN\_ERROR\_LOG=/dev/stderr" -e "KONG\_ADMIN\_LISTEN=0.0.0.0:8001" -e "KONG\_ADMIN\_GUI\_URL=http://localhost:8002" -e KONG\_LICENSE\_DATA -p 8000:8000 -p 8443:8443 -p 8001:8001 -p 8444:8444 -p 8002:8002 -p 8445:8445 -p 8003:8003 -p 8004:8004 kong/kong-gateway:3.8.0.0
  + Chạy container Kong gateway để khởi chạy
* Sau khi chạy xong chúng ta sẽ có nhiều url khác nhau
  + Cổng 8000, 8443 là cổng chính để Kong gateway nhận requests từ client và chuyển tiếp đến cho các services khác nhau mà Kong quản lý
    - Localhost:8000: Giao tiếp HTTP
    - Loclhost:8443: Giao tiếp HTTPS
  + Cổng 8001, 8444: là Admin API URL sử dụng để quản lý cấu hình của Kong, cổng này cho phép nhìn, thêm, sửa, xóa cấu hình, plugin, … Chúng ta có thể gửi request đến cho cổng này thông qua POST, GET, PUT, DELETE. Api “/services”: là nơi quản lý các services, 1 service sẽ đại diện cho 1 backend API hoặc 1 service mà Kong quản lý
    - Localhost:8001: Admin api qua HTTP, dùng để quản lý các dịch vụ, plugin, users và các thiết lập kong khác
    - Localhost:8444: Admin api qua HTTPS, dùng để bảo mật khi quản lý Kong từ xa
  + Cỏng 8002, 8445: là cổng cho việc truyển tải dữ liệu gRPC
    - Localhost:8002/workspaces: Đây là URL liên quan đến [Kong Enterprise Edition](#_Kong_Enterprise_Edition:). Là 1 tính năng cho phép phân vùng các services theo từng nhóm riêng biệt, giúp dễ dàng quản lý và phân quyền cho các nhóm phát triển khác nhau.
* Lưu ý: Trong môi trường Docker, 127.0.0.1 thường chỉ container hiện tại, không phải host hay là container khác

🡺 Việc sử dụng Kong gateway kết nối đến servervice 127.0.0.1:xxxx sẽ gây lỗi do đó nên sử dụng local ipv4 trong wifi ví dụ như 192.168.1.2

# Giải thích

## Ngnix: Là 1 trong những web service có hiệu xuất vô cùng cao

## Event-driven: Là mô hình thiết kế phần mềm trong đó các thành phần của hệ thống tương tác với nhau thông qua việc phát sinh và xử lý event

|  |  |
| --- | --- |
| **Thành phần** | **Tác dụng** |
| Event Producers | - Khi user tương tác với giao diện dẫn đến sự thay đổi về database, Event Producers sẽ gửi event và hệ thống |
| Event Consumers | - Đây là phần nhận và xử lý event mỗi khi được phát sinh |
| Event Channels | - Event Channels sẽ giúp truyền event từ Producers đến Consumers. Nó sẽ na ná như message queue |
| Event Processors | - Đây có thể là 1 Consumers hoặc là 1 bên trung gian có nhiệm vụ xử lý event trước khi chuyển tiếp hoặc phản hồi cho các thành phần khác.  - Quy trình xử lý sẽ gồm: Lọc, Chuyển đổi hoặc Tổng hợp dữ liệu |

## Load balancing: Là kỹ thuật phân phối lưu lượng truy cập hoặc tải công việc giữa nhiều máy chủ hoặc tài nguyên để tối ưu hóa hiệu xuất, độ tin cậy và thời gian phản hồi của hệ thống

## Clustering: Là kỹ thuật trong khoa học máy tính để có thể nhóm đối tượng, dữ liệu lại thành các nhóm sao cho các đối tượng được nhóm có nhiều điểm tương đồng với nhau (1 số tác dụng tiêu biểu: phân tích hành vi người dùng, phân khúc khách hàng, …)

## TH Kong gặp khó khăn:

* + Xử lý lưu lượng phức tạp với nhiêu giao thức không phải HTTP/HTTPS: Do Kong chủ yếu hỗ trợ các giao thức HTTP/HTTPS/WebSocket nên là 1 số giao thức khác như MQTT, AMQP sẽ gặp khó khăn
  + Tích hợp sâu với hệ thống message queue: Mặc dù Kong có thể hỗ trợ tích hợp các plugin như RabitMQ, Kafka, ActiveMQ để xử lý request không đồng bộ nhưng có 1 số API GateWay khác có thể tích hợp với các plugin khác dễ dàng hơn Kong rất nhiều ví dụ như Zuul, Envoy

## Kong Enterprise Edition: là 1 phiên bản của Kong gateway, cung cấp nhiều tính năng cao cấp và công cụ hỗ trợ tốt hơn cho các doanh nghiệp khi so sánh với phiên bản mã nguồn mở