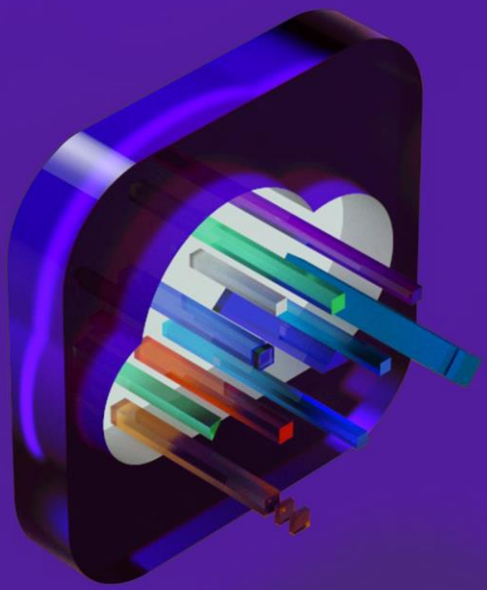
Kong

ebook

Kong 入门指南

面向多云和混合组织的下一代

API 平台



Kong 入门指南

本入门指南将向您介绍 Kong 的概念以及 API 网关的基本功能。

在本指南中，您将：

1.如何学习 Kong

2. Kong 总览

3. 准备管理 Kong Gateway

4. 通过 Kong Gateway 公开您的服务

5. 保护您的服务

6. 通过代理缓存提高性能

7. 使用身份验证保护服务

8. 设置智能负载平衡

1.如何学习 Kong

人生在勤，不索何获。

QQ 群：1065610885

加入我们每天分享多一点，收获多一点，进步多一点。

2. 总览

Kong

Kong Gateway 是一个开源的轻量级 API 网关，针对微服务进行了优化，提供了无与伦比的延迟性能和可伸缩性。如果只需要网关的基本功能，开源版本可以完全满足。

企业版和免费版

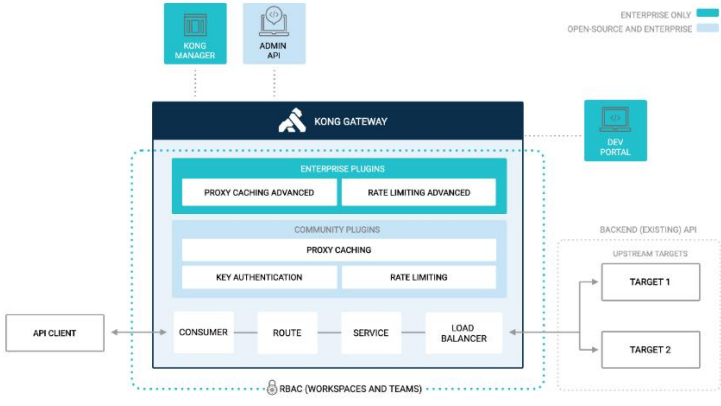
本指南还包括一些 Kong Enterprise 和 Kong Gateway 版的特定功能。在架构图中以绿色方块标识或在 Kong Manager 标签中被标识出。

Kong Enterprise 企业版功能通过插件的形式，扩展了 Kong Gateway，以实现安全性，协作，大规模性能以及高级协议的应用。

如果您目前还没有 Kong Enterprise，但想体验一下，请查看我们的免费试用版。

本指南中的概念和功能

这是本指南中介绍的内容，以及各个部分如何组合在一起：



功能	描述	开源/企业版
服务	服务对象是 Kong Gateway 用来引用其管理的上游 API 和微服务的 ID。	两者都有
路线	路由指定在请求到达 API 网关后如何（以及是否）将请求发送到其服务。单个服务可以具有多个路由。	两者都有

消费者	使用者代表您的 API 的最终用户。使用使用者对象，您可以控制谁可以访问您的 API。他们还允许您使用日志记录插件和 Kong Vitals 报告流量。	两者都有
Kong 管理器	Kong Manager 是基于视觉浏览器的工具，用于监视和管理 Kong Enterprise。	仅企业版
管理员 API	Kong Gateway 随附一个内部 RESTful API，用于管理目的。API 命令可以在群集中的任何节点上运行，并且配置将一致地应用于所有节点。	两者都有，但在 Kong Enterprise 中增加了功能
外挂程式	插件提供了用于修改和控制 Kong Gateway 功能的模块化系统。例如，为了保护您的 API，您可能需要一个访问密钥，可以使用 key-auth 插件进行设置。插件提供了广泛的功能，包括访问控制，缓存，速率限制，日志记录等等。	两者都有，但在 Kong Enterprise 中增加了功能
限速插件 限速高级插件	使用此插件，您可以限制客户端在给定时间内可以发出的 HTTP 请求数量。此插件的高级版本还提供了滑动窗口支持，并可以通过标题和服务进行限制。	两者都有 但在 Kong Enterprise 中增加了功能
代理缓存插件 代理缓存高级插件	该插件提供了反向代理缓存实现。它在给定的时间段内根据响应代码，内容类型和请求方法缓存响应实体。此插件的高级版本支持 Redis 和 Redis Sentinel 部署。	两者都有，但在 Kong Enterprise 中增加了功能
密钥验证插件	密钥验证-加密插件 该插件可让您向服务或路由添加密钥身份验证（也称为 API 密钥）。此插件的高级版本将 API 密钥以加密格式存储在 Kong Gateway 数据存储区中。	两者都有，但在 Kong Enterprise 中增加了功能
负载均衡	Kong Gateway 提供了两种负载平衡方法：基于 DNS 的直接方法或使用环形平衡器的方法。在本指南中，您将使用环形平衡器，该平衡器需要配置上游和目标实体。使用这种方法，后端服务的添加和删除由 Kong Gateway 处理，并且不需要 DNS 更新。	两者都有
用户授权（RBAC）	Kong Gateway（Enterprise）通过基于角色的访问控制（RBAC）处理用户授权。启用后，RBAC 允许您创建团队和管理员，	仅企业版

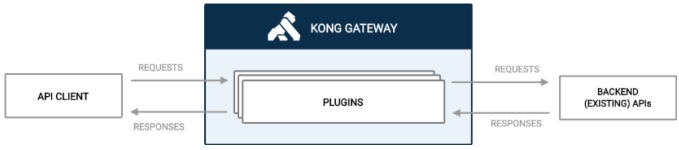
	并在工作空间内或跨工作空间为他们分配精细权限。	
开发者门户	开发人员门户网站为所有开发人员提供了一个单一的真实来源，以查找，访问和使用服务。	仅企业版

了解 Kong Gateway 中的流量

默认情况下，Kong Gateway 在其配置的代理端口 8000 和 8443 上侦听流量。它评估传入的客户端 API 请求，并将它们路由到适当的后端 API。在路由请求和提供响应时，可以根据需要通过插件应用策略。

例如，在路由请求之前，可能需要客户端进行身份验证。这带来了许多好处，包括：

- 由于 Kong Gateway 正在处理身份验证，因此该服务不需要自己的身份验证逻辑。
- 该服务仅接收有效请求，因此不会浪费周期来处理无效请求。
- 记录所有请求以集中查看流量。



在你开始之前

在开始使用本指南之前，请注意以下事项：

安装

- 本指南假定您已在所选平台上安装并运行了 Kong Gateway 或 Kong Gateway（Enterprise），或者已注册了 Enterprise 免费试用版。
- DocKer 安装 Koong 方法

- 在安装过程中，请注意 KONG_PASSWORD；您将在本指南的稍后部分中需要它来设置用户授权。

部署准则

- 您可以使用本指南在生产环境中入门，但是本指南并未提供生产环境所需的所有必要配置和安全设置。

- 本指南中的所有示例均用于引用 Kong Gateway 实例的 Admin API URL。确保将变量替换为 Kong Gateway 安装的实际 URL。要查找 URL，请检查文件中的 admin_listen 属性 /etc/kong/kong.conf。

使用本指南

- 作为 Kong Enterprise 或 Free Trial 用户，可以使用基于 REST 的 Admin API 或使用 Kong Manager GUI 以编程方式管理功能。作为 Kong Gateway 用户，由于 Manager 是企业功能，因此您需要遵循 Admin API 步骤。在本指南中，如果有可用的选项，则可以选择首选的方法-您不必同时遵循这两种方法。
- 本指南以 HTTPie 和 cURL 提供了 Kong Admin API 示例。如果要使用 HTTPie，请从此处安装它。
- 对" Kong Gateway"的任何引用均指 Kong Gateway 和 Kong Gateway（Enterprise）共有的功能或概念。

3. 准备管理 Kong Gateway

在开始使用 Kong Gateway 之前，请确认它已正确安装，并且已准备好进行管理。

在你开始之前

在开始本节之前，请确保：

- Kong Gateway 已安装并正在运行。
- Kong Manager（如果适用）和 Kong Admin API 端口正在侦听适当的端口/ IP / DNS 设置。
- 在本指南中，通过引用了 Kong Gateway 的实例。确保用控制面板实例的主机名替换。

验证 Kong Gateway 配置

使用管理 API
使用 Kong Manager
确保您可以使用 cURL 或 HTTPie 将请求发送到 Kong Admin API。

通过在终端窗口中发出以下命令来查看当前配置：

使用 cURL：

```
$ curl -i -X GET http://<admin-hostname>:8001
```

4. 通过 Kong Gateway 公开您的服务

在本主题中，您将学习如何使用路由公开服务。

如果您遵循“入门”工作流程，请确保在继续之前完成“ 准备管理 Kong Gateway”的准备工作。

如果您不遵循“入门”工作流，请确保已安装并启动了 Kong Gateway。

什么是服务和路线？

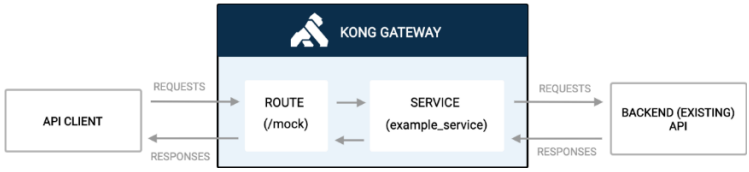
服务和路由对象使您可以通过 Kong Gateway 向客户端公开服务。在配置对 API 的访问权限时，您将从指定服务开始。在 Kong Gateway 中，服务是表示外部上游 API 或微服务的实体，例如，数据转换微服务，计费 API 等。

服务的主要属性是其 URL，服务在其中侦听请求。您可以使用单个字符串指定 URL，也可以分别指定其协议，主机，端口和路径。

在开始对服务提出请求之前，您需要为其添加一条路由。路由确定请求到达 Kong Gateway 后如何（以及是否）发送到其服务。单个服务可以具有多个路由。

配置服务和路由后，您将可以开始通过 Kong Gateway 发出请求。

该图说明了通过服务路由到后端 API 的请求和响应的流程。



添加服务

就本示例而言，您将创建一个指向 Mockbin API 的服务。Mockbin 是一个“回声”型公共网站，可将请求作为响应返回给请求者。该可视化将有助于了解 Kong Gateway 代理 API 请求的方式。

Kong Gateway 在 port 上公开了 RESTful Admin API :8001。网关的配置（包括添加服务和路由）是通过对 Admin API 的请求完成的。

1. 使用名称 example_service 和 URL 定义服务 http://mockbin.org。

使用 cURL：

```
$ curl -i -X POST http://<admin-hostname>:8001/services \
```

```
 --data name=example_service \

 --data url='http://mockbin.org'
```

如果服务创建成功，您将收到 201 成功消息。

2. 验证服务的端点。
   使用 cURL：

```
$ curl -i http://<admin-hostname>:8001/services/example_service
```

添加路由

为了使服务可以通过 Kong Gateway 访问，您需要为其添加一条路由。
使用客户需要请求的特定路径/mock 为服务（example_service）定义一条路由（）。请注意，必须至少设置主机，路径或方法之一，才能使路由与服务匹配。
使用 cURL：

```
$ curl -i -X POST http://<admin-
hostname>:8001/services/example_service/routes \

  --data 'paths[]=/mock' \

  --data 'name=mocking'
```

将返回一条 201 消息，表明路由已成功创建。

确认路由将请求转发到服务

使用 Admin API，发出以下命令：

使用 cURL：

```
$ curl -i -X GET http://<admin-hostname>:8000/mock
```

5. 保护您的服务

在本主题中，您将学习如何使用 Rate Limiting 插件来实施速率限制。

如果根据入门指南学习，请确保之前完成“Kong 入门指南 - 公开您的服务 ”学习。

什么是速率限制？

速率限制使您可以限制您的上游服务从 API 使用者接收的请求数量，或每个用户可以调用 API 的频率。

为什么要使用速率限制？

速率限制可保护 API 免受意外或恶意的过度使用。在没有速率限制的情况下，每个用户都可以根据自己的意愿进行多次请求，这可能导致大量的请求挤占其他消费者的请求。启用速率限制后，每秒将 API 调用限制为固定数量的请求。

设置速率限制

在端口上调用 Admin API 8001 并配置插件以在节点上每分钟限制五（5）个请求，这些请求存储在本地和内存中。

使用 cURL：

```
$ curl -i -X POST http://<admin-hostname>:8001/plugins \

--data "name=rate-limiting" \

--data "config.minute=5" \

--data "config.policy=local"
```

验证速率限制

要验证速率限制，请从 CLI 访问 API 6 次，以确认请求受到速率限制。

使用 cURL：

```
$ curl -i -X GET http://<admin-hostname>:8000/mock/request
```

在第 6 次请求之后，您应该收到“超出 API 速率限制”错误：

```
{

"message": "API rate limit exceeded"

}
```

6.通过代理缓存提高性能

在本文中，您将学习如何使用 Kong Gateway 的代理缓存插件来使用代理缓存来提高响应效率。

如果您根据入门指南学习，请确保已完成"Kong 入门指南 - 保护您的服务 "学习。

什么是代理缓存？

Kong Gateway 通过缓存提供了快速的性能。代理缓存插件使用反向代理缓存实现来提供这种快速性能。它根据请求方法，可配置的响应代码，内容类型来缓存响应实体，并且可以按消费者或 API 进行缓存。

缓存实体将存储一段可配置的时间。达到超时后，Kong Gateway 会将请求转发到上游，对结果进行缓存并从缓存进行响应，直到超时为止。该插件可以在 Redis 中将缓存的数据存储在内存中，或提高性能.

为什么要使用代理缓存？

使用代理缓存，以使上游服务不会因重复的请求而面临请求压力，而 Kong Gateway 可以响应缓存的结果。

设置代理缓存插件

在端口上调用 Admin API 8001 并配置插件以全局启用内存中缓存，Content-Type 的超时时间为 30 秒 application/json。
使用 cURL：

```
$ curl -i -X POST http://<admin-hostname>:8001/plugins \

--data name=proxy-cache \

--data config.content_type="application/json" \

--data config.cache_ttl=30 \

--data config.strategy=memory
```

验证代理缓存

让我们检查代理缓存是否有效。

1. 使用 Admin API 访问/ mock 路由，并记下响应标头。

使用 cURL

```
$ curl -i -X GET http://<admin-hostname>:8000/mock/request
```

特别是要密切关注的响应头 X-Cache-Status，X-Kong-Proxy-Latency 以及 X-Kong-Upstream-Latency：

```
HTTP/1.1 200 OK

...

X-Cache-Key: d2ca5751210dbb6fefda397ac6d103b1

X-Cache-Status: Miss

X-Content-Type-Options: nosniff

...

X-Kong-Proxy-Latency: 25

X-Kong-Upstream-Latency: 37
```

2. 再次访问/ mock 路由。
   这个时候，注意到的响应头的差异 X-Cache-Status，X-Kong-Proxy-Latency 和 X-Kong-Upstream-Latency。缓存状态为 hit，这意味着 Kong Gateway 直接从缓存中响应请求，而不是将请求代理到上游服务。

此外，请注意响应中的最小延迟，这使 Kong Gateway 可以提供最佳性能：

```
HTTP/1.1 200 OK

...

X-Cache-Key: d2ca5751210dbb6fefda397ac6d103b1

X-Cache-Status: Hit

...

X-Kong-Proxy-Latency: 0

X-Kong-Upstream-Latency: 1
```

3. 为了更快地进行测试，可以通过调用 Admin API 删除缓存：

使用 cURL：

```
$ curl -i -X DELETE http://<admin-hostname>:8001/proxy-cache
```

7.使用身份验证保护服务

在本主题中，您将了解 API 网关身份验证，设置密钥身份验证插件以及添加使用者。

如果您根据入门指南学习，请确保已完成“Kong 入门指南 - 通过代理缓存提高性能 ”。

什么是身份验证？

API 网关身份验证是控制允许使用您的 API 传输的数据的重要方法。基本上，它使用一组预定义的凭据来检查特定使用者是否有权访问 API。

Kong Gateway 有一个插件库，这些插件提供了实现 API 网关身份验证的最广为人知和使用最广泛的方法的简单方法。以下是一些常用的：

- 基本认证
- 密钥认证
- OAuth 2.0 身份验证
- LDAP 认证高级
- OpenID 连接

身份验证插件可以配置为应用于 Kong Gateway 内的服务实体。反过来，服务实体与它们表示的上游服务是一对一映射的，从本质上讲意味着认证插件直接应用于那些上游服务。

为什么要使用 **API** 网关身份验证？

启用身份验证后，除非客户端首先成功进行身份验证，否则 Kong Gateway 不会代理请求。这意味着上游（API）不需要对客户端请求进行身份验证，也不会浪费用于验证凭据的关键资源。

Kong Gateway 可以查看所有身份验证尝试（成功，失败等等），从而可以对这些事件进行分类和控制，以证明适当的控制措施已经存在并实现合规性。身份验证还使您有机会确定如何处理失败的请求。这可能意味着仅阻止请求并返回错误代码，或者在某些情况下，您可能仍希望提供有限的访问权限。

在此示例中，您将启用密钥验证插件。API 密钥身份验证是进行 API 身份验证的最流行的方法之一，可以实现以根据需要创建和删除访问密钥。

有关更多信息，请参见什么是 API 网关身份验证？。

设置密钥认证插件

1. 在端口上调用 Admin API 8001 并配置插件以启用密钥身份验证。对于此示例，将插件应用于您创建的/ mock 路由。

使用 cURL：

```
$curl -X POS Thttp://<admin-hostname>:8001/routes/mocking/plugins \

 --data name=key-auth
```

2. 尝试再次访问该服务：
   使用 cURL：

```
$ curl -i http://<admin-hostname>:8000/mock
```

由于添加了密钥认证，因此您将无法访问它：

```
HTTP/1.1 401 Unauthorized

 ...

 {

    "message": "No API key found in request"

 }
```

在 Kong 代理请求此路由之前，它需要一个 API 密钥。对于此示例，由于安装了密钥身份验证插件，因此需要首先创建具有关联密钥的使用者。

设置使用者和凭证

1. 要创建使用者，请调用 Admin API 和使用者的端点。下面创建了一个新的消费者，称为 Consumer。

使用 cURL：

```
$ curl -i -X POST -d "username=consumer&custom_id=consumer"
http://<admin-hostname>:8001/consumers/
```

2. 设置后，调用 Admin API 为上面创建的使用者设置密钥。对于此示例，将密钥设置为 apikey。如果未输入任何密钥，则 Kong 将自动生成密钥。

使用 cURL：

```
$ curl -i -X POST http://<admin-
hostname>:8001/consumers/consumer/key-auth -d 'key=apikey'
```

结果：

```
HTTP/1.1 201 Created

...

{

   "consumer": {

      "id": "2c43c08b-ba6d-444a-8687-3394bb215350"

   },

   "created_at": 1568255693,

   "id": "86d283dd-27ee-473c-9a1d-a567c6a76d8e",

   "key": "apikey"

}
```

现在，您已经为使用者提供了 API 密钥，以访问该路由。

验证密钥验证

要验证密钥身份验证插件，请使用密钥值为的标头再次访问模拟路由。
apikeyapikey

使用 cURL：

```
$ curl -i http://<admin-hostname>:8000/mock/request -H 'apikey:apikey'
```

您应该得到一条 HTTP/1.1 200 OK 消息作为回应。

（可选）禁用插件

如果您按照主题逐个遵循此入门指南，则在以后的所有请求中都需要使用此 API 密钥。如果您不想一直指定密钥，请在继续操作之前禁用插件。

1. 找到插件 ID 并复制。

使用 cURL：

```
$ curl -X GET http://<admin-hostname>:8001/routes/mocking/plugins/
```

输出：

```
"id": "2512e48d9-7by0-674c-84b7-00606792f96b"
```

2. 禁用插件。
   使用 cURL：

```
$ curl -X PATCH http://<admin-hostname>:8001/routes/mocking/plugins/{<plugin-id>} \

 --data "enabled=false"
```

8.设置智能负载平衡

在本主题中，您将学习配置上游服务，并创建多个目标来进行负载平衡。

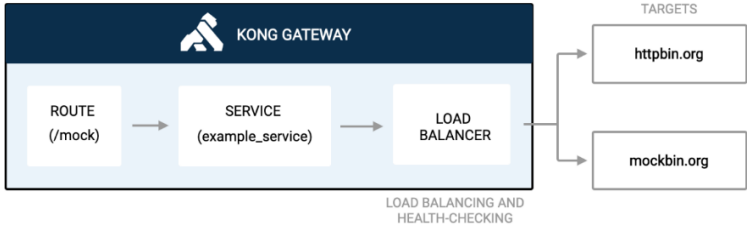
如果根据入门指南学习，请确保已完成"Kong 入门指南 - 使用身份验证的安全服务" 。

什么是上游？

上游指的是你的上游 API/服务,是客户端到网关请求转发的目标服务。在 Kong Gateway 中，一个上游对象代表一个虚拟主机名，可用于运行状况检查，断路和负载平衡多个服务（目标）上的传入请求。

在本主题中，您将配置先前创建的服务（example_service）指向上游而不是主机。在我们的例子而言，上游将指向两个不同的目标，httpbin.org 和 mockbin.org。在实际环境中，上游将指向在多个系统上运行的同一服务。

这是说明设置的图：



为什么要在上游目标之间实现负载平衡？

在以下示例中，您将使用跨两个不同服务器或上游目标部署的应用程序。Kong Gateway 需要在两台服务器之间实现负载平衡，以便如果其中一台服务器不可用，它将自动检测到问题并将所有流量路由到工作服务器。

配置上游服务

在本部分中，您将创建一个名为的上游 upstream，并向其添加两个目标。

1. 在端口上调用 Admin API 8001 并创建名为的上游 upstream。

使用 cURL：

```
$ curl -X POST http://<admin-hostname>:8001/upstreams \

 --data name=upstream
```

2. 更新您先前创建的服务以指向该上游。

使用 cURL：

```
$ curl -X PATCH http://<admin-hostname>:8001/services/example_service \

 --data host='upstream'
```

3. 向上游添加两个目标，每个目标都有端口 80：mockbin.org:80 和 httpbin.org:80。

使用 cURL：

```
$ curl -X POST http://<admin-hostname>:8001/upstreams/upstream/targets \

 --data target='mockbin.org:80'
```

```
$ curl -X POST http://<admin-hostname>:8001/upstreams/upstream/targets \

 --data target='httpbin.org:80'
```

现在，您有一个具有两个目标的上游 httpbin.org 和 mockbin.org，以及一个指向该上游的服务。

验证上游服务

1. 配置上游后，通过 http://:8000/mock 使用 Web 浏览器或 CLI 访问路由来验证其是否正常工作。
2. 继续访问端点，站点应从更改 httpbin 为 mockbin。