体验下，大厂在使用功能的API网关！

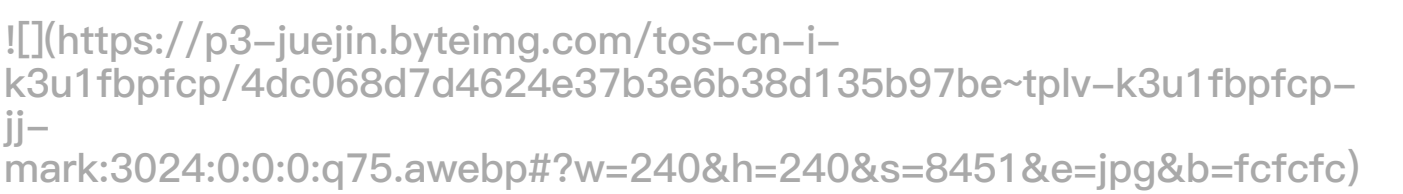
==========

作者：小傅哥
博客：[bugstack.cn](http://cxyroad.com/ ”https://bugstack.cn”)

> 沉淀、分享、成长，让自己和他人都能有所收获！

大家好，我是技术UP主小傅哥。

还是在22年的时候，小傅哥做了一套基于 Netty 协议转换和通信的 API网关，分享给伙伴们学习使用，增加一些业务开发以外的知识积累。不过很多伙伴都问过小傅哥，为啥要自研网关呢？SpringCloud Gateway 不就可以用吗？那你知道为什么自研吗？



不少伙伴问为啥自研， 但其实从我进入互联网大厂，核心的分布式技术框架，几乎就全部都是公司自研。从 rpc、mq、缓存组件（配合redis集群）、配置中心、分库分表、任务调度、全链路监控，再到我们提到的 API 网关，全部的都是自研。

后来才知道，因为之前用过一些开源组件，在流程承载方面发生过重大事故。因为是开源的组件，没法对每一个细节进行把控。而全部的自研，就会有非常强的把控力度，各个细节实现都可以做具体的优化方案，同时所有的组件自研，还可以更好的串联起来使用。另外还有一个点，就是这些开源的组件，更容易被攻击，如果有漏洞要升级，那公司全升级一遍的成本，不亚于一次大规模裁员的赔偿！当然，一些中小厂还是用市面开源的就好，因为自研的成本并不低！

那么为了让大家更好的了解下大厂的API，今天我们就来体验一款大厂开发的元原生API网关。有了这样一个学习，在看API网关项目，也会更清楚自己在做什么。

* 官网：[higress.io/zh-cn/](http://cxyroad.com/ "https://higress.io/zh-cn/")
* 源码：[github.com/alibaba/hig...](http://cxyroad.com/ "https://github.com/alibaba/higress")
* 部署：[github.com/higress-gro...](http://cxyroad.com/ "https://github.com/higress-group/higress-standalone") - `独立运行版，用于测试`
* 案例：[gitcode.net/KnowledgePl...](http://cxyroad.com/ "https://gitcode.net/KnowledgePlanet/road-map/xfg-dev-tech-higress") - `部署测试`

> Higress 是一套比较庞大的工程，为了让大家可以方便的体验到，小傅哥会教给大家怎么做一个独立的部署和配置网关负载。此外文末还提供了基于 Netty 的 API网关学习教程。

一、Higress 介绍

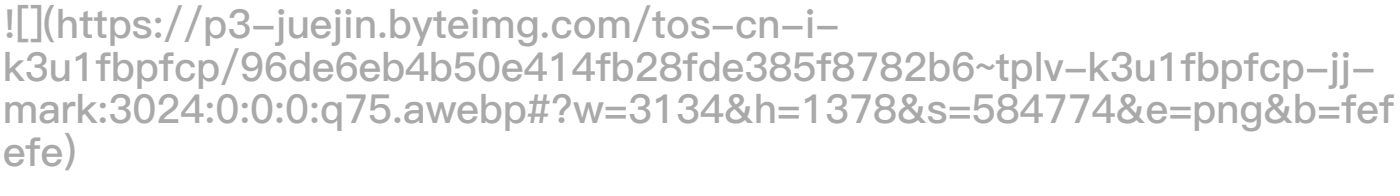
Higress 是基于阿里内部两年多的 Envoy Gateway 实践沉淀，以开源 [Istio](http://cxyroad.com/ "https://github.com/istio/istio") 与 [Envoy](http://cxyroad.com/ "https://github.com/envoyproxy/envoy") 为核心构建的云原生 API 网关。Higress 实现了安全防护网关、流量网关、微服务网关三层网关合一，可以显著降低网关的部署和运维成本。

* **生产等级**；支持每秒请求量达数十万级的大规模场景。彻底摆脱 reload 引起的流量抖动，配置变更毫秒级生效且业务无感。
* **平滑演进**；支持 Nacos/Zookeeper/Eureka 等多种注册中心，可以不依赖 K8s Service 进行服务发现，支持非容器架构平滑演进到云原生架构。同时支持 Nginx、ServiceMesh。这些方面是非常重要的，也就是你之前部署的到各个方面的服务，都可以被 Higress 统一管理，这也是我们设计网关的目的。
* **便于扩展**；提供 Wasm、Lua、进程外三种插件扩展机制，支持多语言编写插件，生效粒度支持全局级、域名级，路由级。插件支持热更新，变更插件逻辑和配置都对流量无损。

> 接下来，小傅哥就带着大家安装部署体验下。什么东西都是上手了才有感觉。

二、环境部署
------

* 云服务器：2c4g 最低，我是用的 2c8g 体验的。[yun.xfg.plus](http://cxyroad.com/ "https://yun.xfg.plus") - 价格实惠。
* 基础环境：Docker、Portainer、Git 【在小傅哥的 bugstack.cn 路书中都有讲解安装和使用】



* 整体安装完会如图所示。

1. 软件准备

1.1 方式一

在你的 Linux 服务器，通过 Git 命令检出安装项目；

```
git clone https://github.com/higress-group/higress-standalone.git
```

1.2 方式二

手动下载；[github.com/higress-gro...](http://cxyroad.com/ "https://github.com/higress-group/higress-standalone/archive/refs/heads/main.zip") 在通过 ssh 的 sftp 工具上传到云服务器。之后解压 unzip

2. 安装 higress

2.1 执行 configure.sh

```
[root@lavm-aqhgp9nber github]# cd higress-standalone-main/bin/
[root@lavm-aqhgp9nber bin]# ls
base.sh  configure.sh  logs.sh  reset.sh  shutdown.sh  startup.sh
status.sh  update.sh
[root@lavm-aqhgp9nber bin]# ./configure.sh
```

* 执行 ./configure.sh 后，注意选择 nacos 其他的默认值就可以，直接回车。
* 执行过程会自动检测，nacos 的安装和删掉。这些不用操作。
* 如果执行中遇到了失败或者自己选择错了，可以重新执行 ./configure.sh -r

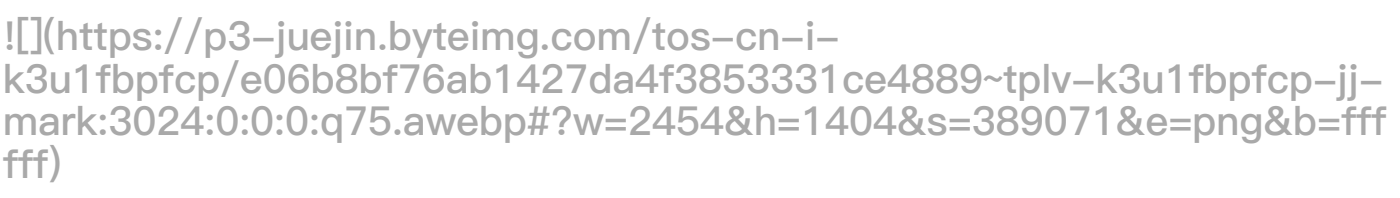
2.2 执行 startup.sh

```
[root@lavm-aqhgp9nber bin]# ./startup.sh
```

* 这一步就傻瓜式的了，直接就可以安装完成。

3. 安装 nginx

通过安装 Nginx 模拟出2个请求服务地址，如果你部署 SpringBoot 提供出 HTTP 接口也是可以。

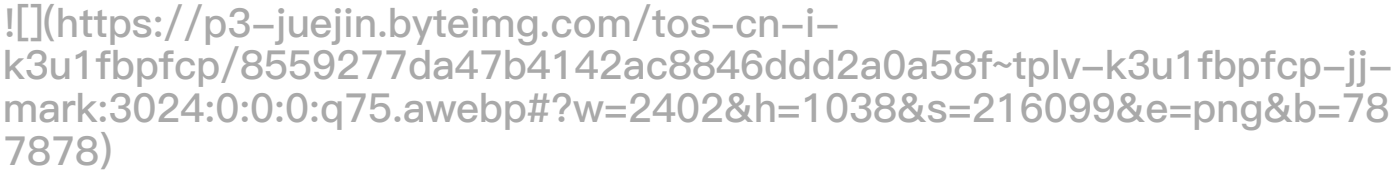


* 上传到服务器端执行脚本 `docker-compose -f nginx-docker-compose.yml up -d`

三、网关配置
---

1. 服务来源

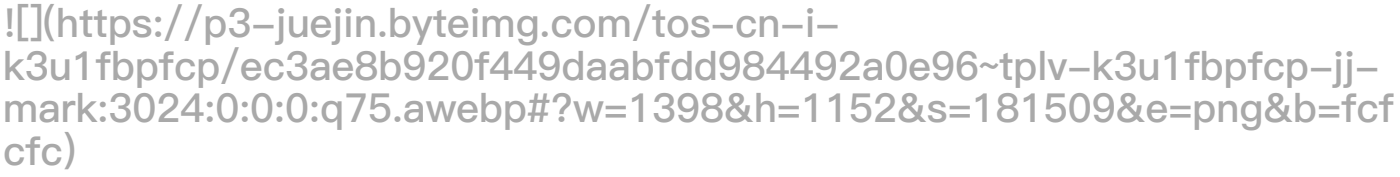
这里要配置的是，通过 [https://xxx/api](http://cxyroad.com/”https://xxx/api”) 访问到网关服务后，要访问到哪些服务来源上。



* 服务来源支持非常多的类型，包括；Nacos、Zookeeper、Consul、Eureka、固定地址、DNS 域名。这里小傅哥选择固定地址配置。
* 分别配置了 nginx-01、nginx-02 这样我们配置路由的时候可以负载到这2个地址。

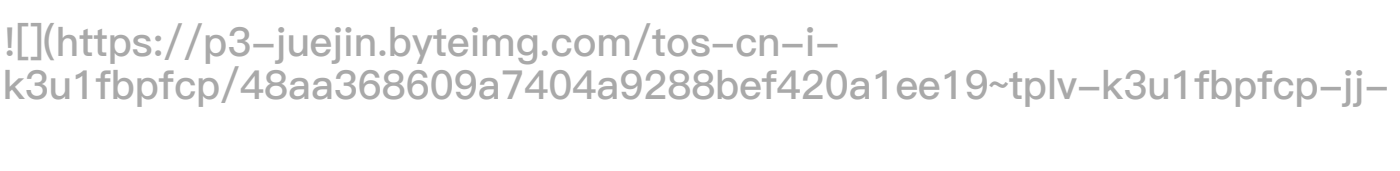
2. 路由配置

路由配置的作用就是指定你通过网关地址负载到对应的目标服务上，这里我们会让 [http://117.72.37.243/api/](http://cxyroad.com/”http://117.72.37.243/api/”) 请求负载到2个 nginx 上。



* 通过路由api地址，访问到目标服务。
* 这里可以配置的玩法还有很多，可以自己在尝试下。

3. 策略配置

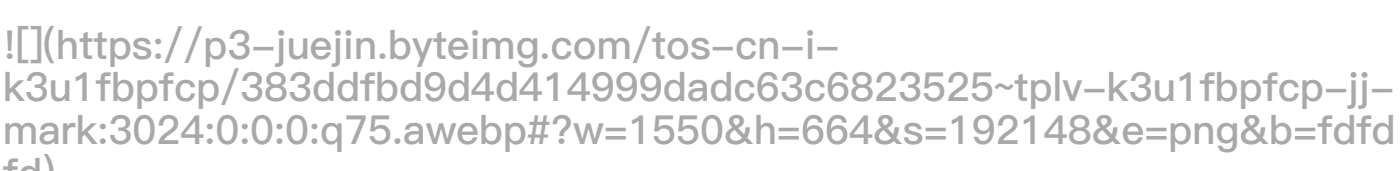


mark:3024:0:0:0:q75.awebp#?w=2140&h=866&s=167159&e=png&b=fefefe)

* 你可以为访问自己的路由接口配置对应的插件，比如重写URI、跨域、限流等各项功能。

四、服务验证
------

* 地址：[http://117.72.37.243/api/](http://cxyroad.com/”http://117.72.37.243/api/”) – 你需要换成自己的IP地址

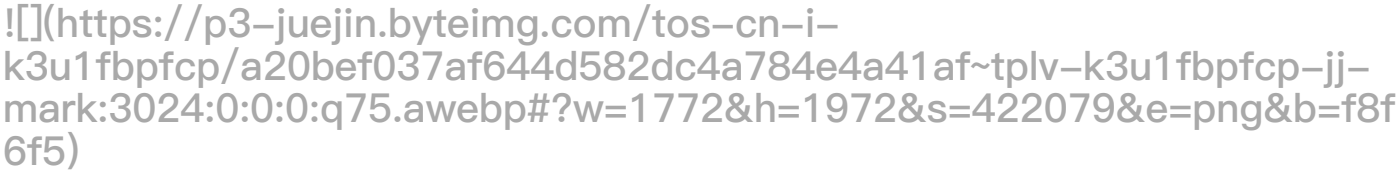


* 首先，我在 Nginx 的 HTML 中，配置了2个不同的请求结果，一个 01、一个 02
* 之后，访问网关地址加上 /api 接下来访问就会看到结果的变化了。

> 有了这个大厂网关的体验，大家就了解了一套网关是如何使用的，作用是什么啦。接下来，如果感兴趣技术的积累，想扩展下自己，也可以学习一套网关代码的实现。

五、网关学习

除了业务开发，小傅哥自己也是非常感兴趣于这样的网关技术组件的实现，所以在日常的工作中也积累了很多网关的设计。后来在22年做了一套轻量的网关系统，把核心的内核逻辑实现出来让大家学习。帮助了很多伙伴学习项目后找到了不错的工作。



整个**API网关**设计核心内容分为这么五块；

* `第一块`：是关于通信的协议处理，也是网关最本质的处理内容。这里需要借助 NIO 框架 Netty 处理 HTTP 请求，并进行协议转换泛化调用到 RPC 服务返回数据信息。
* `第二块`：是关于注册中心，这里需要把网关通信系统当做一个算力，每部署一个网关服务，都需要向注册中心注册一个算力。而注册中心还需要接收 RPC 接口的注册，这部分可以是基于 SDK 自动扫描注册也可以是人工介入管理。当 RPC 注册完成后，会被注册中心经过AHP权重计算分配到一组网关算力上进行使用。
* `第三块`：是关于路由服务，每一个注册上来的Netty通信服务，都会与他对应提供的分组网关相关联，例如：wg/(a/b/c)/user/... a/b/c 需要匹配到 Nginx 路由配置上，以确保不同的接口调用请求到对应的 Netty 服务上。
PS：如果对应错误或者为启动，可能会发生类似B站事故。
* `第四块`：责任链下插件模块的调用，鉴权、授信、熔断、降级、限流、切量等，这些服务虽然不算是网关的定义下的内容，但作为共性通用的服务，它们通常也是被放到网关层统一设计实现和使用的。
* `第五块`：管理后台，作为一个网关项目少不了一个与之对应的管理后台，用户接口的注册维护、mock测试、日志查询、流量整形、网关管理等服务。

> 项目学习地址：[bugstack.cn/md/assembly...](http://cxyroad.com/”https://bugstack.cn/md/assembly/api-gateway/api-gateway.html”)