# APISIX 的选型和持续集成

温铭 https://github.com/moonming

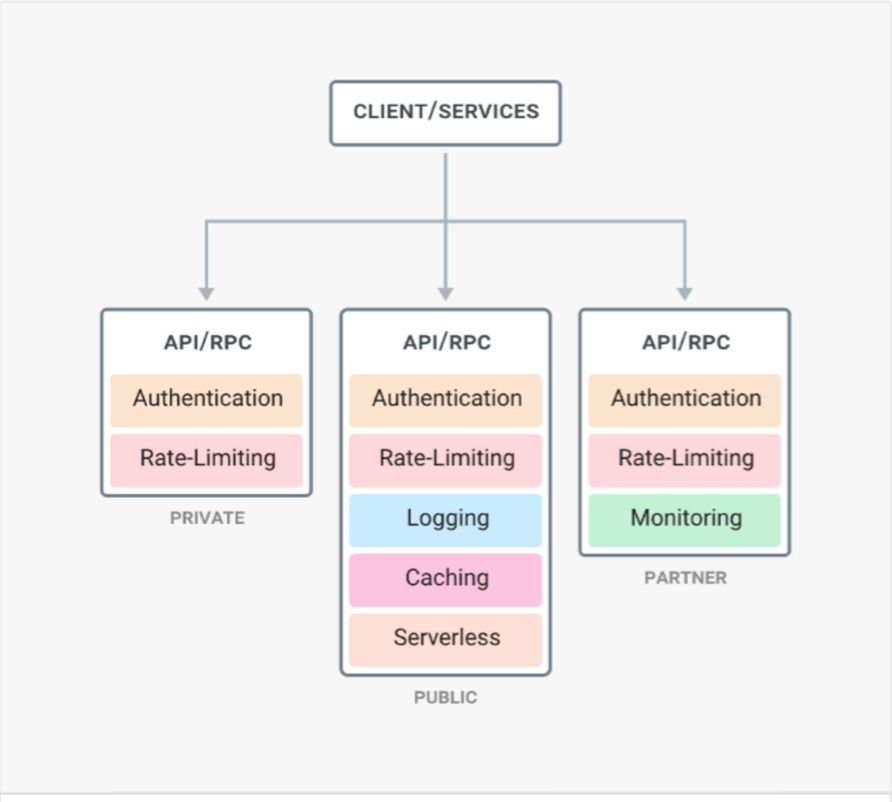
# 关于我

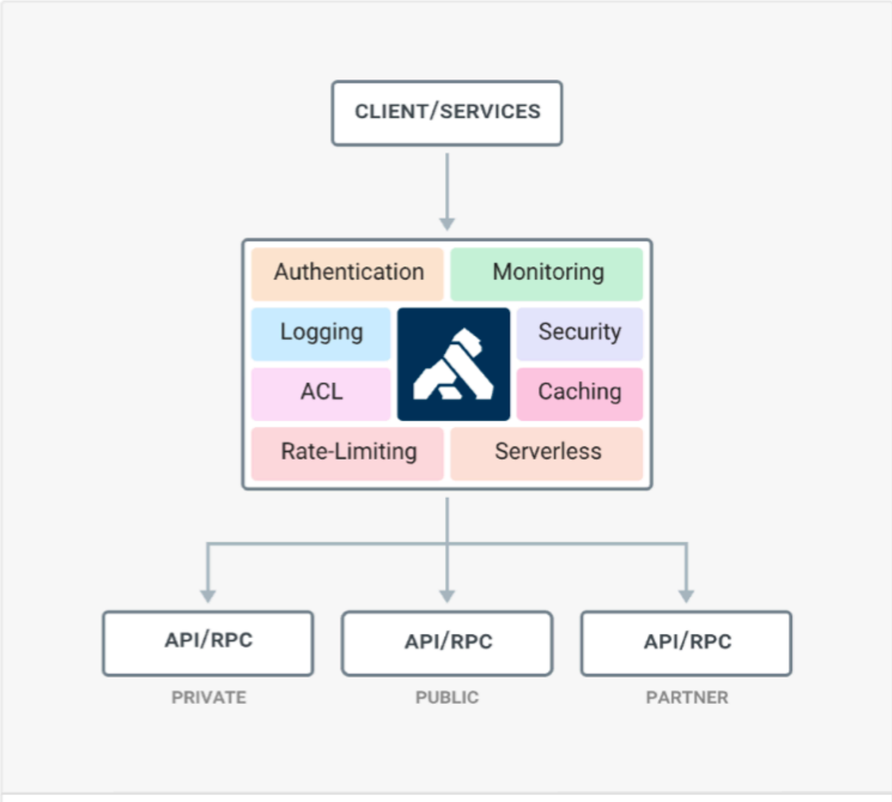
- 开源微服务 API 网关 APISIX 作者
- OpenResty 软件基金会发起人
- 《OpenResty 从入门到实战》专栏作者
- 曾在 OpenResty Inc. 担任合伙人
- 曾在奇虎 360 担任架构师，开源委员会发起人、委员

# 大纲

- API 网关的作用
- APISIX 的选型
- 持续集成的最佳实践

# 什么是 API 网关?


CLIENT/SERVICES
API/RPC
Authentication
Rate-Limiting
PRIVATE
API/RPC
Authentication
Rate-Limiting
Logging
Caching
Serverless
PUBLIC
API/RPC
Authentication
Rate-Limiting
Monitoring
PARTNER


CLIENT/SERVICES
Authentication
Monitoring
Logging
Security
ACL
Caching
Rate-Limiting
Serverless
API/RPC
PRIVATE
API/RPC
PUBLIC
API/RPC
PARTNER

# API网关的传统功能

- 让 API 请求更安全、更高效的得到处理
- 覆盖 Nginx 的所有功能
- 动态上游、动态 SSL 证书、动态限流限速
- 主动/被动健康检查、服务熔断
- 全生命周期管理：Google Apigee

# 云原生下的新功能

- 对接 Prometheus、Zipkin、Skywalking
- gRPC 代理和协议转换（REST <=> gRPC）
- 无状态、随意扩容和缩容
- 支持多云、混合云
- 容器优先，Kubernetes 友好

# 举例

- API 的身份认证：OpenID Connect、Auth0...
- 混合云

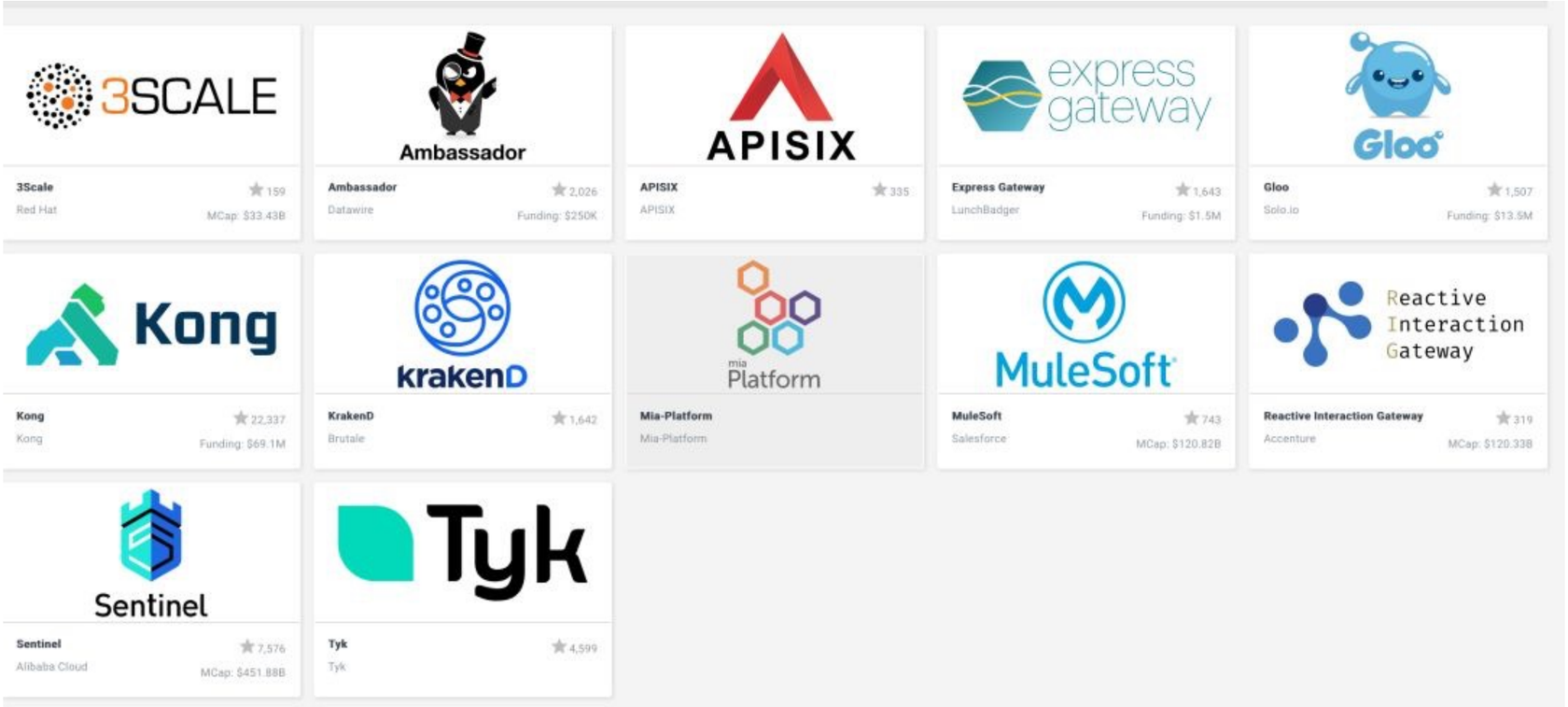
# 企业用户的需求

- 不锁定，可回退
- 保持核心稳定
- 支持企业个性需求的开发
- 社区快速响应
- 商业支持

# 选型

# 了解行业：Gartner 报告

# CNCF 全景图

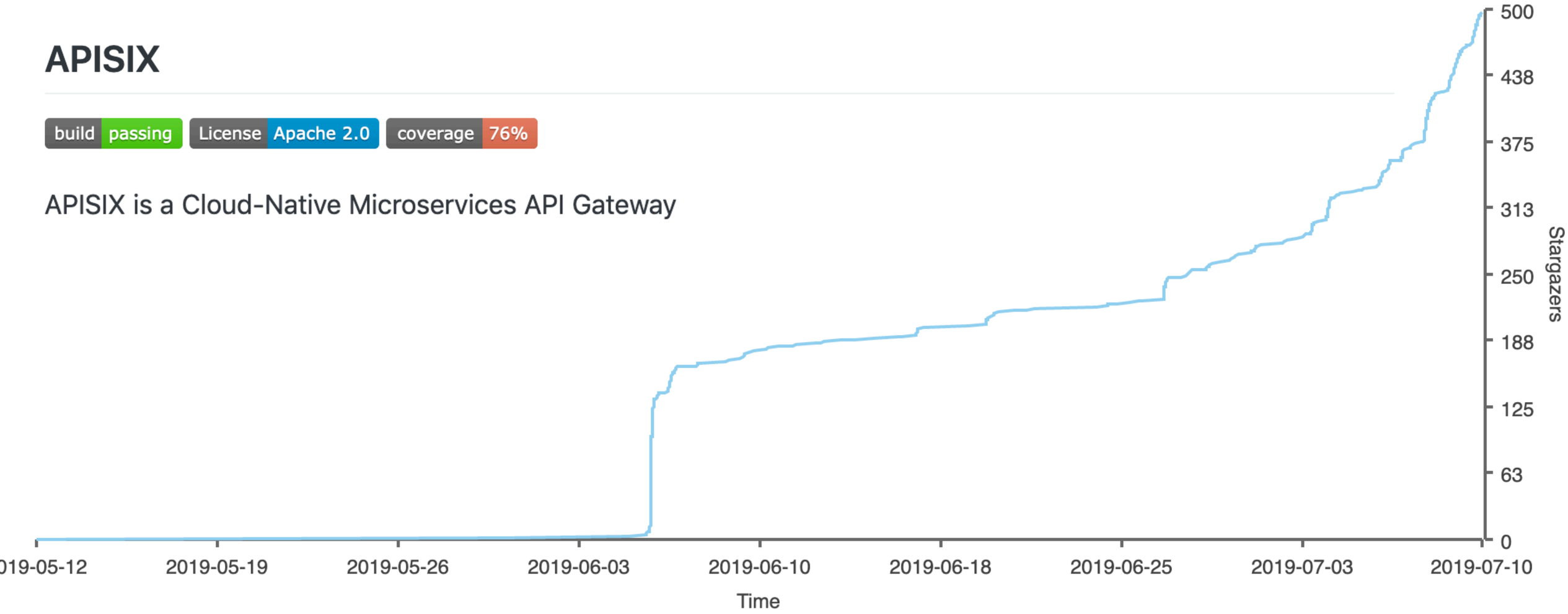

3SCALE
3Scale
Red Hat
159
MCap: $33.43B
Ambassador
Ambassador
Datawire
2,026
Funding: $250K
APISIX
APISIX
APISIX
335
express gateway
Express Gateway
LunchBadger
1,643
Funding: $1.5M
Gloo
Gloo
Solo.io
1,507
Funding: $13.5M
Kong
Kong
Kong
22,337
Funding: $69.1M
krakenD
KrakenD
Brutale
1,642
mia Platform
Mia-Platform
Mia-Platform
MuleSoft
MuleSoft
Salesforce
743
MCap: $120.82B
Reactive Interaction Gateway
Reactive Interaction Gateway
Accenture
319
MCap: $120.33B
Sentinel
Sentinel
Alibaba Cloud
7,576
MCap: $451.88B
Tyk
Tyk
Tyk
4,599


APISIX

# APISIX

build passing | License Apache 2.0 | coverage 76%

APISIX is a Cloud-Native Microservices API Gateway




# 产品选型

| 产品/特性 | 优势 | 劣势 | 二次开发难度 |
| --- | --- | --- | --- |
| apigee | 谷歌背书、<br>全生命周期 | 性能差、不开源 | 不支持 |
| Kong | 开源社区活跃、<br>产品思路清晰 | 代码量大、封装多、<br>依赖关系型数据库、<br>底层架构跟不上趋势 | 支持，难度大 |
| APISIX | 基于 etcd、代码易<br>读、插件热加载 | 开源时间短 | 支持，难度小 |

# 如何做网关的技术选型？

# API 网关的核心组件

- 路由：遍历、哈希、前缀树，以及他们的组合
- 插件：允许自定义、PDK（插件开发包）、热加载
- schema：参数校验、前后端配合
- 存储：关系型数据库、键值数据库、etcd

# APISIX 的选型原则

- APISIX：云原生、高性能、开源的 API 网关
- 云原生友好
- 开发者第一：架构简洁、方便扩展、代码易读、部署简单
- 极致性能：时间复杂度、FFI 和 Lua 代码

# APISIX 的选型

- 路由：lua-resty-r3，FFI
- 插件：灵感来自 Kong，大幅度简化编写难度，热加载
- schema：rapidjson， json schema
- 存储：etcd， lua-resty-etcd

# APISIX 的独有功能

- 超高性能，是 Kong 的 10 倍
- 插件热更新
- 路由插件化
- 变更的版本控制
- 以身份为基础的零信任

# 测试

# 测试驱动开发

- 开源项目必须要 TDD
- OpenResty 有 70 个开源项目，都是兼职
- 没有专职 QA
- 提交功能代码的同时，必须有测试案例
- 代码覆盖率不能低于 70%

# APISIX 的实践

- 开发即测试
- 单元测试完全基于 test::nginx
- 代码风格检测：luackeck 和 lua-relang
- 代码覆盖率：luacov

- 合并 PR 的前提是跑完上述三种所有测试
- 发版本前跑性能测试，分析火焰图
- 定期跑 fuzzing 测试

# test::nginx 上手难度大

- 并非只使用 Lua 编写
- 对 test::nginx、perl、OpenResty 不熟悉的话，很可能写不出来测试案例
- 一直是代码贡献者的拦路虎

# 持续集成

# APISIX 的做法

- 强依赖 GitHub：issue、Milestone、code review、PR approved
- 强依赖 travis CI：单元测试、代码风格检测、多平台测试（ubuntu 和 mac）、前端打包、自动提交...
- coveralls.io

| COVERAGE | Δ | FILE |
| --- | --- | --- |
| 100.0 | | ...ravis/build/iresty/apisix/lua/apisix/core/json.lua |
| 100.0 | | ...is/build/iresty/apisix/lua/apisix/core/request.lua |
| 100.0 | | ...ome/travis/build/iresty/apisix/lua/apisix/core.lua |
| 96.15 | | ...travis/build/iresty/apisix/lua/apisix/core/ctx.lua |
| 94.87 | | ...s/build/iresty/apisix/lua/apisix/admin/plugins.lua |
| 92.77 | | ...vis/build/iresty/apisix/lua/apisix/core/schema.lua |
| 92.31 | | ...ravis/build/iresty/apisix/lua/apisix/core/http.lua |
| 91.79 | | /home/travis/build/iresty/apisix/lua/apisix.lua |
| 91.3 | | ...travis/build/iresty/apisix/lua/apisix/core/log.lua |
| 90.91 | | ...travis/build/iresty/apisix/lua/apisix/consumer.lua |

# 总结

- 资源少，不一定是坏事儿
- APISIX 的选型、测试和 CI 都是找“取巧”和自动化的方式
- 这三者很重要，比性能重要
- GitHub 和 SaaS 提供的，绝对不自己造轮子

# Q&A