OpenResty典型应用场景

张顺 AISpeech
OpenResty咨询委员会
2016年11月

介绍

OpenResty

„OpenResty 是一个全功能的 Web 应用服务器。它打包了标准的 Nginx 核心，很多的常用的第三方模块，以及它们的大多数依赖项。

通过众多进行良好设计的 Nginx 模块，OpenResty 有效地把 Nginx 服务器转变为一个强大的 Web 应用服务器，基于它开发人员可以使用 Lua 编程语言对 Nginx 核心以及现有的各种 Nginx C 模块进行脚本编程，构建出可以处理 C100k 以上并发请求的极端高性能的 Web 应用。"

反向代理

web服务器　　代理缓存

负载均衡

性能好

43.7%

c模块开发成本高

web应用开发?

简单的脚本语言?

跨平台

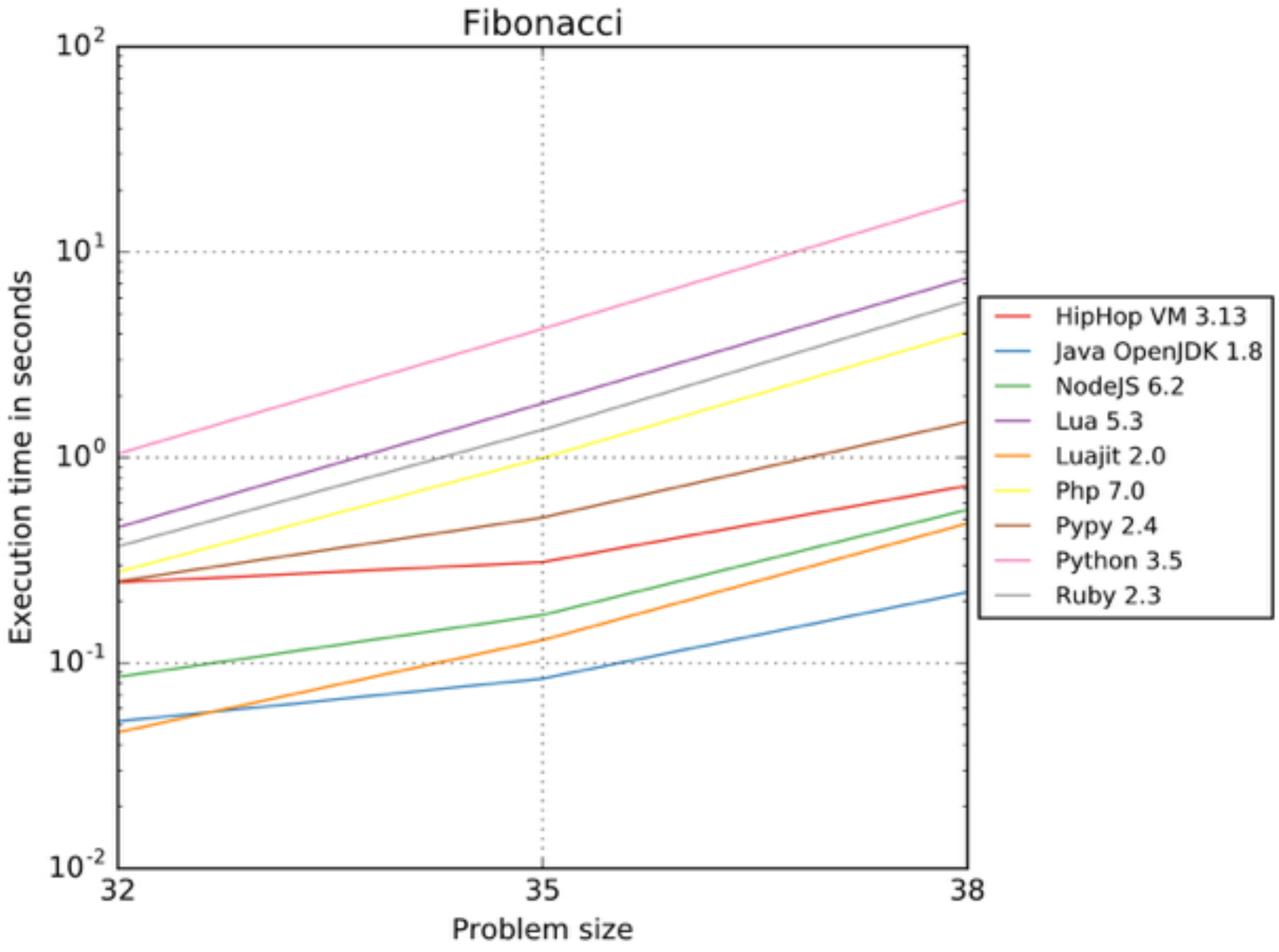
胶水语言

游戏脚本

Torch

LuaJIT

最快的JIT之一



https://github.com/gareins/dynamic_benchmarks

Nginx 核心模块

Nginx 第三方模块

lua-nginx-module

stream-lua-nginx-module

LuaJIT

lua-resty-*

C10K - C2000K

cosocket

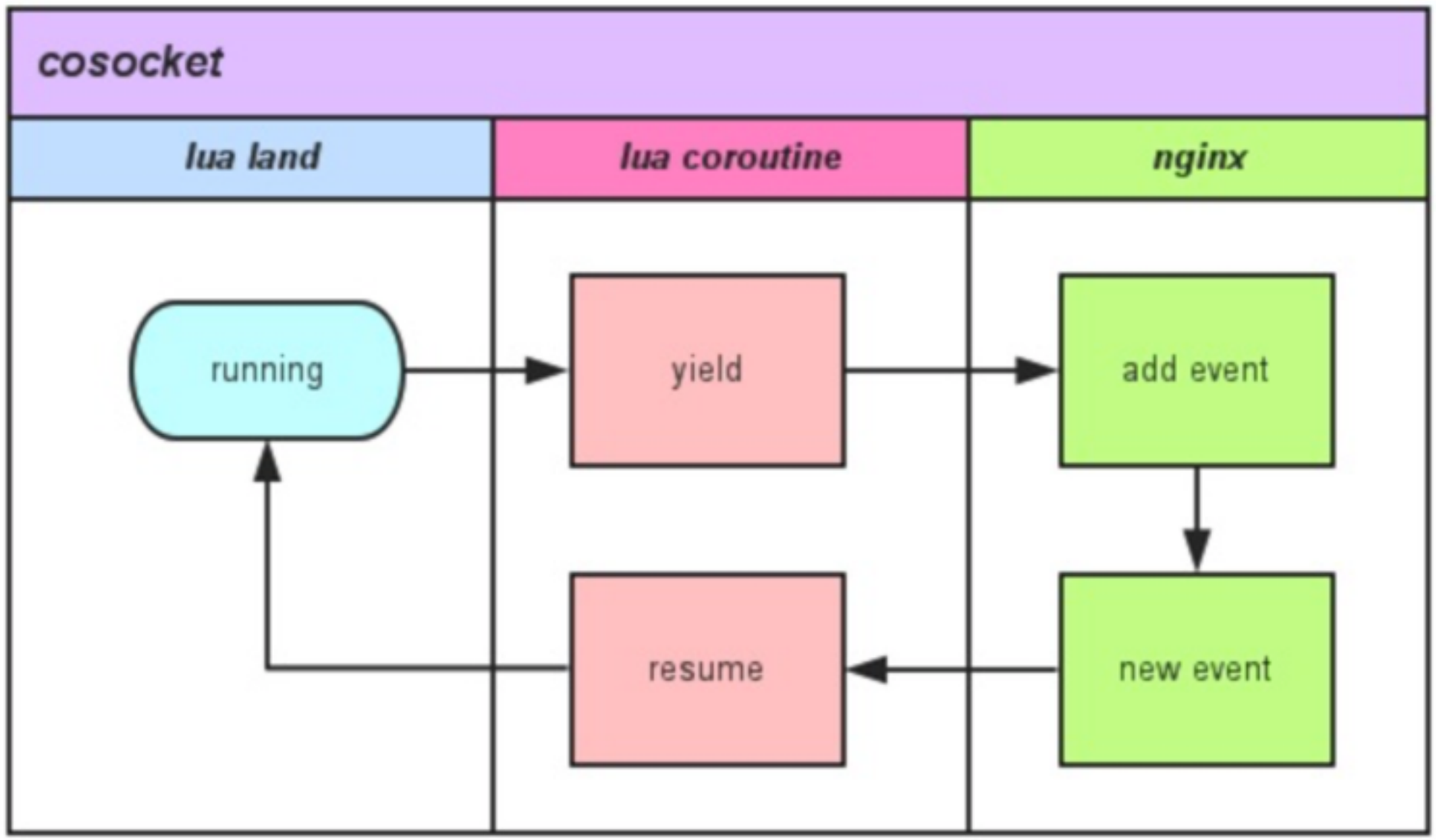
异步网络IO

coroutine

顺序写代码

不需要考虑异步

没有回调打乱思维


cosocket
lua land
lua coroutine
nginx
running
yield
add event
resume
new event

你喜欢回调吗

```
/* NodeJs */
var redis = require('redis');
var client = redis.createClient(6379,
                '127.0.0.1',{});
client.on("connect", function(error) {
    console.log("connect....");
});
```

```
client.get('foo', function(error, res) {
    if (error) {
        console.log(error);
    } else {
        console.log(res);
    }
    client.end();
});
```

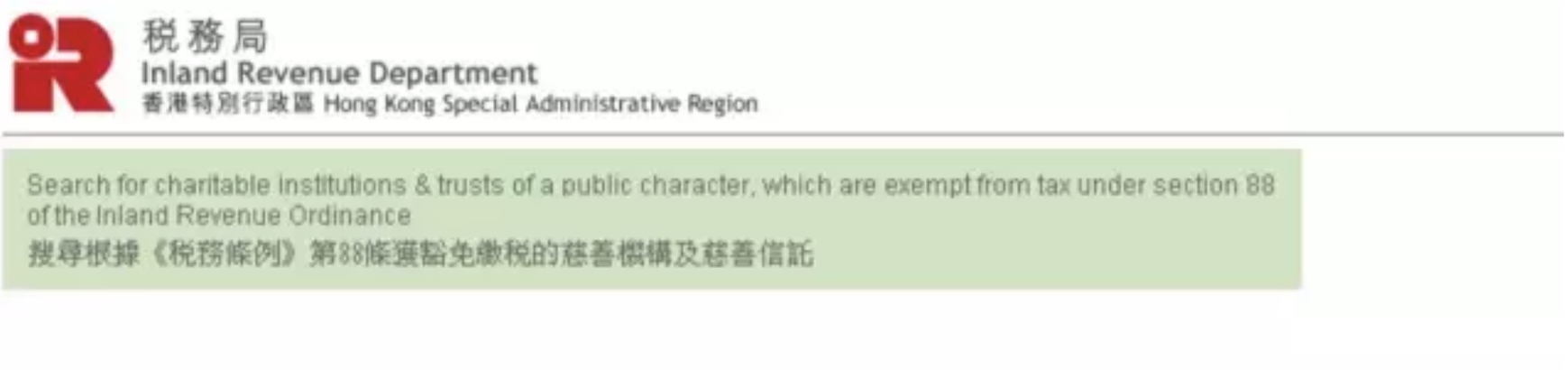
```
-- OpenResty
local redis = require('resty.redis')
local red = red:new()
local ok, err =
red:connect('127.0.0.1', 6379)
local res, err = red:get('foo')
ngx.say(res)
```

截止2015年底统计数据:

135个国家和地区, 共计127万次下载

京东，新浪，奇虎，锤子，优酷，又拍云，魅族，阿里云，网易，CloudFlare, Github, CSDN, 金山, 腾讯, 小米, 同程 …

中国首个的软件基金会



税務局
Inland Revenue Department
香港特別行政區 Hong Kong Special Administrative Region

Search for charitable institutions & trusts of a public character, which are exempt from tax under section 88 of the Inland Revenue Ordinance
搜尋根據《稅務條例》第88條獲豁免繳稅的慈善機構及慈善信託

Operated by 由以下團體主辦:

Name of organization: OPENRESTY SOFTWARE FOUNDATION LIMITED

慈善團體名字:

典型应用场景

1. 扩展nginx配置

```
location / {
    content_by_lua_block {
        ngx.say('Hello, OpenResty')
    }
}
```

```
location /download1 {
        limit_rate 10k;
}

location /download2 {
    access_by_lua_block {
        local h = os.date('*t').hour
        if 18 <= h and h <= 20 then
            ngx.var.limit_rate='10k'
        end
    }
}
```

download1修改限速值, 需要 nginx -s reload

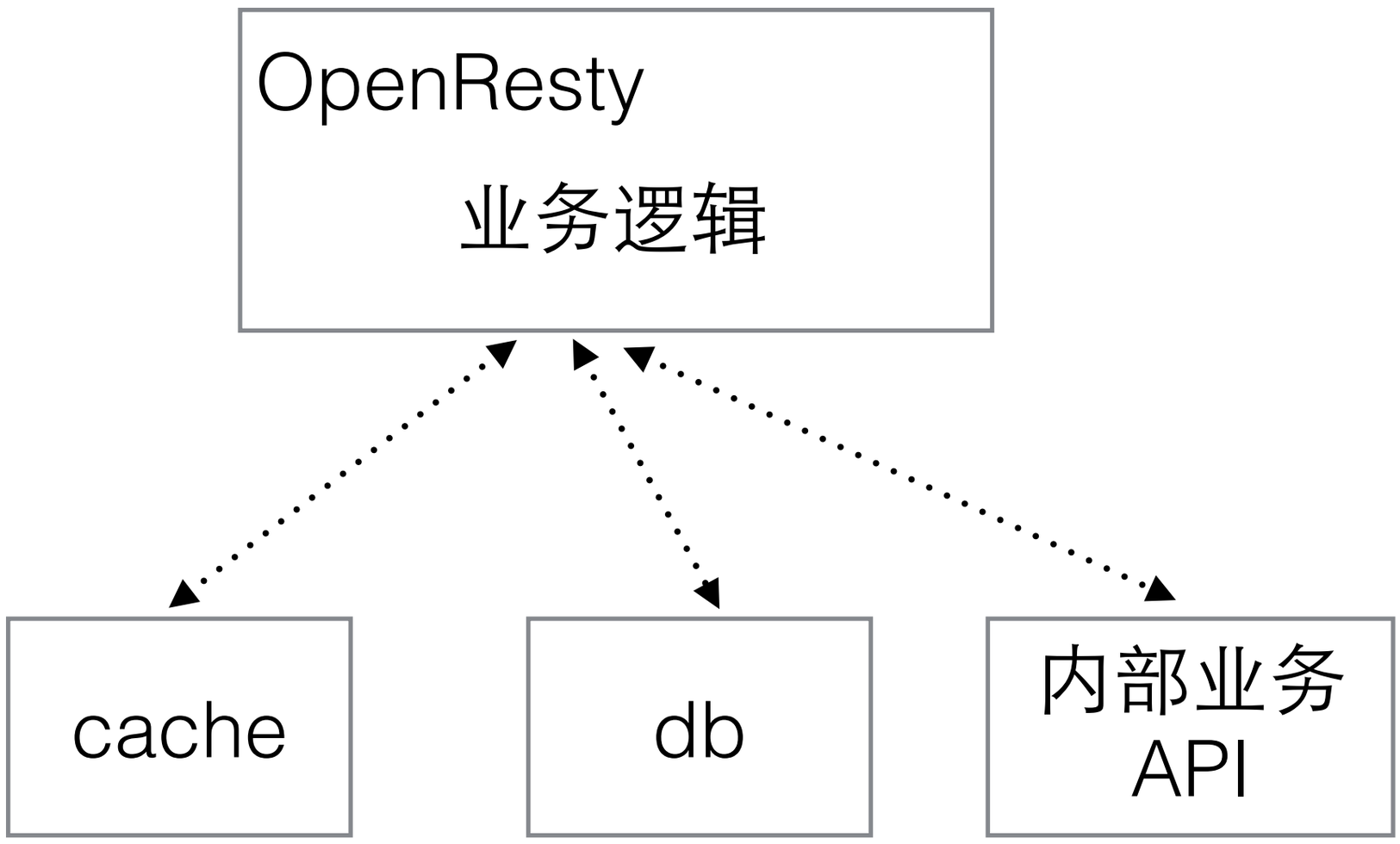
download2可以灵活访问内置变量, 不需要reload

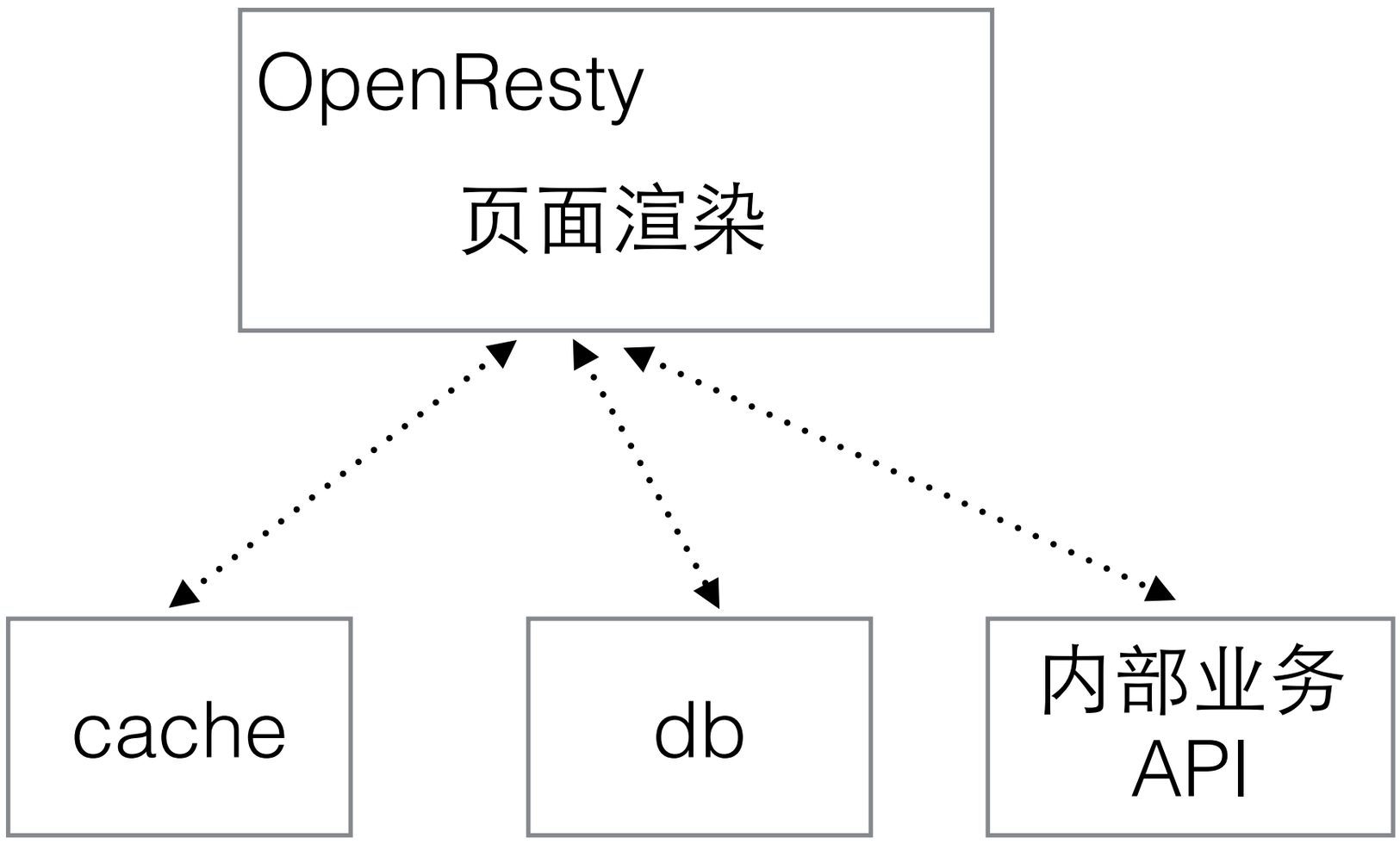
```
location /blog {
    proxy_pass http://mybackend;
    body_filter_by_lua_block {
        ngx.arg[1] = string.gsub(ngx.arg[1],
                                 '敏感词', '***');
    }
}
```

lua_use_default_type
lua_malloc_trim
lua_code_cache
lua_regex_cache_max_entries
lua_regex_match_limit
lua_package_path
lua_package_cpath
init_by_lua
init_by_lua_block
init_by_lua_file
init_worker_by_lua
init_worker_by_lua_block
init_worker_by_lua_file
set_by_lua
set_by_lua_block
set_by_lua_file
content_by_lua
content_by_lua_block
content_by_lua_file
rewrite_by_lua
rewrite_by_lua_block
rewrite_by_lua_file
access_by_lua
access_by_lua_block
access_by_lua_file
header_filter_by_lua
header_filter_by_lua_block
lua_ssl_crl
lua_ssl_protocols
lua_ssl_trusted_certificate
lua_ssl_verify_depth
lua_http10_buffering
rewrite_by_lua_no_postpone
access_by_lua_no_postpone
lua_transform_underscores_in_response_headers
body_filter_by_lua
body_filter_by_lua_block
body_filter_by_lua_file
log_by_lua
log_by_lua_block
log_by_lua_file
balancer_by_lua_block
balancer_by_lua_file
lua_need_request_body
ssl_certificate_by_lua_block
ssl_certificate_by_lua_file
ssl_session_fetch_by_lua_block
ssl_session_fetch_by_lua_file
ssl_session_store_by_lua_block
ssl_session_store_by_lua_file
lua_shared_dict
lua_socket_connect_timeout
lua_socket_send_timeout
lua_socket_send_lowat

2. 高性能web应用开发

web api




web页面
OpenResty
页面渲染
cache
db
内部业务
API

常用的驱动库

* lua-resty-mysql
* lua-resty-postgres
* lua-resty-orm
* lua-resty-mvc
* lua-resty-redis
* lua-resty-mongo
* lua-resty-fastdfs
* lua-resty-kafka

https://github.com/bungle/awesome-resty#databases-and-storages

模板引擎库

* lua-resty-template
* lemplate
* etlua
* lua-resty-tmpl

https://github.com/bungle/awesome-resty#templating

web framework

* lapis
* vanilla
* lor
* GIN
* lusty

https://github.com/bungle/awesome-resty#web-frameworks

4. 高并发入口

灵活的proxy/balancer

接入层

OpenResty

负载均衡

上游业务

php
php-fpm

php
php-fpm

java
tomcat

java
tomcat

```
upstream backend {
    server 0.0.0.1;   # just a place holder
    balancer_by_lua_block {
        local peers = {{'127.0.0.1', 8080},{'127.0.0.1', 8081}}
        local balancer = require "ngx.balancer"
        local i = ngx.crc32_long(ngx.var.remote_addr) % #peers + 1
        balancer.set_current_peer(unpack(peers[i]))
    }
    keepalive 10;
}
server {
    listen 80;
    location / {
        proxy_pass http://backend/;
    }
}
server {
    listen 127.0.0.1:8080;
    location = / { echo "this is server1";}
}
server {
    listen 127.0.0.1:8081;
    location = / { echo "this is server2";}
}
```

灵活实现负载均衡算法
无需nginx -s reload

视频演示 OpenResty作为语音服务的入口

https://v.qq.com/x/page/f0155wqxlmz.html

3. 缓存

* proxy_cache - nginx 反向代理

* lua-resty-lrucache - woker进程内
* ngx.shared.dict - worker进程间
* lua-resty-memcached
* lua-resty-redis

缓存失效风暴

* lua-resty-lock - non-blocking lock

5. WAF

```
access_by_lua_block {
    local black_ips = {["127.0.0.1"]=true}
    local ip = ngx.var.remote_addr
     if true == black_ips[ip] then
         ngx.exit(ngx.HTTP_FORBIDDEN)
    end
};
```

* NAXSI - nginx
* VeryNginx - openresty
* lua-resty-waf - openresty

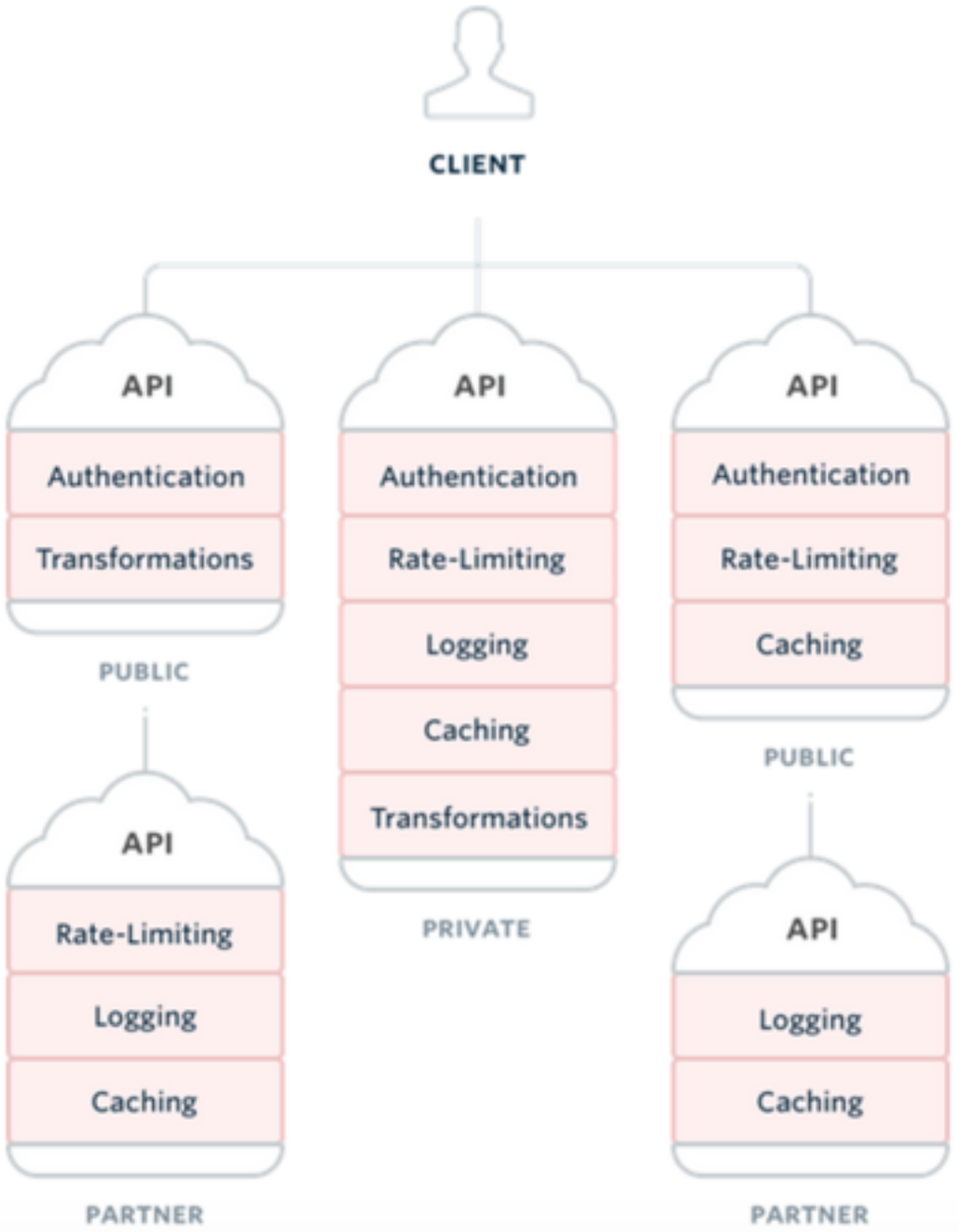
OpenResty official WAF?

Streaming Regex

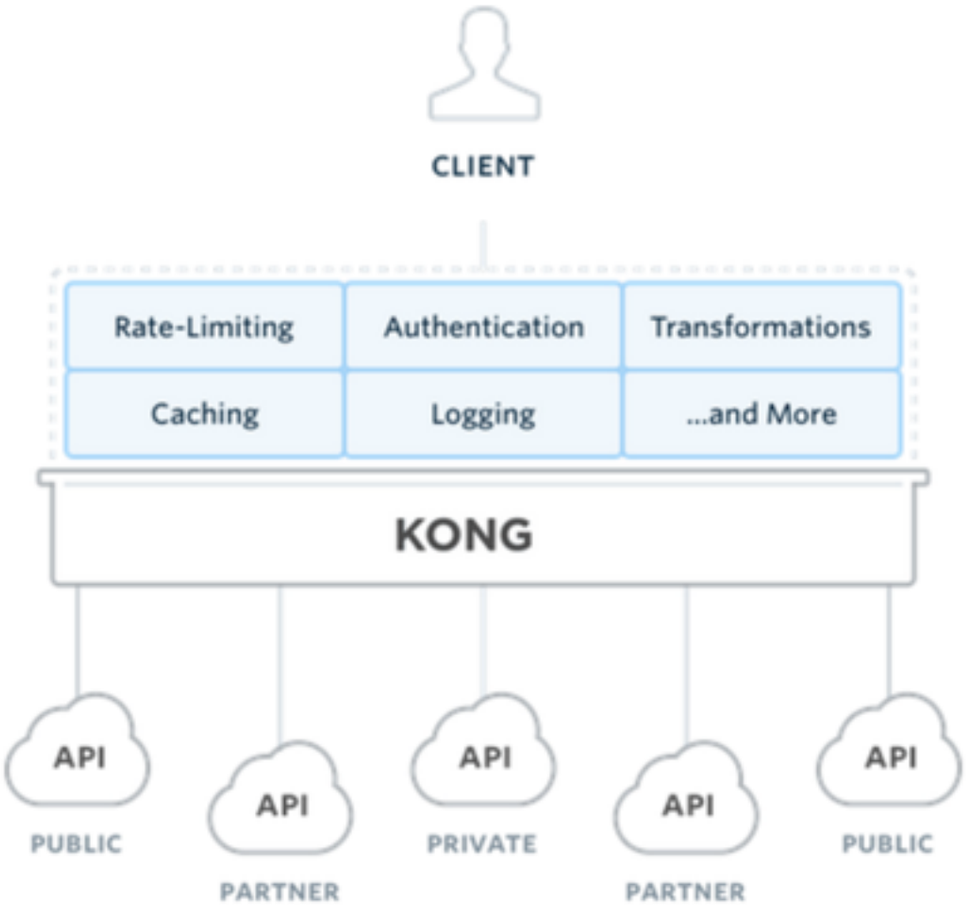
DSL for WAF

6. API Gateway

Current Architecture



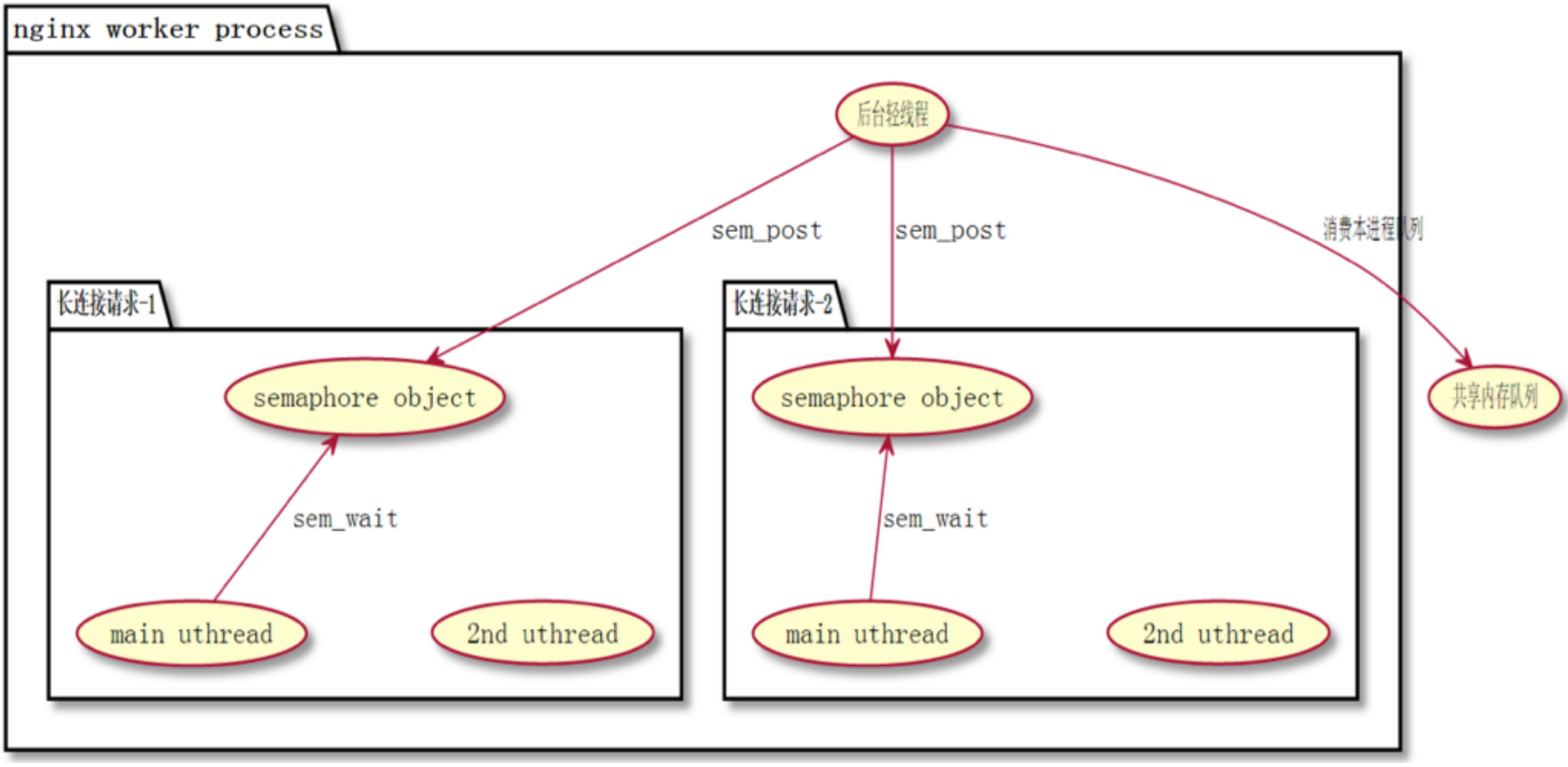
Kong Architecture



* Mashape/kong
* sumory/orange
* adobe-apiplatform/apigateway

7. 百万长连接推送(测试)

引自《基于OpenResty的百万级长连接推送》 朱德江@酷狗


nginx worker process
后台轻线程
sem_post
sem_post
消费本进程队列
长连接请求-1
长连接请求-2
semaphore object
semaphore object
共享内存队列
sem_wait
sem_wait
main uthread
2nd uthread
main uthread
2nd uthread

* cosocket - 全双工socket
* ngx.thread - 轻线程
* ngx.semaphore - 进程内 “轻线程”间通讯
  sessionid - serverid+workerid+uid + incrnum
* ngx.shared.dict- 进程间通讯
* websocket - 与client端通讯协议

* 200w 连接消耗约 40G 内存（约20k 每连接）
* 维持心跳消耗 7% CPU

* send 100byte message 20w/2s
* CPU: 50%
* Mem: 基本不变
* 得益于各级 buffer，实际上需要动态申请的内存很少

8. tcp server

stream-lua-nginx-module

not only http

lua脚本简单灵活

luajit高性能

cosocket异步非阻塞

顺序写代码

4层负载均衡

自定义协议的tcp server

ssl tcp server

Datanet, CRDT based data synchronization system

http://datanet.co/

章亦春 OpenResty 开源项目创建者

Q&A