Nginx+Lua
阿里实战经验谈

阿里云-CDN事业部 卫越

2015.11

阿里云 aliyun.com

目录

第一部分

创始

- 创始人是当时在淘宝的王晓哲和章亦春
- 量子统计是淘宝第一个使用Nginx + Lua的应用
- OpenResty创立，章亦春将ngx_lua推向全球社区，受到社区推崇

尝试

- 使用Nginx + Lua代替Java
- 等同于Java中间件功能的Lua中间件

着陆

- 附身FastCGI的业务系统
- 附身接入系统
- 附身安全系统
- 放弃Java

第二部分

Lua的优势

- 内存开销小：VM < 100KB
- LuaJIT的运行效率与C、Java处于同一数量级

测试	Lua	LuaJIT	Java	PHP
fasta	7.02	0.8	0.42	27.96
nbody	58.6	1.34	0.96	143.42
spectral-norm	113.3	2.59	2.98	705.54
binary-trees	29.29	2.95	0.46	110.95
mandelbrot	59.71	1.8	1.05	219.55
fannkuchredux	193.31	5.4	2.66	639.50

- 原生支持协程：与非阻塞IO结合非常好
- [1] https://github.com/chaoslawful/shootout

阿里巴巴的需求

1. 快速开发，迅速完成需求迭代
2. 快速执行
3. 运行稳定

Tengine+Lua

- 在阿里巴巴，Tengine是Lua的搭档
  - Tengine是由阿里巴巴发起的Web服务器项目。
  - Tengine基于Nginx，高于Nginx。
  - Tengine的性能和稳定性已经得到了检验。
  - Tengine的最终目标是高效、稳定、安全、易用。

阿里云 aliyun.com

Tengine+Lua的适用场景

- 胶水功能：简单的HTTP头处理功能，各类业务逻辑
- 反向代理：HTTP、HTTPS接入层路由模块
- 数据处理层：与数据库有交互并有数据处理
- CPU计算型：图片处理（需要加入异步逻辑）
- 安全处理：安全规则嵌入，内容扫描

取数据计算

- 使用Tengine+Lua最典型的场景
- 简化Tengine子请求的复杂性和避免不确定性
- 核心是*ngx.location.capture*的方法
- capture的对象是Tengine的本地location
- 本地location可以使用proxy_pass或者fastcgi_pass获取其他服务器上的内容

取数据计算

```
location /fetch {

    fastcgi_pass X.X.X.X:P;

}

location /main {

    content_by_lua "res = ngx.location.capture ('/fetch');";

}
```

取数据计算

- 局限性:
  - 放弃了流式处理的优势
  - 内存消耗增加
  - 获取的响应大小有限制
  - 并发可能受到限制

处理响应body

- 一般使用body_filter_by_lua
- 有一个奇怪的现象，使用一个Lua本地变量得到arg[1]中的body内容，再对变量进行处理，比直接对arg[1]进行处理快得多。
- ngx_lua = 0.9.4，LuaJIT = 2.0。

插入处理逻辑

- 目标：在Tengine/Nginx的rewrite中插入复杂逻辑
- 背景：原生的rewrite功能有限
- 方法：有两种
  - rewrite_by_lua
  - set_by_lua

rewrite_by_lua

- 适于编写完整的rewrite逻辑
- 不适用于和已有的Tengine/Nginx的rewrite功能混用

rewrite_by_lua

```
server {
    rewrite_by_lua 'ngx.return 403';
    location /test1 {
        return 200;
    }
}
```

rewrite_by_lua

```
server {
    location /test1 {
        return 200;
        rewrite_by_lua 'ngx.return 403';
    }
}
```

set_by_lua

```
server {
    set_by_lua  $val 'return 1';
    location /test1 {
        if ($val = 1) { return 200; }
    }
}
```

第三部分

阿里云CDN动态配置

- 阿里CDN提供很多功能，比如视频切片，比如缓存过期时间等等
- 这些功能有的是C模块，有的是Lua模块
- 每个功能都有一个或多个控制入口，比如开启关闭功能，或者控制参数。

阿里云CDN动态配置

- 所有的控制入口的取值都对应到某个Tengine的变量
  - C模块：ngx_http_get_flushed_variable
  - Lua模块：ngx.var.VNAME
- 使用Lua更新配置，即对变量进行赋值

阿里云CDN动态配置

- 更新配置的时机
- 更新的来源
- 效果
  - 支持百万级的域名
  - 配置变更的时间<1分钟

第四部分

遇到的问题

- 单进程内存限制，某些场景下会超出限制而无法工作。
- 缺乏隔离性，不同模块的全局数据需要开发者自行解决命名冲突。
- ngx_lua提供的各种钩子指令不支持定义多次。
- 调试不方便。

Lua的竞争对手

- https://www.nginx.com/blog/launching-nginscript-and-looking-ahead/

阿里云 aliyun.com

Q&A