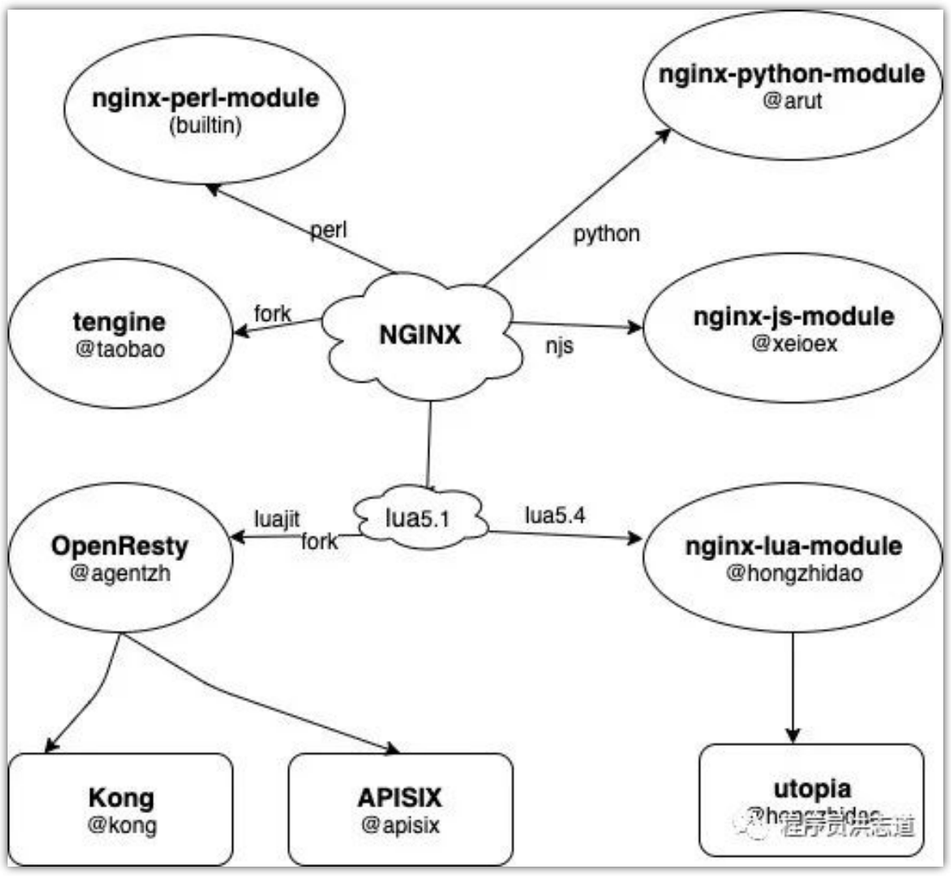
一文看透NGINX开发史

发表于2020-10-25 17:42 | 浏览 2k

我的职业生涯大部分时间都在跟NGINX打交道，有足够的经验分享整个NGINX开发史的演进。本文以事后诸葛的角度揭示怎么形成现在这个生态。



此图展示了现如今活跃在NGINX生态的重要开源模块和产品。

一切从C开始

NGINX是纯C实现的软件，源码质量很高。即使不从事NGINX的人也可以将它作为很好的学习软件。作者Igor很早就有支持脚本语言的意图。所以问题就变成了C如何跟脚本语言引擎的结合了。这些主流脚本语言python, php, v8(js), perl, lua都有C的API，考虑到两方面：轻量级和性能，perl成为了当时的NGINX首选。甚至他还实现了一个迷你的SSI模块，自定义的脚本功能。但是从现在看，个人觉得Lua是和C交互最好的语言，天生为C设计的脚本语言，足够轻量，足够快。@agentzh将Lua引入NGINX，怎么看都是正确和极具工程实用的选择。估计NGINX社区也没料到Lua模块能如此成功。

但是呢，Igor一直有脚本语言的情节，于是在2015年NJS诞生了，也是纯C的JS引擎。Lua天生能跟C交互，但不意味着它能天生跟服务器软件交互。这里最大的问题是语言虚拟机的实现。简单讲，一个请求会有个vm，非常轻量级，不用担心。对NGINX的每个请求，Lua的vm虽然做了些处理，但还是可以相互影响，如果你想做的话。但是Lua有GC，这也让这问题变的不是很严重。Igor的NJS是专门为NGINX量身定制的JS引擎，还考虑到JS有如此大的用户群体。NJS的每个请求的vm都是相互独立的，不会有任何影响，但是它没有GC。早期GC是列入NJS的计划的，但现在已经变的很遥远了，实现成本太大。这个在 HTTP请求也不是问题，NGINX对每个请求都是统一分配和销毁的。如果用户想在配置上增强NGINX，比如鉴权，NJS是个很好的选择，这也是它的设计初衷。而且NJS通过NGINX的subrequest能跟第三方交互，我们（我是NJS的开发者）最近还打算支持内置的HTTP库。所以想做应用的同学，NJS是个值得入手的选择。

模块化，鱼与熊掌

NGINX的模块化机制从第一个版本就有了，但是当时Igor并不是为了第三方考虑的，只是为了方便自己的开发。用NGINX的人大都会为它的模块化机制感到惊叹，谁都可以不用改NGINX源码，只需加入自己的模块，以满足自己的需求。Lua就是最好的例子。

这问题也让Igor很头大，为什么呢？大量的第三方模块，质量参差不齐，它们严重依赖NGINX的API。NGINX是20年前的软件，当时的服务器架构跟如今已经不可同日而语。软件需要进化，就要做重构，但是API不能轻易改。关注NGINX社区的人知道，Igor亲自设计了另一个跟NGINX不同的软件Unit，这软件不会再支持模块化了，这是他们的选择。

所以从短期看，NGINX的模块化让它快速建立了整个生态。从长期看，整个生态也束缚在它的架构上。NGINX最大的问题不具备热加载，这种现在主流软件里已经不是问题，反而在它这里变成很棘手的问题，好在很多以Lua为主的应用可以解决这个问题。

OpenResty

NGINX的C实现和模块化机制，让Lua的引入变得顺其自然。虽然也有其它的语言已经支持，但如今证明只有Lua玩转的最好，或者说OpenResty做的最成功。

严格来说，OpenResty里的Lua不能叫Lua。它是LuaJIT，以Lua5.1语法为主的另一个分支，而且看着没有跟进官方Lua的计划。Lua5.1以后的版本在语法上有了不少的改进，类似位运算，性能更是如此。对单纯使用Lua而言，非常推荐Lua5.4。

OpenResty让很多开发者大幅提升了开发生产力，并且在它上面衍生了不少开源软件，尤其在API方面，比如Kong和APISIX。很多公司也有内部的自研尝试。

看懂设计

不管你是多么熟悉NGINX源码的开发者，还是只是想用它作为应用服务器。脚本语言都是其中的选择之一。从两方面看NGINX：通用功能和业务功能。

通用功能：将它扔进NGINX里，如果你能做模块开发，这点尤为重要。它意味着你将享受未来稳定和维护的红利。举个实际例子，我们在开发NJS里，有个querystring的功能，开发需要一定的成本，用JS语言来写会简单很多，但是我们依然选择将它放在JS引擎里。个人觉得类似Lua里的http request这种库，如果原生Lua模块里支持是再好不过了。对NJS，我们会选择放在js模块里，用户可以直接使用，而不用再引入任何库。还有不少的能用功能，比如常用工具函数md5，sha2之类的。

业务功能：这个不用多说，维护好业务模块就行了。

从整体看，Lua既封装了NGINX的HTTP请求，也提供了独立于请求的功能，比如timer（定时器）和cosocket（跟第三方交互的基础机制）。不管什么模块，也都是基于这两方面进行设计的。我一直推进NJS在这方面的能力，因为目前NJS只能处理请求，但即将引入内置的HTTP库，完全独立于请求的。重复一遍：请求和非请求。

我自己的模块，也一直遵守这样的设计，https://github.com/hongzhidao/nginx-lua-module

关于作者

洪志道
这家伙很懒还未留下介绍～

11 文章 | 0 问答 | 52 粉丝

+ 关注 Ta | Ta 的主页

相关文章

后分布式时代: 多数派读写的'少数派'…
本文链接:https://blog.openacid.com/algo…
点赞 1 浏览 924

nginx命令大全
nginx-sreopen#重启Nginxnginx-sreload…
点赞 1 浏览 13.6k

奇怪的proxy_redirect指令问题
背景：浏览器访问方向代理Nginx，到…
点赞 0 浏览 648

给开发者的建议

如果想学习NGINX，参考这篇 如何高效的学习NGINX

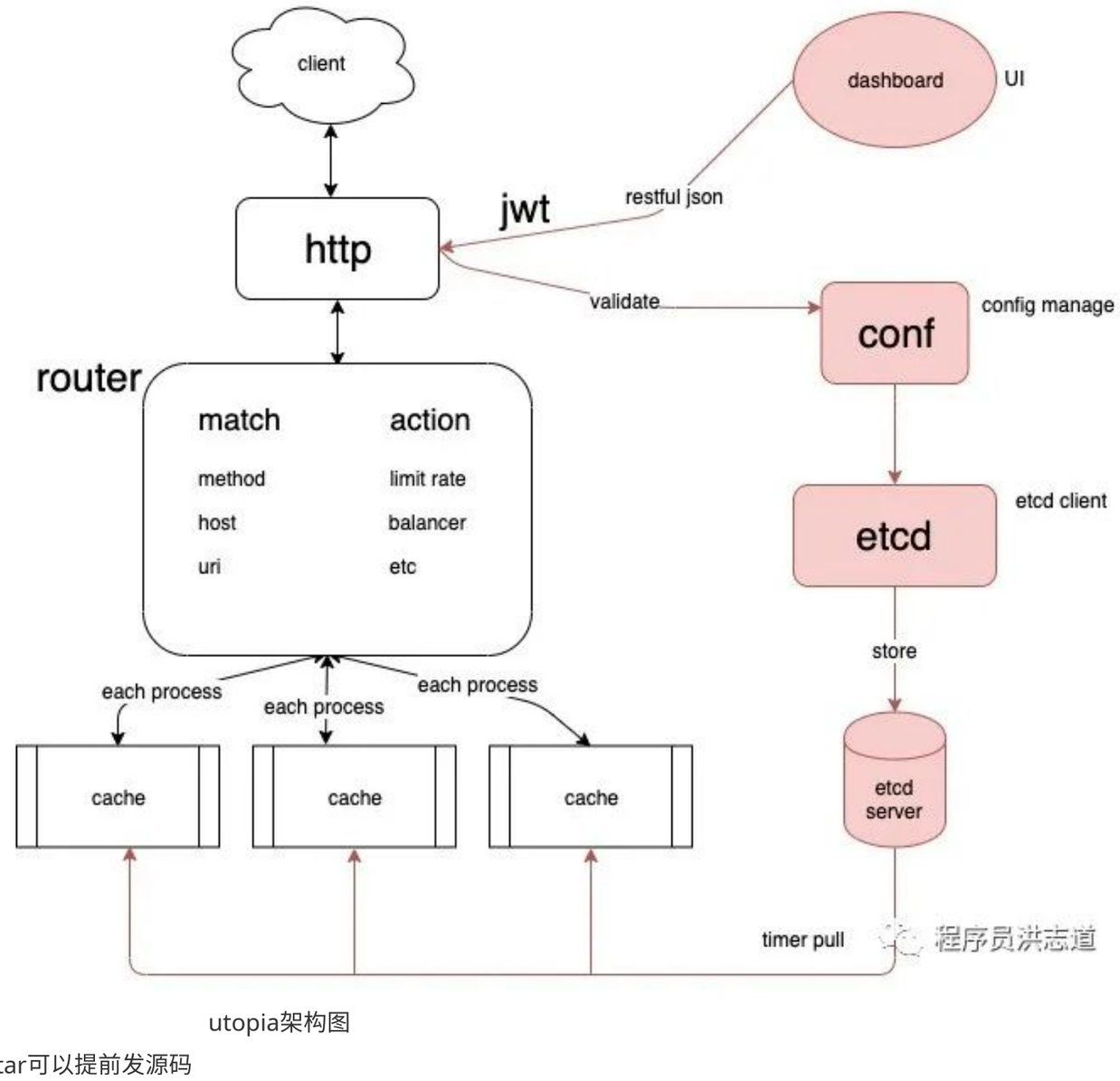
对程序员，长期保持自己的竞争力，自研能力是很重要的技能之一。

用Lua作为开发的，不妨自己写个框架，如今API框架是非常好的练手对象。

来乌邦托的世界

基于nginx-lua-module的API框架，一千行左右。

```
find. -type f -name '*.lua' -exec wc -l {} +
109./balancer.lua
311./conf.lua
78./etcd.lua
286./http.lua
160./jwt.lua
214./router.lua
1158total
```



utopia架构图

即将开源，Star可以提前发源码

[utopia] https://github.com/hongzhidao/utopia

感谢好友xp取寓，分布式研究小院微信公众号：gh_d5b90ac0d668

已修改于2023-03-09 02:04

洪志道

暂无个人介绍

+ 关注

写下您的评论

发表评论

点赞

全部评论（0）

按点赞数排序 | 按时间排序

评论

收藏

NGINX Part of F5

开放 · 包容 · 沟通 · 贡献

分享

举报

NGINX开源社区

- 关于我们
- 联系我们
- 社区准则
- 社区公告
- 服务条款
- 隐私政策

官方站点

- NGINX 中文官网
- NGINX 中文社区
- NGINX 英文官网
- NGINX 英文社区
- NGINX 官方文档

NGINX 的 GitHub 仓库

- NGINX 开源版
- NGINX Unit
- NGINX JavaScript
- NGINX 现代应用参考架构 (MARA)
- NGINX Ingress Controller
- NGINX Kubernetes Gateway
- 更多 NGINX 开源项目

参与贡献

- 新手指南
- 社区排行
- 任务中心
- 最新活动
- 发表文章
- 问答互助
- 热门标签
- 提交反馈

更多资源

- 软件下载
- 官方文档
- 三方插件
- 官方博客
- 技术支持
- 电子书
- 公开课

社交媒体

- 微信公众号
- B站主页
- 知乎主页
- 微博主页