**Nginx与Lua**

发表于[**2012-08-31**](http://huoding.com/2012/08/31/156)

火云邪神语录：天下武功，无坚不破，唯快不破！Nginx的看家本领就是速度，Lua的拿手好戏亦是速度，这两者的结合在速度上无疑有基因上的优势。

最先将Nginx，Lua组合到一起的是[OpenResty](http://openresty.org/)，它有一个[ngx\_lua](http://wiki.nginx.org/HttpLuaModule)模块，将Lua嵌入到了Nginx里面；随后[Tengine](http://tengine.taobao.org/)也包含了ngx\_lua模块。至于二者的区别：OpenResty是Nginx的Bundle；而Tengine则是Nginx的Fork。值得一提的是，OpenResty和Tengine均是国人自己创建的项目，前者主要由[春哥](http://weibo.com/n/agentzh)和[晓哲](http://weibo.com/n/chaoslawful)开发，后者主要由淘宝打理。

至于OpenResty和Tengine孰优孰劣，留给大家自己判断，如下资料可供参考：

* [ngx\_openresty: an Nginx ecosystem glued by Lua](http://agentzh.org/misc/slides/ngx-openresty-ecosystem.pdf)
* [淘宝网Nginx应用、定制与开发实战](http://tengine.taobao.org/download/taobao_nginx_2012_06.pdf)

推荐看看春哥在[Tech-Club](http://www.tech-club.org/)上关于『[由Lua粘合的Nginx生态环境](http://blog.zoomquiet.org/pyblosxom/oss/openresty-intro-2012-03-06-01-13.html)』的演讲实录，有料！

**安装**

需要最新版的[Nginx](http://nginx.org/en/download.html)，[LuaJIT](http://luajit.org/download.html)，[ngx\_devel\_kit](https://github.com/simpl/ngx_devel_kit/tags)，[ngx\_lua](https://github.com/chaoslawful/lua-nginx-module/tags)等安装文件。

安装Lua或者LuaJIT都是可以的，但是出于效率的考虑，推荐安装LuaJIT。

shell> wget http://luajit.org/download/LuaJIT-<VERSION>.tar.gz

shell> tar zxvf LuaJIT-<VERSION>.tar.gz

shell> cd LuaJIT-<VERSION>

shell> make

shell> make install

因为安装在缺省路径，所以LuaJIT对应的lib，include均在/usr/local目录里。

shell> export LUAJIT\_LIB=/usr/local/lib

shell> export LUAJIT\_INC=/usr/local/include/luajit-<VERSION>

下面就可以编译Nginx了：

shell> wget http://nginx.org/download/nginx-<VERSION>.tar.gz

shell> tar zxvf nginx-<VERSION>.tar.gz

shell> cd nginx-<VERSION>

shell> ./configure

--add-module=/path/to/ngx\_lua \

--add-module=/path/to/ngx\_devel\_kit

shell> make

shell> make install

试着启动一下Nginx看看，如果你运气不好的话，可能会遇到如下错误：

*cannot open shared object file: No such file or directory*

这是神马情况？可以用ldd命令来看看：

shell> ldd /path/to/nginx

libluajit-<VERSION>.so => not found

此类问题通常使用ldconfig命令就能解决：

shell> echo "/usr/local/lib" > /etc/ld.so.conf.d/usr\_local\_lib.conf

shell> ldconfig

再试着启动Nginx看看，应该就OK了。

**应用**

我们先用一个简单的程序来暖暖场：把下面的代码加入到Nginx的配置文件nginx.conf，并重启Nginx，然后浏览，就能看到效果了。

location /lua {

set $test "hello, world.";

content\_by\_lua '

ngx.header.content\_type = "text/plain";

ngx.say(ngx.var.test);

';

}

在深入学习ngx\_lua之前，建议大家仔细阅读一遍春哥写的[Nginx教程](https://github.com/agentzh/nginx-tutorials)。

这里我就说关键的：Nginx配置文件所使用的语言本质上是『声明性的』，而非『过程性的』。Nginx处理请求的时候，指令的执行并不是由定义指令时的物理顺序来决定的，而是取决于指令所属的阶段，Nginx常用的阶段按先后顺序有：rewrite阶段，access阶段，content阶段等等。演示代码中的set指令属于rewrite阶段，content\_by\_lua指令属于content阶段，如果试着把两条指令的顺序交换一下，会发现程序依然能够正常运行。

下面我们尝试结合Redis写个更实战一点的例子。

首先，我们需要创建一个Redis配置文件config.json，内容如下：

{

"host": "<HOST>",

"port": "<PORT>"

}

然后，我们创建一个解析配置文件的脚本init.lua，其中用到了[Lua CJSON](http://www.kyne.com.au/~mark/software/lua-cjson.php)模块：

local cjson = require "cjson";

local config = ngx.shared.config;

local file = io.open("config.json", "r");

local content = cjson.decode(file:read("\*all"));

file:close();

for name, value in pairs(content) do

config:set(name, value);

end

说明：代码里用到了共享内存，这样就不必每次请求都解析一遍配置文件了。

接着，我们创建一个渲染内容的脚本content.lua，用到了[Resty Redis](https://github.com/agentzh/lua-resty-redis)模块：

ngx.header.content\_type = "text/plain";

local redis = require "resty.redis";

local config = ngx.shared.config;

local instance = redis:new();

local host = config:get("host");

local port = config:get("port");

local ok, err = instance:connect(host, port);

if not ok then

ngx.log(ngx.ERR, err);

ngx.exit(ngx.HTTP\_SERVICE\_UNAVAILABLE);

end

instance:set("name", "laowang");

local name = instance:get("name")

instance:close();

ngx.say("name: ", name);

说明：建议把Resty Redis模块放到vendor目录下，稍后在Nginx中统一设置。

最后，我们需要在Nginx配置文件里设置一下：

lua\_shared\_dict config 1m;

lua\_package\_path "/path/to/vendor/?.lua;;";

init\_by\_lua\_file /path/to/init.lua;

server {

lua\_code\_cache off;

location /lua {

content\_by\_lua\_file /path/to/content.lua;

}

...

}

说明：为了方便调试，我关闭了lua\_code\_cache，如果是生产环境，应该开启它。

另外，安装CJSON的时候，需要注意Makefile文件里头文件的路径，缺省是：

PREFIX = /usr/local

LUA\_INCLUDE\_DIR = $(PREFIX)/include

如果安装的是LuaJIT的话，最好把头文件拷贝到相应目录：

cp /usr/local/include/luajit-<VERSION>/\* /usr/local/include/

…

我最近参与的一个项目，提供了一些用于Web轮询的接口，都是用Nginx+Lua实现的，虽然总共只有十几台服务器，但是每天可以提供几十亿次的请求量，贼拉拉的强。

最后，让我引用某位屌丝的语录做结束语吧：Lua，未婚男性程序员的最爱。