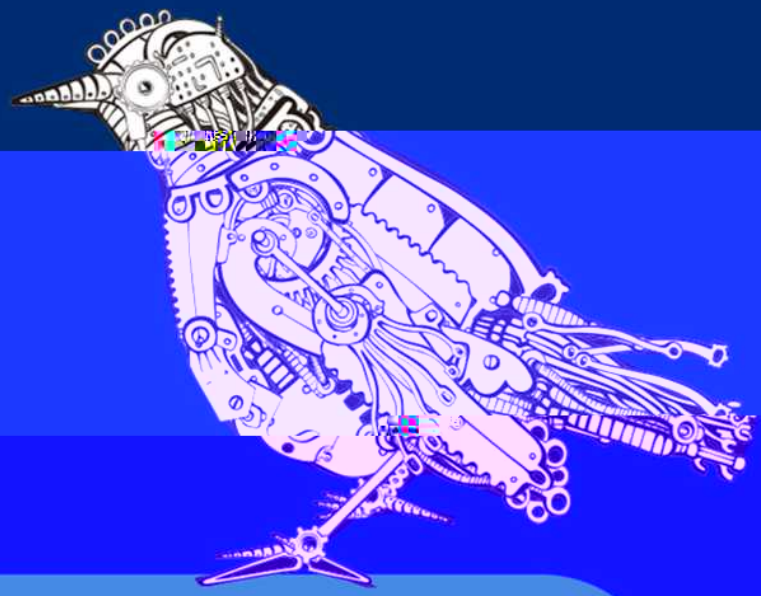
本书深入浅出地介绍了Nginx+Lua在实战场景中的各种使用技巧和方法，涉及Nginx配置、常用模块、缓存系统、日志分析、静态容灾、反向代理、爬虫、性能分析与优化等众多方面，掌握这些知识有助于提升你所开发的服务的性能。

Broadview
www.broadview.com.cn



Nginx 实战

基于Lua语言的配置、开发与架构详解

王力 汤永全 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

Nginx Nginx

Nginx+Lua Lua Lua

Nginx

Nginx

图书在版编目（CIP）数据

Nginx Lua . — 2019.3
ISBN 978-7-121-35460-1

. N… . … … . . TP368.5

CIP 2018 254599

173 100036
787 × 980 1/16 21.5 465
2019 3 1
2019 3 1
2500 79.00

010 88254888 88258888
zlts@phei.com.cn dbqq@phei.com.cn
010 51260888-819 faq@phei.com.cn

Nginx 2004

Nginx

800 800

800

Nginx

Nginx

Nginx

Nginx

Nginx

Nginx

C C++ Nginx

C C++

Nginx

OpenResty Nginx+Lua

Lua Nginx

Nginx

Nginx

OpenResty

API

Nginx+Lua

Nginx

Nginx

Nginx

Nginx

Nginx

Nginx

OpenResty

Nginx

Nginx

Nginx+Lua

“ ”

Nginx

18 1~5 Nginx
Nginx 6~10 Nginx+Lua Nginx+Lua
11~18 Nginx+Lua

800

leehomewl@gmail.com

2018 12

---

www.broadview.com.cn

- 提交勘误：
- 交流互动：

http://www.broadview.com.cn/35460

1

Nginx HTTP HTTP HyperText Transfer Protocol TCP Transmission Control Protocol HTTP Web Nginx Nginx

Nginx

Nginx 1.12.2 CentOS 6

HTTP Nginx HTTP HTTP HTTP HTTP 3

- URL Uniform Resoure Locator HTTP GET HEAD POST HTTP 1.0 HTTP 1.1 PUT DELETE CONNECT OPTIONS TRACE PATCH HTTP 1.1
- key/value key value key Cookie User_Agent Accept-Encoding

- POST

HTTP

HTTP HTTP 3

- HTTP HTTP 1-1

表 1-1 HTTP 状态码说明

状 态 码	作 用
1XX	
2XX	
3XX	
4XX	401 404 URL
5XX	500 504

- key Content-Type Content-Encoding
-

```
HTTP/1.1 200 OK        # HTTP
Server: nginx/1.12.2                                    #
Date: Sun, 01 Jul 2018 03:10:11 GMT                     #
Content-Type: application/octet-stream                  #
Transfer-Encoding: chunked                              #
Connection: keep-alive                                  #
{"test":"Nginx","hello":"world!"}                       #
```

Nginx

return echo

Nginx

Ubuntu　　sudo apt-get install nginx

CentOS　　Nginx　　lib

```
    # yum -y install wget  gcc gcc-c++ autoconf automake make  zlib zlib-
devel  pcre-devel pcre
```

```
    # wget https://nginx.org/download/nginx-1.12.2.tar.gz
```

```
    # cd nginx-1.12.2
    # ./configure
    # make && make install
```

Nginx　　Nginx

./configure　　./configure

1-2

表 1-2　./configure 命令的常见参数说明

参　数	说　明
--prefix=PATH	Nginx　　/usr/local/nginx
--conf-path=PATH	/usr/local/ nginx/conf/nginx.conf Nginx　　-c
--with-threads	Nginx　　Nginx
--with-file-aio	Linux 2.6.22　　I/O
--with-http_gzip_static_module	ngx_http_gzip_module
--with-http_realip_module	IP
--with-http_ssl_module	HTTPS
--without-http_gzip_module	ngx_http_gzip_module --without-http_[　　]

./configure --help

Nginx

conf　　nginx.conf　　/usr/local/nginx/conf/nginx.conf

vim

HTTPS OpenSSL lib Nginx HTTPS

```
# ./configure  --prefix=/usr/local/nginx --with-http_ssl_module
# make && make install
```

OpenSSL 2014 CentOS OpenSSL OpenSSL Nginx

```
# ./configure  --prefix=/usr/local/nginx --with-http_ssl_module \
  —with-openssl=/path/openssl-1.0.2o
# make && make install
```

lib PCRE lib --with-pcre=DIR zlib lib --with-zlib=DIR

Nginx Nginx --add-module=PATH PATH

Nginx Wiki Nginx

Nginx HTTP HTTP Nginx

2

Nginx

Nginx

http server location URI Uniform Resource Identifier URI URL

http server location

location expires 1m expires location location expires expires http 1d

```
http   {
    expires 1d;  #
    #      server
    server  {
        location / { expires 1m; }
    }
}
```

2.1 Nginx

2.2.1 main 配置

http 2.1 Main 1 Nginx

```
user  nobody;
worker_processes  1;
error_log  /var/log/error_log
worker_rlimit_nofile 1024;
events {
    worker_connections  1024;
    use epoll;
}
```

2.2.2　与客户端有关的配置

http　　2.1　　Main 1　server

Main 2

2-1

表 2-1　客户端配置常用的指令

指　令	说　明
client_body_buffer_size	32　x86-64 8KB　64　16KB
client_body_temp_path	3
client_body_timeout	HTTP　408
client_header_buffer_size	1KB
client_max_body_size	1MB
client_header_timeout	
etag	on　ETag
large_client_header_buffers	
keepalive_timeout	HTTP
send_timeout	
server_names_hash_bucket_size	server_names　Nginx
server_names_hash_max_size	server_names
server_tokens	Nginx
tcp_nodelay	TCP_NODELAY
tcp_nopush	sendfile

Main　　2.1

Main 1　Main 2　Main 3　client_body_timeout　http　server　location

server_names_hash_bucket_size　http

2-1　　Nginx

2.2.3　server 块

server　　Host　server_name

server

```
server {
        server_name testnginx.com www.testnginx.com;
}
```

server_name server

1

2 * *.testnginx.com

3 * testnginx.*

4 1

5 server_name default_server

2.2.4 location 块

location server URL

URL location 2-2 URL location

表 2-2 URL 在 location 块中的匹配规则说明

配置格式	作 用
location = /uri	=
location ^~ /uri	^~ URL
location ~	~
location ~*	~*
location /uri	
location /	
location @	

2-2

“ = ” “ = ” “ ^~ ” “ ~ ”

“ ~ ” “ /uri ”

“ / ” “ @ ”

URL

建议： Debug

location

location

```
location /a {
        location /a {
[ configuration A]
        }
        location /a/b {
[ configuration AB]
        }
}
```

location 3

- internal location Nginx rewrite error_page HTTP
- limit_except location HTTP GET
- alias

```
location /a/ {
    alias /c/x/a/;
}
```

/a/test.json location /c/x/a/test.json

include Nginx include include location server

include .conf Nginx

```
include /usr/local/nginx/conf/vhost/*.conf;
```

Nginx

Nginx main server location include

Nginx

2.4.1 常见配置注解

```
user  www www;                          #       Nginx
worker_processes  2;                    #Nginx
worker_cpu_affinity auto;               #    Nginx      CPU
error_log  /var/log/error_log  info; #                            error
worker_rlimit_nofile 65535;             #    worker
pid /var/run/nginx.pid;                 #
worker_priority -10;                    #  Linux
worker_shutdown_timeout  30;  #    30s   Nginx    “    ”

events {
    #                       Nginx           =                       ×
    worker_connections  10000;
    #epoll    Linux 2.6                        I/O
    #       FreeBSD        kqueue
    use epoll;
}

http {
    include       conf/mime.types;            #
    default_type  application/octet-stream;  #
    log_format main     '$remote_addr - $remote_user [$time_local]'
                         '"$request" $status $bytes_sent'
                         '"$http_referer" "$http_user_agent" '
                         ' "$http_cookie" ';  #
    client_header_buffer_size    1k;   #                    buffer
    large_client_header_buffers  4 4k;#
    server_names_hash_bucket_size 128; #    server_names
                                       #
    client_header_buffer_size 32k;     #
    gzip on;                           #    gzip
    gzip _comp_level 6;                #
    gzip _min_length  1100;            #
    gzip _buffers     4 8k;            #                                gzip
                                       #                4 8k
                                       #             8KB        4
    gzip _types   text/plain text/css; #    MIME
    output_buffers   2 32k;            #
                                       #     2 32k
                                       #    32KB        2
```

```
    sendfile         on;               #    sendfile()
    tcp_nopush       on;               #                    sendfile
    tcp_nodelay      on;               #                    sendfile
    keepalive_timeout 90s;             #
    upstream backend {}                #upstream    weight
    server 192.168.1.12:8081 weight=3; #
    server 192.168.1.13:8081 weight=2;

    server {
        listen       80;                                  #HTTP
        server_name  your.example.com;                    #
        access_log   /var/log/nginx.access_log  main;    #
        charset  koi8-r;                                  #
        location / {
            proxy_pass         http://backend ;
            proxy_redirect     off;
            proxy_set_header   Host             $host;
            proxy_set_header   X-Real-Ip        $remote_addr;
            #            X-Forwarded-For             IP
            proxy_set_header    X-Forwarded-For  $proxy_add_x_forwarded_for;
    }
    error_page  404  /404.html;  #                   404
        location /404.html {
            root  /spool/www;
        }
                          #     jpg  jpeg  gif      URL
        location ~* \.(jpg|jpeg|gif)$ {
            root         /spool/www;
            expires 30d;   #                      CDN    CDN
        }
    }

}
```

2.4.2　常见配置实战技巧

Nginx

Nginx

- user

  nobody　　nobody

root www Linux SSH Secure Shell

- worker_processes

worker CPU Central Processing Unit

worker_processes auto Nginx 1.2.5

CPU

Nginx

- worker_cpu_affinity auto

Nginx 1.9.10 CPU CPU

Nginx

- error_log & access_log

error_log error

info Linux

I/O Linux

I/O

message Nginx

- worker_priority

Nginx Linux Nginx

-10

- gzip_comp_level

9 CPU

7 9

5~7

- upstream

proxy_next_upstream

- error_page

  error_page

- location & root

  root　　Nginx 1.7.11　　Nginx

  I/O　　/　　proxy_cache

Nginx　　Nginx　　Wiki

Nginx　　HTTP　TCP

注意：Nginx 1.9　　TCP　　Nginx
HTTP　　TCP

2.5.1　常见内置变量

2-3

表 2-3　常见内置变量的说明

变　量　名	说　　明
$arg_name	URL　　name
$args	URL
$binary_remote_addr	
$body_bytes_sent	
$bytes_sent	
$document_uri	$uri
$hostname	Nginx
$http_referer	
$http_user_agent	

续表

变 量 名	说 明
$remote_addr	IP
$remote_port	
$remote_user	Auth Basic
$request_filename	root alias URI
$request_time	Nginx
$request_uri	URI
$request	URL HTTP
$request_length	
$server_name	server_name
$server_port	
$server_addr	IP
$request_method	POST GET
$scheme	HTTP HTTPS
$sent_http_name	name name
$realip_remote_addr	在 real_ip
$server_protocol	HTTP/1.0 HTTP/1.1
$uri	URI URI
$nginx_version	Nginx
$pid	worker PID
$pipe	HTTP pipelined pipe "p" "."
$connection_requests	
$cookie_name	name Cookie Cookie
$status	HTTP
$msec	
$time_local	
$upstream_addr	IP
$upstream_port	
$upstream_response_time	
$upstream_status	HTTP
$geoip_city	geoip

注意： Nginx Nginx
Nginx Wiki

2.5.2　常见内置变量实战技巧

Nginx

- $arg_name

```
location     {
     if ($arg_at= '5') {
        proxy_pass http://b;
     }
     proxy_pass http://a;
}
```

http://a　　URL　　at=5　　http://b

if　location

- $body_bytes_sent　$bytes_sent

HTTP

proxy_buffer　　proxy_buffer　　error.log

```
2017/11/24 11:49:06 [error] 8376#0: *28071 upstream sent too big header
while reading response header from upstream,
```

```
proxy_buffers   4 256k;
proxy_buffer_size   128k;
proxy_busy_buffers_size   256k;
```

- $realip_remote_addr

Nginx 1.9.7　　ngx_http_realip_module　　IP

CDN　　IP

- $request_time　$upstream_response_time

$upstream_response_time　　Nginx　　upstream　　Nginx

$request_time

$upstream_response_time

- $uri $request_uri

$uri URL
$args

$request_uri URL
$request_uri $uri
Nginx $request_uri $uri

- $scheme

HTTPS HTTP HTTPS
https:// HTTPS
HTTP HTTPS $scheme
HTTP HTTPS

```
if ($scheme = 'http') {
    rewrite ^/(.*)$     https://$host/$1 redirect;
}
```

Nginx Nginx
Nginx

3

Nginx

Nginx

注意： Nginx Nginx

Context Nginx

Context

3-1 的

3-1

3-1

```
if ($arg_id) {
    proxy_set_header X-id '1';    #
}
```

```
    proxy_pass http://backend
```

Nginx

```
    nginx: [emerg] "proxy_set_header" directive is not allowed here in
/usr/local/nginx/conf/test.ngx.conf:69
```

Nginx Wiki

```
    Syntax: proxy_set_header field value;
    Default:
    proxy_set_header Host $proxy_host;
    proxy_set_header Connection close;
    Context: http, server, location
```

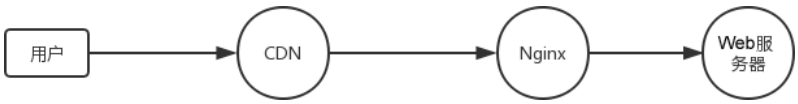
proxy_set_header Context http server location 3
if

Nginx IP $remote_addr
IP

3.2.1 获取用户的真实 IP 地址

3-2 CDN
Nginx CDN Content Delivery Network
Nginx Nginx IP CDN IP

IP realip



3-2 CDN

realip --with-http_realip_module Nginx http

```
    set_real_ip_from  CDN_IP;
    real_ip_header    X-Forwarded-For;
    real_ip_recursive on;
```

- set_real_ip_from　　IP　　real_ip_header　　IP
- real_ip_header　　IP　　IP　　Nginx　　IP　　IP　　$remote_addr　　X-Forwarded-For
- real_ip_recursive　　on　　realip_module　　real_ip_header　　IP　　IP　　off　　IP

IP　　CDN　　IP　　CDN　　$realip_remote_addr　　2.5.2

3.2.2　防止 IP 地址伪造

X-Forwarded-For　　IP　　Nginx　　CDN

- CDN　　X_Cdn_Ip
- CDN　　CDN　　IP　　CDN　　3-3　　CDN　　IP

3-3　CDN　　IP

注意：　　CDN　　Nginx　　Nginx　　IP　　proxy_set_header X-Real-IP $remote_addr

3.2.3　后端服务器对 IP 地址的需求

IP　　IP

IP IP
IP IP

HTTP
Nginx

3.3.1 限制 IP 地址的访问

IP 访问进行限制 allow deny allow deny
3-1

表 3-1 allow 和 deny 指令的说明

指 令	作 用	配置环境
allow	IP IP	http server location limit_except
deny	IP IP	http server location limit_except

allow deny 3-4 deny
allow

3-4 deny

location IP

```
location / {
    deny  192.168.1.1;        #     192.168.1.1
    allow 192.168.1.0/24;     #     192.168.1.0/24
    allow 17.1.1.2;           #     17.1.1.2
    deny  all;                #   allow   IP            IP
}
```

IP 限制 IP

3.3.2　auth 身份验证

allow　deny　IP

auth_basic

IP

```
server {
    listen       80;
    server_name  localhost;
    location / {
       auth_basic           " Nginx Basic ";
       auth_basic_user_file   conf/htpasswd;  #
    }
}
```

指令：auth_basic

auth_basic string/off;

auth_basic off;

http　server　location　limit_except

string　off

off　string

指令：auth_basic_user_file

auth_basic_user_file file;

http　server　location　limit_except

file　testuser 1XlKs2P mC$xfxImYPQPMTloK5J7dar.1

htpasswd　OpenSSL　OpenSSL　HTTPS

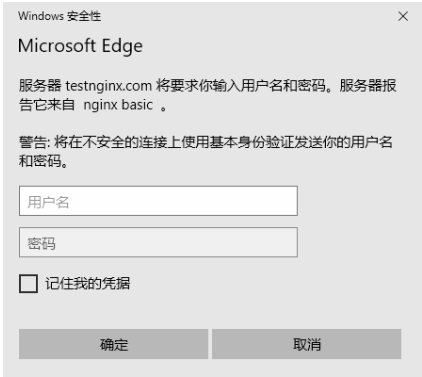
testuser　Pass123

```
# printf "testuser:$(openssl passwd -1 Pass123)\n" >> .htpasswd
```

testuser:1DRCZTLTx$dRBMISe3SBnw/VZdBfhCg1

file Nginx IP

3-5



3-5

注意： Nginx

403

3.3.3 利用 LDAP 服务加强安全

auth_basic

LDAP Lightweight Directory Access Protocol

LDAP LDAP

Nginx

Nginx LDAP

```
# yum  install openldap-devel -y
```

LDAP Nginx

```
# git clone https://github.com/kvspb/nginx-auth-ldap      #
# ./configure --add-module=path_to_http_auth_ldap_module  #
# make
# make install
```

Nginx http

```
        ldap_server testldap {
          URL  ldap://192.168.1.100:3268/DC=testnginx,DC=com?sAMAccountName?
```

```
sub?(objectClass=testuser);  #                                    LDAP
            binddn "TEST\\LDAPUSER";
            binddn_passwd LDAPPASSWORD;
            group_attribute uniquemember;
            group_attribute_is_dn on;
            require valid_user;
     }
```

server

```
    server {
        listen       80;
        server_name  testnginx.com;
        auth_ldap "Forbidden";
        auth_ldap_servers testldap;
        location / {
            root   html;
        }
    }
```

LDAP

3.3.4　satisfy 二选一的访问限制功能

satisfy　　satisfy

```
    satisfy any;
    auth_ldap "Forbidden";
    auth_ldap_servers testldap;

    allow 192.168.0.0/16;
    allow 10.10.10.10/32;
```

IP　　192.168.0.0/16　　10.10.10.10/32

LDAP　　IP　　LDAP

3.4.1 proxy_pass 请求代理规则

proxy_pass URL;

location if in location limit_except

IP HTTP/HTTPS

URI /test 127.0.0.1 81 HTTP

```
location = /test {
    proxy_pass http://127.0.0.1:81;
}
```

URL

```
location /test/v1/ {
    #     URL            /test/v1/          /abc/
    #    /test/v1/xxx?a=1                 /abc/xxx?a=1
    proxy_pass http://127.0.0.1:81/abc/;
}
location /aaa/ {
    #     URL                  /test/v1/                  "/"
    #                           /abc
    proxy_pass http://127.0.0.1:81/;
}
location /abc {
    #                           URL
    proxy_pass http://127.0.0.1:81;
}
```

注意： location URI proxy_pass URI proxy_pass proxy_pass http://127.0.0.1:81/abc/

3.4.2 减少后端服务器的网络开销

URL

URL

off

- proxy_pass_request_body HTTP
  http server location
- proxy_pass_request_headers HTTP

http　server　location

3.4.3　控制请求头和请求体

3-2

表 3-2　请求头和请求体的控制指令

指令名称	作　　用	支持配置的环境
proxy_hide_header	Cache-Control proxy_hide_header Cache-Control　"Date" "Server" "X-Pad" "X-Accel-..."	http　server　location
proxy_pass_header	proxy_hide_header Cache-Control	http　server　location
proxy_set_header	proxy_set_header HOST 'www.abc.co'	http　server　location
proxy_set_body	proxy_set_body 'b=123xxx'	http　server　location

注意：　　proxy_set_header
　　proxy_set_header

```
server {
    listen 80;
    proxy_set_header Host $host;
    proxy_set_header AB 'ab';

    location /abc {
    #     location          proxy_set_header              server
    # proxy_set_header                              A
    #       server                AB
    proxy_set_header A  'acv';
    proxy_pass http://127.0.0.1:81;
}
```

　　A　AB

```
server {
    listen 80;
    proxy_set_header Host $host;
    proxy_set_header AB 'ab';
    location /abc {
       #
       proxy_set_header Host $host;
       proxy_set_header AB 'ab';
       proxy_set_header A 'acv;'
       proxy_pass http://127.0.0.1:81;
    }
```

3.4.4 控制请求和后端服务器的交互时间

3-3

表 3-3 控制请求和后端服务器交互时间的指令

指令名称	作 用	支持的配置环境
proxy_connect_timeout	60s proxy_connect_timeout 5s;	http server location
proxy_read_timeout	60s proxy_read_timeout 10s;	http server location
proxy_send_timeout	60s proxy_send_timeout 10s;	http server location

60s

proxy_next_upstream*

proxy_pass

ngx_http_upstream_module

3.5.1　代理多台服务器

```
#        HTTP
upstream test_servers {
  #   server     HTTP
  server 127.0.0.1:81  max_fails=5 fail_timeout=10s weight=10;
  server 127.0.0.1:82  max_fails=5 fail_timeout=10s weight=5;
  server test.123.com  backup;
  server 127.0.0.1:82  down;

}
server {
    listen 80;
    location / {
       #                  upstream      HTTP
       proxy_pass http://test_servers;
    }
}
```

指令：upstream

upstream name { ... }

http

HTTP　　TCP　UNIX

upstream　　TCP　UNIX

指令：server

server address [parameters];

upstream

IP

server　　3-4

表 3-4　server 指令参数说明

参　数	作　用
weight	1 30　　1　　20　　2　　10

续表

参　数	作　用
max_fails	1　0
fail_timeout	fail_timeout =10s　10s　max_fails　10s　server 10s
down	
backup	upstream　upstream backup

3.5.2　故障转移

proxy_next_upstream　fastcgi_next_upstream　uwsgi_next_upstream　scgi_next_upstream　memcached_next_upstream　grpc_next_upstream

proxy_next_upstream

指令：proxy_next_upstream

proxy_next_upstream error | timeout | invalid_header | http_500 | http_502 | http_503 | http_504 | http_403 | http_404 | http_429 | non_idempotent | off ...;

proxy_next_upstream error timeout;

http　server　location

Nginx　HTTP　proxy_next_upstream

Nginx

max_fails　fail_timeout

off

指令：proxy_next_upstream_tries

proxy_next_upstream_tries number;

proxy_next_upstream_tries 0;

http　server　location

指令：proxy_next_upstream_timeout

proxy_next_upstream_timeout time;

proxy_next_upstream_timeout 0;

http　server　location

0　　proxy_read_timeout proxy_send_timeout　proxy_connect_timeout

注意：

proxy_next_ upstream_tries　　proxy_next_upstream

3.5.3　负载均衡

Nginx　　upstream　　3-5

表 3-5　负载均衡指令

指　令	用　途
hash	key　　key　　$request_uri hash　$user_agent　key
ip_hash	IP　　IP down　　IP　hash
least_conn	round-robin
sticky	Cookie　　Cookie “　” “　”

3.5.4　通过 hash 分片提升缓存命中率

HTTP　　Nginx proxy_cache　varnish　squid　　hash URL

URL URL

```
upstream test_servers {
  hash $request_uri;
  server 127.0.0.1:81  max_fails=5 fail_timeout=10s weight=10;
  server 127.0.0.1:82  max_fails=5 fail_timeout=10s weight=5;
}
```

注意：

- hash
- max_fails proxy_next_

upstream HTTP

Nginx Nginx

3.5.5 利用长连接提升性能

Nginx upstream

```
keepalive_requests 1000;
keepalive_timeout 60;

upstream test_servers {
  server 127.0.0.1:81  max_fails=5 fail_timeout=10s weight=10;
  server 127.0.0.1:82  max_fails=5 fail_timeout=10s weight=5;
  keepalive 100;
}
server {
    listen 80;
    proxy_set_header Host  $Host;
    proxy_set_header x-forwarded-for $remote_addr;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_http_version 1.1;
    proxy_set_header Connection "";
```

```
        location / {
            proxy_pass http://test_servers;
        }
    }
```

3-6

表 3-6　长连接配置指令说明

指　令	作　用	支持配置的环境
keepalive_requests		http server location
keepalive_timeout	keep-alive	http server location
keepalive	worker	upstream
proxy_http_version 1.1	HTTP HTTP 1.1	http server location
proxy_set_header Connection ""	Connection	http server location

注意：

QPS Query Per Second Nginx “

” QPS 0

QPS 0 timewait

3.5.6　利用 resolver 加速对内部域名的访问

proxy_pass IP

```
proxy_pass http:// test2.zhe800.com:82
```

DNS Domain Name System

DNS DNS

```
server {
    listen 80;

    location / {
        resolver 10.19.7.33 valid=30s;
        resolver_timeout 5s;
        set $upstream_host test2.zhe800.com;
        proxy_pass http://$upstream_host:82;
    }
}
```

resolver 3-7

表 3-7 resolver 指令说明

指 令	作 用	支持配置的环境
resolver	DNS 10.19.7.33:5353 53 valid DNS resolver DNS IP	http server location
resolver_timeout		http server location

注意： DNS set $upstream_host test2.zhe800.com IP proxy_pass Nginx DNS IP valid DNS DNS DNS IP Nginx IP

upstream Nginx zone

rewrite ngx_http_rewrite_module rewrite

3.6.1 内部重定向

rewrite server location if break last Nginx location URL HTTP

```
#      /a      URI                  /b   URI                  URI     /b
#      rewrite                     proxy_pass
   rewrite /a$  /b break;

#      /a      URI                  /b   URI
#      rewrite                     proxy_pass
   rewrite ^/a  /b break;

#    /a   URI              /b   URI
```

```
#       rewrite                      proxy_pass
   rewrite ^/a$  /b break;

#        /a   URI                   /b   URI
#       rewrite                     proxy_pass
   rewrite /a   /b  break;

#      /a/             /a/      URI         (.*)              URI
#            /b/$1 ,    $1              URI                  /b   URI
#       rewrite                     proxy_pass

  rewrite ^/a/(.*)   /b/$1  break;

# last           break                        URI
#               location   “   ”              location
   rewrite /a /b last;
   proxy_pass http://test_servers;
```

rewrite_log

指令：rewrite_log

rewrite_log on | off;

rewrite_log off;

http　server　location　if

注意：　rewrite　　URL

3.6.2　域名跳转

rewrite　　301　302

```
# permanent
#     (.*)  URL                          HTTP        301
  rewrite ^/(.*)$ http://www.zhe800.com/$1 permanent;

# redirect
#     (.*)  URL                          HTTP      302
  rewrite ^/(.*)$ http://www.zhe800.com/$1 redirect;
```

301

302

302

3.6.3 跳转 POST 请求

301 302 POST POST
GET HTTP 1.1 307 308

307 302 308 301
return

```
return 307 http://www.zhe800.com/$request_uri;
return 308 http://www.zhe800.com/$request_uri;
```

指令：return

return code [text];

code return HTTP
Default_Type

return

- return code URL; 301 302 303 307 308
- return URL; 302

server location if

注意： return proxy_pass return
proxy_pass location

3.6.4 设置变量的值

指令：set

set $variable value;

server location if

```
location / {
```

```
  set $a '1';
  set $b '2';
  set $ab  $a$b;      #
  return 200 $ab;     #       12          200
}
```

Nginx　　ngx_http_limit_req_module　　ngx_http_limit_conn_module

User-Agent

User-Agent

```
#        MSIE                        2KB/s
if ($http_user_agent ~* "MSIE") {
  limit_rate 2k;
}
```

Wiki

```
# geo                   IP
#        IP                          0   1                        default
#                    $wip
http {
  geo $wip  {
     default 0;
     #       IP
     127.0.0.1 1;
     172.16.0.0/16 1;
     172.17.0.0/16 1;
     172.18.0.0/16 1;
     10.0.0.0/8 1;
   }
# map         $wip   0             default         $binary_remote_addr
#      $limit      $wip   1     IP
   map $wip  $limit {
      0 $binary_remote_addr;
      1 "";
   }
```

```
#               $limit
limit_req_zone $limit zone=zone_acode:100m rate=25r/s;
server {...}
}
```

注意：　　http

Nginx　　ngx_http_log_module

3.8.1　记录自定义变量

指令：log_format

log_format name [escape=default|json|none] string ...;

log_format combined "...";

http

Nginx　　Cookie　IP User_Agent　　server_ip

-

```
set $a '123';
```

-

```
log_format  main  '$remote_addr - $remote_user [$time_local] $a'
```

3.8.2　日志格式规范

Nginx 1.11.8　　[escape=default|json|none]　　Nginx JSON　　POST

```
log_format json_log escape=json '{"ip":"$remote_addr","timestamp":
            "$time_iso8601",'
            '"host":"$http_host","request":"$request",'
            '"cookie":"$http_cookie","req_time":"$request_time",
            "uri":"$uri","referer":"$http_referer" }';
```

escape=json

JSON

3.8.3　日志存储

access_log

指令：access_log

access_log path [format [buffer=size] [gzip[=level]] [flush=time] [if=condition]];

access_log off;

access_log logs/access.log combined;

http　server　location　if in location　limit_except

-
- access_log
-
-
-

access_log　　error_log

```
# json_log
access_log /data1/access.log  json_log;

#                  5MB
access_log /data1/access_1.log combined gzip flush=5m;

#                  if    0
#                HTTP        4
map $status $loggable {
    ~^[4]  0;
```

```
        default 1;
    }
    access_log /path/to/access.log combined if=$loggable;
```

注意：Nginx I/O

Nginx HTTP Lua

HTTP ngx_http_core_module Nginx src/http/ngx_http_core_module.h

ngx_http_core_module.h

```
typedef enum {
    NGX_HTTP_POST_READ_PHASE = 0,

    NGX_HTTP_SERVER_REWRITE_PHASE,

    NGX_HTTP_FIND_CONFIG_PHASE,
    NGX_HTTP_REWRITE_PHASE,
    NGX_HTTP_POST_REWRITE_PHASE,

    NGX_HTTP_PREACCESS_PHASE,

    NGX_HTTP_ACCESS_PHASE,
    NGX_HTTP_POST_ACCESS_PHASE,

    NGX_HTTP_TRY_FILES_PHASE,
    NGX_HTTP_CONTENT_PHASE,

    NGX_HTTP_LOG_PHASE
} ngx_http_phases;
```

Nginx 11 Nginx HTTP 3-8

表 3-8　HTTP 执行阶段的作用

阶段顺序	阶段名称	作　　用
1	NGX_HTTP_POST_READ_PHASE = 0	
2	NGX_HTTP_SERVER_REWRITE_PHASE	URL
3	NGX_HTTP_FIND_CONFIG_PHASE	URL location
4	NGX_HTTP_REWRITE_PHASE	location URL
5	NGX_HTTP_POST_REWRITE_PHASE	URL 4 3 10 10
6	NGX_HTTP_PREACCESS_PHASE	
7	NGX_HTTP_ACCESS_PHASE	IP
8	NGX_HTTP_POST_ACCESS_PHASE	7
9	NGX_HTTP_TRY_FILES_PHASE	try_files
10	NGX_HTTP_CONTENT_PHASE	HTTP
11	NGX_HTTP_LOG_PHASE	

HTTP　　Nginx

4

Nginx Nginx
Nginx Nginx

注意： Nginx 1.2

HTTP

ID

Nginx

4.1.1 使用 ngx_http_headers_module 设置响应头

ngx_http_headers_module Nginx add_header expires

1．expires

expires [modified] time;

expires epoch | max | off;

expires off;

http　server　location　if in location

Expires　Cache-Control

CDN

```
expires -1;    #              cache-control: no-cache
```

1h

```
expires 1h;    #              cache-control: max-age=3600
               #         1h  max-age
```

HTTP　200　201　204　206　301　302　303　304　307　308

Nginx 1.7.9　expires

Content-Type

Content-Type　application/pdf　cache-control: max-age=3600

Content-Type　image/　ache-control: max-age=36000

Content-Type　default　off

```
map $sent_http_content_type $expires {
    default         off;
    application/pdf 1h;
    ~image/         10h;
}
expires $expires;
```

map　Content_Type　$expires

$sent_http_content_type　Content-Type

2. add_header

add_header name value [always];

http　server　location　if in location

Cache-Control　expires

expires

```
add_header Cache-Control no-cache;  #       expires -1;
```

add_header　HTTP　200　201　204　206　301　302

303 304 307 308 404 500
Nginx 1.7.5 always HTTP

```
add_header Cache-Control no-cache always;  #       expires -1;
                                           #     HTTP          500
```

```
 [root@testnginx ~]# curl -I http://testnginx.com/
HTTP/1.1 500 Internal Server Error
Server: nginx/1.12.2
Date: Tue, 30 Jan 2018 08:17:39 GMT
Content-Type: application/octet-stream
Content-Length: 1
Connection: keep-alive
Cache-Control: no-cache
```

500 always add_header

3．实战经验

- expires 1h

expires 1h 1h Last-Modified

Last-Modified

- add_header

always add_header
add_header
Bug

```
[root@testnginx ~]# curl -I http://testnginx.com/
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Tue, 30 Jan 2018 08:50:40 GMT
Content-Type: application/octet-stream
Content-Length: 1
```

```
Connection: keep-alive
Cache-Control: 1000
Cache-Control: no-cache
```

4.1.2　使用 headers-more-nginx 控制请求头和响应头

add_header　　HTTP

headers-more-nginx　　HTTP

https://github.com/openresty/headers-more-nginx-module.git

```
# wget 'http://nginx.org/download/nginx-1.12.2.tar.gz'
# git clone https://github.com/openresty/set-misc-nginx-module.git
# tar -xzvf nginx-1.12.2.tar.gz
# cd nginx-1.12.2/  &&  ./configure --prefix=/opt/nginx \
    --add-module=/path/to/headers-more-nginx-module
# make && make install
```

add_header

1

```
more_set_headers 'Cache-Control: 1000';
```

“ add_header Cache-Control 1000 always ”

2

```
more_set_headers  -s '200 301' 'Cache-Control: 1000';
```

add_header Cache-Control 1000　　200　301

more_set_headers

more_set_headers

headers-more-nginx　　more_set_headers　more_clear_headers　more_set_ input_headers　more_clear_input_headers

1．根据 HTTP 状态控制响应头

指令：more_set_headers

more_set_headers [-t <content-type list>]... [-s <status-code list>]... <new-header>...

http server location location if

output-header-filter

more_set_headers -s 404 -s '500 502' 'Result: error' 'F: X-re';

-s

404 500 502 'Result: error' 'F: X-re'

more_set_headers

2．根据 HTTP 状态清除响应头

指令：more_clear_headers

more_clear_headers [-t <content-type list>]... [-s <status-code list>]... <new-header>...

http server location location if

output-header-filter

more_clear_headers -s 200 -t 'text/plain' F Result;

200

'text/plain' F Result 3 -s

X-

more_clear_headers -s 200 -t 'X-*'

3．设置 HTTP 请求头

指令：more_set_input_headers

more_set_input_headers 'Host: testnginx.com ';

http server location location if

rewrite tail

rewrite

```
location = /testnginx {
    set $test 'testnginx';
    more_set_input_headers 'Host: $test';
```

```
}
```

4．清除 HTTP 请求头

指令：more_clear_input_headers

more_clear_input_headers -t Cache-Control;

http　server　location　location if

rewrite tail

rewrite　　Cache-Control

“ more_clear_input_headers 'Test-*'; "　　Test

5．实战经验

- more_clear_headers

varnish

PHP

- more_set_input_headers　more_clear_input_headers

rewrite tail

- headers-more-nginx

set-misc-nginx　rewrite　　URL

SQL

https://github.com/openresty/set-misc-nginx-module.git

```
# wget 'http://nginx.org/download/nginx-1.12.2.tar.gz'
# git clone https://github.com/simplresty/ngx_devel_kit.git
# git clone https://github.com/openresty/set-misc-nginx-module.git
# tar -xzvf nginx-1.12.2.tar.gz
```

```
# cd nginx-1.12.2/
# ./configure --prefix=/opt/nginx \
     --add-module=/path/to/ngx_devel_kit \
     --add-module=/path/to/set-misc-nginx-module
# make && make install
```

注意： ngx_devel_kit ngx_devel_kit Nginx

set-misc-nginx

4.2.1 设置变量

指令：set_if_empty

set_if_empty $dst <src>

location location if

rewrite

```
location = /a  {
     set $t $arg_test;        #     URL        test
     set_if_empty $t 123;     #                          123
}
```

Nginx Nginx

if if “ ” 18

```
location = /a  {
     set $t $arg_test;       #     URL        test
     if ($t = '') {
        set  $t 123;         #                          123
     }
}
```

4.2.2 防止 SQL 注入

指令：set_quote_sql_str

set_quote_sql_str $dst <src> set_quote_sql_str $dst

location location if

rewrite

```
location /test {
    set $v1 "testnginx\n\r'\"\\"; #
    set_quote_sql_str $v2 $v1;    #     set_quote_sql_str  $v1      $v2
                                  #       MySQL                    SQL
    echo $v1;                     #      echo        testnginx\n\r\'\"\\
}
```

MySQL SQL

PostgreSQL set_quote_pgsql_str set_quote_sql_str

```
location /test {
    set $v1 "testnginx\n\r'\"\\";
    echo $v1;
}
```

```
[root@testnginx ~]# curl http://testnginx.com/test
testnginx
'"\
```

4.2.3　字符串非转义和转义

指令：set_unescape_uri

set_unescape_uri $dst <src>　set_unescape_uri $dst

location　location if

rewrite

```
location /test {
                  #                                          $key
    set_unescape_uri $key 'hello+world%21';
    echo $key;    #     echo        hello world!
}
```

URL set_unescape_uri $key $arg_name

指令：set_escape_uri

set_unescape_uri set_escape_uri
set_unescape_uri

4.2.4 基于键值的集群分片

Nginx API Application Programming Interface

set_hashed_upstream

指令：set_hashed_upstream

set_hashed_upstream $dst <upstream_list_name> <src>

location location if

rewrite

```
upstream memcache1 { ... }
upstream memcache2 { ... }
upstream memcache3 { ... }

upstream_list universe memcache1 memcache2 memcache3;

location /test_ngx {
    set_unescape_uri $key $arg_key;
    set $list_name universe;
    set_hashed_upstream $backend $list_name $key;
    echo $backend;
}
```

upstream_list 3 HTTP /test_ngx URL key hash upstream_list upstream Memcahed

4.2.5 base 编码

base

指令：set_encode_base32

set_encode_base32 $dst <src> / set_encode_base32 $dst

location　location if

rewrite

1

```
location /test {
    set $a "nginx";
    #      $a     base32            $test  $a
    set_encode_base32 $test $a;
    echo $test;
}
```

```
[root@testnginx ~]# curl http://testnginx.com/test
dpjmirjo
```

2

```
location /test {
    set_encode_base32 $test $request_uri;  #   request_uri
    echo $test;
}
```

```
[root@testnginx ~]# curl http://testnginx.com/test?sdaddsads12312
5tq6asrk7tpm8ob4chpm2p3j64p36c9i
```

指令：set_decode_base32

set_decode_base32 $dst <src> / set_decode_base32 $dst

location　location if

rewrite

set_encode_base32　　<src>　　base32

set_encode_base64　base64　　set_decode_base64　base64

base32

4.2.6 md5 编码

指令：set_md5

set_md5 $dst <src> | set_md5 $dst

location location if

rewrite

<src> md5 $dst

```
location /test {
    #   $request_uri          $test        md5
    set_md5  $test $request_uri;
    echo $test;
}
```

```
[root@testnginx ~]#  curl http://testnginx.com/test?sdaddsads12312
9a9d7101420a744b68debe3d7bb91eb3
```

4.2.7 生成随机数

指令：set_random

set_random $res <from> <to>

location location if

rewrite

from to

```
location /test {
    set $from 1;
    set $to 100;
    #           1  100                    $result
    set_random $result $from $to;
```

```
    echo $result;
}
```

```
[root@testnginx ~]#  curl http://testnginx.com/test
96
```

指令：set_secure_random_alphanum

set_secure_random_alphanum $res <length>

location　location if

rewrite

<length>　　a~z　　A~Z　　0~9

```
location /test {
    set_secure_random_alphanum $result 16;
    echo $result;
}
```

```
[root@testnginx ~]# curl http://testnginx.com/test
e3nwGeMF39FebMVY
[root@testnginx ~]# curl http://testnginx.com/test
aF1c89KLugUXaU9T
[root@testnginx ~]# curl http://testnginx.com/test
Yd8dkzXjE5DdBUx7
[root@testnginx ~]# curl http://testnginx.com/test
CSa2m3fyoUDywRAv
```

指令：set_secure_random_lcalpha

set_secure_random_lcalpha $res <length>

location　location if

rewrite

set_secure_random_alphanum　　<length>

a~z

4.2.8 本地时间的输出

指令：set_local_today

set_local_today $dst

location location if

rewrite

("yyyy-mm-dd") $ds

```
location /test {
    set_local_today $a;#                    ("yyyy-mm-dd")          $a
    echo $a;
}
```

```
[root@testnginx ~]# curl http://testnginx.com/test
2018-03-13
```

Nginx Ngx_Lua

4.2.9 实战经验

- set_if_empty

  if set_if_empty Nginx

- set_quote_sql_str

  Nginx set_quote_sql_str SQL

- set_unescape_uri set_escape_uri

  URL

- set_hashed_upstream

  upstream “ ”

Nginx　　ngx_http_image_filter_module

- Nginx　　--with-http_image_filter_module
- libgd　gd-devel　　CentOS　　yum

4.3.1　image_filter 图片处理

指令：image_filter

```
image_filter off;
image_filter test;
image_filter size;
image_filter rotate 90 | 180 | 270;
image_filter resize width height;
image_filter crop width height;
```

image_filter off;

location

- image_filter off;

- image_filter test;

JPEG　GIF　PNG　　WebP　　415

- image_filter size;

JSON

```
{ "img" : { "width": 750, "height": 286, "type": "jpeg" } }
```

- image_filter rotate 90 | 180 | 270;

resize

crop

- image_filter resize width height;

  width height

  -

  415

- image_filter crop width height;

  width height

  -

  415

1

JPG test.jpg /usr/local/nginx_1.12.2/conf

```
location /test.jpg {
    image_filter size;
    root /usr/local/nginx_1.12.2/conf;
}
```

JSON

```
{ "img" : { "width": 750, "height": 286, "type": "jpeg" } }
```

2

```
location /test.jpg {
    image_filter resize 150 100;  #
    image_filter rotate 180;      #              180°
    root /usr/local/nginx_1.12.2/conf;
}
```

4-1

4-1

4-2

4-2

```
location /test.jpg {
    image_filter crop 150 100;    #      150×100
    image_filter rotate 180;      #              180°
    root /usr/local/nginx_1.12.2/conf;
}
```

4-3

4-3

4.3.2　采用渐进式方式打开 JPEG 图片

Nginx

image_filter_interlace

指令：image_filter_interlace

image_filter_interlace on | off;

image_filter_interlace off;

http server location

on JPEG

注意：

- Nginx 1.3.15 1.4
- CPU
- KB

4.3.3 WebP 格式

WebP WebP

30%

Nginx 1.11.6 WebP WebP

WebP

4.3.4 优化图片

指令：image_filter_buffer

image_filter_buffer size

image_filter_buffer 1M

http server location

415

注意：

image_filter_buffer image_filter_buffer

error.log

```
[error] 13313#0: *272 image filter: too big response: 161036 while sending response to client
```

Û ¸ Ö image_filter_jpeg_quality

AÁ"© Ûmage_filter_jpeg_quality quality¥

T¬Ax Ö ´

)ƒ W Öhttp ÃserverÃlocation

ÿ ÖA’5B Ò(E@ 6 JPEG ã â,XBüG£ Ä – D ,X8× È ü 1~100 KÈ È – D C^ ã ã G- Ò(BüG£C^ Â È à Ê È D B ôEgG£ 3C^ ã Ä – D Ã ¹ ¬G£ È |9$ Ô ûA’ 95 Ä

"¼ ã Ö • Ù U 8 J • Ÿ Ú ì • C U 8 J Š L ª O \ å Ñ Î ÷ C U Â J X V ¾ y Z t Í ª ö ? \ C J • Ù š 70~85 è Î Ñ J • Ÿ • ; ? Ø \ 8 Š I J » Í ´ è ; ? ª ö ¶ - ¶ [- z ¶ • Ù Í

Û ¸ Ö image_filter_sharpen

AÁ"© Ûmage_filter_sharpen percen¥t

T¬Ax Öimage_filter_sharpen 0;

)ƒ W Öhttp ÃserverÃlocation

ÿ ÖA’5B Ò(,X#Ù D z È – D ÄJä z,R Ú!¨ Å Ã ¹CYE› 100 ×T¬Ax 0 È></ /U*üJä ê × – D Ã ¹ ¬G£ Ä

Û ¸ Ö image_filter_transparency

AÁ"© Ûmage_filter_transparency on | off;

T¬Ax Öimage_filter_transparency on;

)ƒ W Öhttp ÃserverÃlocation

ÿ ÖA’5B ' ^ Ò(E@ 6 GIF ã ê PNG ã Ê ú ±+-Eã â z Ä PNG ã ,X AlphaEîF'Eã â z Ÿ4œ ± Õ á ¬ È ´!8 ü Ø)Ú PNG ã,X Ò(Ê ÈEã â z á îLc- – D ,X ¬ ê5à ¬ ê Ä

Û ¸ Ö image_filter_webp_quality

AÁ"© Ûmage_filter_webp_quality quality;

T¬Ax Öimage_filter_webp_quality 80;

)ƒ W Öhttp ÃserverÃlocation

ÿ ÖA’5B Ò(E@ 6 WebP ã â,XBüG£ Ä Ã y «,X – D ü 1~100 KÈ È – D C^ ã ã G- Ò(BüG£C^ Â È à Ê È D B ôEgG£ 3C^ ã Ä

"¼ ã Ö Ÿ É > š Nginx 1.11.6 5 è Q « Í WebP « 0 • Ÿ 8 5 Ú í 5 JPEG « 0 • Ÿ 8 5 J ` g Y ^ p WebP « 0 • Ÿ Ú J ; _ X V ¾ • Ÿ • J ^ ? ° ß ! Ž y É > image_filter_webp_quality

4.3.5 r ì4£P` Ö | Õ Û Ò

ngx_http_image_filter_module,X û î D Û ¸FÑ Õ S*ü ¬G£ ÈE- | Õ Ø)Ú Ò(¤ o Z } ÄFw V) B Ò(ã ` û ã1 – D | Õ ¤ o Ò(6 Û ßM6 Ô þ1T),XG!5B/ _ Ö

```
location ~* ^/(.+)\.(\d+|-)x(\d+|-)\.(gif|png|webp|jpg)$ {
    set $w $2;  #          9‹ ª G! ,X    URL  ,X +0ú È ^ )9‹,X  A’  ‘
    set $h $3;  #          9‹ ª G! ,X    URL  ,X +0ú È ^ )9‹,X  A’  P¬
    image_filter resize $w $h;  #                | Õ>• > ¡ 0
    image_filter_buffer 3M;     #                AÏ ª s Ÿ Ò( ,X Ô û
    try_files /$1.$4  /404.jpg;  #               ü.@,¬  AÏ ª [ Ê Ê È V pAÏ ª á  È
                          #                      íAÏ ª  /404.jpg
    root /usr/local/nginx_1.12.2/conf;
}

location = /404.jpg {
    return 404;
}
```

"¼ ã Ö

x U S Z @ G J M F T { y X ` g À y U S Z @ G J M F T J ë 0 Ž y S P P U + y _ Š } ¡ H J G ¹ ¡ Û š Ñ J C X ‹ c r ¨ J X h ! ¨ ? Þ š • Ÿ m Ÿ Ñ c * â Í ë ` g Ž y U S Z @ G J M F T J G ¹ ¡ H ´) À , Ñ J X 0 ° » K Q H ¡ H J í è † g J " È e • è ¹ ¡ H À Û š Í

x J N B H F @ G J M U F S S F T J [F X I X ` g 2 { Þ + Ú J X ' w r ¨ J ; ? ź ñ Þ × 7 3

5 $ 1 ¹ 6 % 1)3

ü Nginx 1.9.0(! È TCP ·)Ú Ô8 î S*ü Haproxy ê Nginx ,X1 Ý • Ù nginx_tcp_proxy_module Ä Nginx ü 1.9.0 (â t 9 Z ngx_stream_proxy_module È W ¤ o Z TCP ·)Ú È ¾LÔ ü4êA¥Nginx Ê t 9 --ngx_stream_proxy_module Ã Ä

4.4.1 ·)ÚG!5BAÈ â

,ß Ô þ/ _ Ö

```
worker_processes 2;

error_log /var/log/nginx/error.log info;

events {
   worker_connections  1024;
}

stream {
   upstream backend {
      hash $remote_addr consistent;
      server x.testnginx.com:11233 weight=5 max_fails=3 fail_timeout=30s;
      server 127.0.0.1:11233       weight=2 max_fails=3 fail_timeout=30s;
      server 192.168.100.10:11233  weight=3 max_fails=3 fail_timeout=30s;
      server unix:/tmp/backend3    weight=1 max_fails=3 fail_timeout=30s;
}

   server {
      listen 11233;
      proxy_connect_timeout 1s;
      proxy_timeout 3s;
      proxy_pass backend;
   }

}
```

ÞEÄG!5B Ú '),„ Z TCP ·)Ú,X(M&• Ä

x TCP ·)Ú ü streamÛ ¸ + YE¯> Ä â È ! b main Y È ` http Û ¸ + à4{ Ä

x ¡ å ·)Ú,X upstream Õ DNS ³ á Ä V ip_hashÃsocketÅG!5B Ã G¡ G LpE@/Ï Ä V max_failsÅG!5B Ä

x ' proxy_pass·)Ú TCP Ê È"u Ý http:// !4Ô È"¼ ãG!5B Ê á?U mJí Ä

x Õ ` HTTP Ô ,XE² yCY Ê – D proxy_timeoutÃproxy_connect_timeouÄ

TCP ·)ÚE¬ Ù ÿ J ª \ î Û ¸ È ßM6 Ú Ÿ4¡ Ô o *ü,X Û ¸ Ä

Û ¸ Ö proxy_bind

AÁ"© Ø proxy_bind address [transparent] | off;

T¬Ax Ö ´

)ƒ W ÖstreamÃserver

ÿ Ö V p!8G!5B off È í></ AË" Ú î S*ü2Ï4³7¾ | ÚG!,X IP È G â0Ã á u

,ß á *ü ,X,ó r IP × V pG!5B proxy_bind $remote_addr transparent;È í â0Ã á u Ã ¹,ß *ü ,X,ó r IP Ä Z S – D*ó ÈLÔ?U ¹CY4{*ü L$E¤> Nginx ,X worker E¯ /ß È JG!5B —CÃ+ >< ¹ þ9‹ ¡ å ·)Ú á u <,X5%4°# G£ Ä

Û ¸ Ö proxy_download_rate

AÁ"© Ø proxy_download_rate rate;

T¬Ax Öproxy_download_rate 0;

)ƒ W ÖstreamÃserver

ÿ ÖA’5B â0Ã á u <Aï ª D B,XEó z ÄEó z n +8V!£/¦ ÄT¬Ax 0 È></ /U*üL$ Eó ÄL$EóA’5B,X Í!£ þE² yFÑ Ý Ä

Û ¸ Ö proxy_next_upstream

AÁ"© Ø proxy_next_upstream on | off;

T¬Ax Öproxy_next_upstream on;

)ƒ W ÖstreamÃserver

ÿ Ö ' ´"© â ' !,X â0Ã á u < Î0ŸE² y Ê ÈA¹ Û ¸*ü 9.B n ú Ú v 0ÃE² y ôEæ4- ß Ô Ä â0Ã á u < ÄE- Ã6Ñ î « ñA© õ D Ä proxy_next_upstream_tries ` ÊKÈ Ä proxy_next_ upstream_timeout,X E ¡ Ä

Û ¸ Ö proxy_next_upstream_timeout

AÁ"© Ø proxy_next_upstream_timeout time;

T¬Ax Öproxy_next_upstream_timeout 0;

)ƒ W ÖstreamÃserver

ÿ ÖA’5B ôEæE² y ß Ô Ä â0Ã á u <,X ÊKÈ ÄT¬Ax 0 È></ GKÁE- þL$ Ä

Û ¸ Ö proxy_next_upstream_tries

AÁ"© Ø proxy_next_upstream_tries number;

T¬Ax Öproxy_next_upstream_tries 0;

)ƒ W ÖstreamÃserver

ÿ ÖA’5B ôEæE² y ß Ô Ä â0Ã á u <,X ñA© õ D ÄT¬Ax 0 È></ GKÁE- þL$ Ä

Û ¸ Ö proxy_pass

AÁ"© Ötproxy_pass address;

T¬Ax Ö ´

)ƒ W Öserver

ÿ ÖA’5B>• ·)Ú,X á u < Ä Ã ¹ Ô þ ³ á È 3 Ã ¹ IP t0Ã · Ë Ä ¢ Nginx 1.11.3(Ô Ÿ ÕG!5B ¬G£ È V proxy_pass $upstreamÆÄ

Û ¸ Ö resolver

AÁ"© Ötresolver address ... [valid=time] [ipv6=on|off];

)ƒ W ÖstreamÃserver

ÿ Ö üE¯> DNS?· d Ê ÈA¹ Û ¸*ü b Í upstream Î)„„X ³ áE¯> IP ?· d È' â ÚAË" ·)Ú ?· d ,X IP Þ × Valid=time ></ DNS ?· d â,X4ç , ÊKÈ È S*ü4ç , ÊKÈ Ã ¹ £ å DNS ?· d,X õ D × G b ipv6=on|off È V pA’5B on È í></ ü ¹A¶ IPv4 ÄInternet Protocol version 4È f6(5% #A,(4 Å ´ p â î4»4Á ¹A¶ IPv6 ÄInternet Protocol Version 6È f 6(5% #A,(6 ÅÈ V p áLÔ?U ¹ R IPv6 È Ã ¹G!5B off È ¹4ý-Á ¹ R ÊKÈ Ä

Û ¸ Ö resolver_timeout

AÁ"© Ötresolver_timeout time;

T¬Ax Öresolver_timeout 30s;

;> L !‰ Ö streamÃserver

ÿ ÖA’5B?· d DNS,XCY Ê ÊKÈ È V pCY Ê ÈDNS î S*ü Þ Ô õ?· d ,X IP Ä

Û ¸ Ö proxy_protocol_timeout

AÁ"© Ötproxy_protocol_timeout timeout;

T¬Ax Öproxy_protocol_timeout 30s;

)ƒ W ÖstreamÃserver

ÿ ÖA’5BAï ª proxy #A, ,X ÊKÈ Ä V p ü A’5B,X ÊKÈ Y"u Ý ¥EÕ ` H,X y È í G KÁE² y Ä

Lc- Nginx(,X á • È „ È „„X – D 3 ü á • t 9 ÈAï5Ù Ã ¹Lc Ê G"¼ l5%,X Wiki Ä

4.4.2 DNS á u,X ¡ å ·)Ú

ü 1.9.13(â È Nginx t 9 Z Í UDP ÄUser Datagram Protocol å·)Ú,X Õ È ´!8 Ã ¹ S*ü Nginx 9 ·)Ú DNS,X UDP0Ã · ÈG!5B V ß Ö

```
stream {
    resolver_timeout 8.8.8.8;

     upstream dns {
       server 192.168.1.3:5343;
       server dns.testnginx.com:5343;
     }

     server {
        listen 127.0.0.1:53 udp;
        proxy_responses 1;
        proxy_timeout 20s;
        proxy_pass dns;
     }

}
```

4.4.3 MySQL Lš5x ·)ÚG!5B

ü MySQL D B g Ô î ¢,X Š X È Ã ¹ S*ü Nginx ,X TCP ·)Ú 91u)Ú `4È x ¢ g,X D B ÈG!5B V ß Ö

```
stream {
    upstream mysql_servers {
        least_conn;  #                AË" î ì ¥4-E² y å,X á u <
        server 192.168.0.34:3306 max_fails=2 fail_timeout=30s;
        server 192.168.0.35:3306 max_fails=2 fail_timeout=30s;
    }

server {
    listen 3306;
    proxy_connect_timeout 1s;
    proxy_timeout 3s;
    proxy_pass mysql_servers;    }
}
```

ÞEÄG!5B,ß Þ • \ #Ù(È V p ¤ Ä MySQL á u < ¬ 6 ê i È Nginx î7¾ | Ú W Cö Î • Ä

4.4.4 r ì4£P`

1 Ê Û ¸ nginx_tcp_proxy_module` ngx_stream_core_moduleÑ Ã ¹ ü TCP ·)Ú S*ü Ä J ngx_stream_core_moduleÙ ÿ Ý G ¹ «A„),X Ô o *ü ¬G£ ÈEîE›E- o ¹ « µ C Ã ¹ Ú d !£ þ TCPAË" ,X ™ ‰ Ä

2 Ê ü . TCP ·)Ú Ê È Nginx ·)Ú8V&• á u < Ã6Ñ á!6 Ô Ä È Ã ¹ S*ü DNS EBA¶ êLVS ÄLinux Virtual ServerÈLinux <. ³ á u < Å1 s6Ñ Í Nginx E¯> BóEQ >5 Ä

3 ÊRedis 2 b) á u È ¹ ü Ô Ä á u < Þ | î þ Redis r _ \!7 ,X Ä V p î þ Redis ¢ gF¼5F ü à Ô Ä á u < Þ È Ã ¹EîE› TCP ·)ÚE¯> E@ ¥ È ú í î ´ 0Ã · á Ô7È È Ð 7ÈG!5BM2 &º)ä Ä

e*U N „ ø3ú

ßM6 Ÿ4¡ Ô o *ü,X Nginx õ + È W À ¡ å ·)Ú ¤ o Z È î,X s6Ñ È J è Ã ¹L! " â0Ã ·-Õ,X á z Ä

4.5.1 Î bA“KÂ IP CÇE@ Í h ¢ Ö

ü Ý o ™ ‰ ß È5%0-NIM6,X)/ `*ü ü,X ¢ Ö Ý G È V ¶C Ã ì ´ á u1 ÄFw È App Ã ¹EîE› GPS n !*ü ü,X ¢ Ö È PC0Ã í Ã ¹EîE›A“KÂ IP 9 ø • Í h,X ¢ Ö Ä

' PC0Ã ¥EÕAË" Ê È î B v 0Ã,X IP ø • J 2 b ¾ þ ¢ Ö È J ÚAË" EîE› 302CÇ E@ Í h ¢ Ö,X5%0-NIM6 Þ È BA“KÂ IP CÇE@ Í h ¢ ÖNIM6,X î uF Ee V Ò 4-4 / Ä

=] 4-4 -Ú2>,ä>⊣P +yD <ÙDË+^,5AÇ)è;1A 6 +eA >R5@0Ž

Ò 4-4 ,X GK &• V ß Ä

x LÔ?U Ô ÑP 2 ¢ Ö,X#Ù) Ä5% Þ Ý \ î Ô$d,X IP Í h ¢ Ö g ÈAÏ5Ù Ã7¾>
ßEQ Ä V p Í IP µ C,X š.B z?U" EWP¬ È ÎA, S*ü ¬C G Ä Ä

x IP Í h ¢ Ö g Ã6Ñ î , üAÃ Â È 3 Ã6Ñ*ü ü,X ¢ Ö"u Ý,Ì G5%0-,X á u È ¹7È
´"© ` äCÇE@ È ´!8LÔ?U . Ô o Ÿ Q ¤/ È • “*ü Û 6 Ç •,X ¢ ÖNIM6 Ä

x V p*ü ;> Z Û 6 ¢ Ö,X ¡ 0 È Ã ¹Ax *ü î üE- þ ¢ Ö4»4ÁA“KÂ È ¹ á ™ ü!£
õAË" ÊFÑ Í IP E¯> ø • È ¾LÔEîE› !0Ã,X JS ·-ÕA’5B Cookie G Ã Ä

Z?· —,X GK &• â È Ã ¹EîE› Nginx E¯> Ø)Ú Z Ä

õ + Ö ngx_http_geo_moduleÄNginx T¬Ax Õ!8 õ + È áLÔ?UNq ê#Ï t Å Ä

Û ¸ Ö geo

AÁ"© Ögeo [$address] $variable { ... }

)ƒ W Öhttp

/ _ V ß Ö

```
geo $geo {
  ranges;  #          ¹ IP    !‰,X 6 ã n    È      IP    !‰ ™NO  üOj ! È è  Z ¤  >™EQ   IP
       #              g,X û6Ñ È h Ý  c f ë
  default  beijing;  #              8¹ ´"© G!  Í h,X     IP    !‰ È í S*üT¬Ax
  1.0.32.0-1.0.63.255 guangzhou;  # IP                  !‰ Í h,X ¢ Ö È*ü0N  Ú Ô
  1.2.2.0-1.2.2.255 beijing;
  1.2.4.0-1.2.4.255 beijing;
  1.24.0.0-1.24.31.255 huhehaote;
  1.24.32.0-1.24.39.255 wulanchabu;
}
```

ÿ Ö 'AË" 4£E› http ;> L !‰ Ê È geo Û ¸T¬Ax î ¢ $remote_addrØ9‹ ª IP ,X È ' â • ` $addressE¯> G! È V p G! ä s È í9‹ ª $addressÍ h,X È J Ú!8 C 4- $geo× V p G! Bù È í S*ü default,X È G beijing Ä

ßM6 ¤ o Ô þ1T),X Nginx G!5B È W Ã ¹ r)„EîE› IP CÇE@ Í h ¢ ÖNIM6,X s6Ñ Ö

```
location = / {
  if ($cookie_current_city){
    set $geo $cookie_current_city;
    # $cookie_current_city          ></ *ü  Ã6Ñ Æ4£EÝ ½E› ¢ Ö È ê5Ù Æ4£  BIP   CÇE@
    #          Z Í h,X ¢ ÖNIM6 È  ¹,È yCÇE@     Cookie   Í h,X ¢ ÖNIM6 G Ã
    rewrite ^/ $geo.testnginx.com redirect;
  }
```

```
        rewrite ^/ $geo.testnginx.com redirect;
    }
```

ü ÞEÄG!5B È ü*ü E¯ 95%0-OjNI â È##?œ < î B*ü ,X IP CÇE@ Í h,X ¢ Ö NIM6 È V p*ü ¥)„CÇE@ ÝAÃ Â ê Ç 6 þ ¢ ÖNIM6,ß,ß ÈFw Ã ¹ Û 6 7¾ Å Ç,ß,X ¢ ÖNI M6 È!8 Ê È !0Ã JS ·-Õ Ã ¹EîE›A'5B CookieÈ.B ± ß õ*ü A“KÂOjNI Ê îT¬AxE¯ 9 Û 6 â,X ¢ ÖNIM6 Ä

"¼ ã Ö

x Cookie Ÿ z ¶ & • 0 Í

x 6 ¸ ´ ˆ Ô â Ê) ¬ • ç Û š J ` g } â Ê å & +) ¬ J g æ Ÿ { +
â Ê † g z Á ¡ ¶ J X z Á , G † g Í

4.5.2 Â ¡ h Y •

4 úF E›E- ,X C Ö ü h*ü á u ¥ × „ ·-Õ â È ¥)„ ¤ þ +0ú ê Y • Ý Bug È5à V p Â ·-Õ,XA± ÈLÔ?UG¡ „ Ü J ÃP`A•!OJÒ Ã ” ¹ á u ÈE¬?UFS ! E ¡ J ª s6Ñ È*î7ÇE¬ Ã 6ÑLÔ?U ² & Ô õ á u ÄFw ü ÊKÈ ÝL$,X ™ ‰ ß ÈA¹ V) Ø)Ú 6 Û

Nginx ,XE›$¸ õ + ngx_http_sub_moduleÃ ¹?· ‡E- þKÂNl Ä W Nginx ,X Y5B õ + È Ã ¹ *ü 9 Â ¡ h Y •,X D B ÄA¹ õ +T¬Ax þ>•%”# È V p?U%”# ,XA± È ¾LÔ ü4êA¥ Nginx Ê#Ï t --with-http_sub_moduleG Ã Ä

Û ¸ Ö sub_filter

AÁ"© Ösub_filter string replacement;

)ƒ W Öhttp ÃserverÃlocation

ÿ Ö Ú ü ¡ h Y • G! ,X stringÄ Ñ+9 û ã m Å Ó 6 ä replacementÄ

/ _ V ß Ö

```
    location / {
        sub_filter            href="http://              href="//      ;
        sub_filter              <img    src="http://img.testnginx.com                  <img    src="//i1.
testnginx.com      ;
    proxy_pass http://servers;
    }
```

ü ÞM6,X/ _ ÈEîE› Û ¸ sub_filter Â Z ¡ h Y •,X ø þ !5B Ä J Ô þ Ú5%4° # A, ¢ HTTP Û 6 HTTPSÈ G Ú http:// Ó 6 ä // ÄE- ÈNIM6 Ù ÿ,XJÒ y Ã ¹ B ' !

A“KÂ,X ³ á #A, 9 ‡ nAË" ,X #A, Z È Ã ¹ • “ #A,,X Û 6 Ä

!8 õ +E¬ Ù ÿ Ô o J ª,X Û ¸ Ä

Û ¸ Ö sub_filter_last_modified

AÁ"© Ösub_filter_last_modified on | off;

T¬Ax Ösub_filter_last_modified off;

)ƒ W Öhttp ÃserverÃlocation

ÿ Ö ü Ó 6 Y • óKÈ È ±+- 97¾ s Ÿ ¡ h,X Last-Modified ¡ h Y • È ¹ “ Ú ¡ h D BE¯> 4ç , Ä V p Ú JA’5B off È í></ á ±+- Ä

Û ¸ Ö sub_filter_once

AÁ"© Ösub_filter_once on | off;

T¬Ax Ösub_filter_once on;

)ƒ W Öhttp ÃserverÃlocation

ÿ Ö V p Ú JA’5B on È></ Ó 6 Ý G! ,X ¡ h Y • × V p Ú JA’5B off È í ¾ î Ó 61 Ô õ G! ,X ¡ h Y • Ä

Û ¸ Ö sub_filter_types

AÁ"© Ösub_filter_types mime-type ...;

T¬Ax Ösub_filter_types text/html;

)ƒ W Öhttp ÃserverÃlocation

ÿ Ö Ó 6 ¡ 0T¬Ax Í MIME ÄMultipurpose Internet Mail Extensions È sî s6Ñ ´(M5%F, Ê =) Å2O _ text/html ,X Y •*ó È VLÔ Ó 6 J ª2O _,X Y • È Ã ¹ S*ü * 9></ Ó 6 Ý2O _,X Y • Ä

4.5.3 LÊ £2ô [Ê,X*ó ä ž J 0*ü

|9$2Ï4³E¥´ HM2 &?'Ú È â0Ã á uEîE› û D B Ú d4-*ü ¤ oLÔ?U,X µ C ÈFw û D B ,X D B ¢)5à 9 6 Û

!¨EW *ü,X • ã È Ú*ü ,XA“KÂ> A„) ü Ô þ URL – DG ÈEîE› JS ¥EÕ á u < Þ È' â S*ü û D B T Ú dE- o URL ,X – D ÄE-G URL ,X 0*ü A„)*ü ,XA“KÂE<EÍ È áLÔ?U Ý ¡ h Y •E¨ ² È ´!8 URL Ã ¹ S*ü Ô þLÊ £2ô,X Ò(Ä5Ñginx ,X õ +

ngx_http_empty_ gif_module¢7 Q ¤ o Z!8 s6Ñ È WT¬Ax ü Nginx 4êA¥ Ê]>™ Ä

Û ¸ Ö empty_gif;

T¬Ax Ö ´

)ƒ W Ölocation

/ _ V ß Ö

```
location ~* fx_\.gif$ {
    add_header Cache-Control "no-cache, max-age=0, must-revalidate";
    empty_gif;
}
```

ÿ Ö Ï) ¹ fx_.gif 4§ ,X [ÊFÑ î*ó ä Ô þ0N,Q,XLÊ £2ô Ò(È J èEîE›A'5B Cache-Control.B ±AË" ™NO ²$d È á6Ñ>•4ç , Ä ü location S*ü Z!7 í><E' ã È ?U Z • "-è ¥ ¶Kó Ä V p !0ÃLÔ?U Ô þ „,XLÊ £2ô Ò(È,È yA"KÂ new1_fx_.gif [Ê Ã ¹ Z È áLÔ?UEî E›E¤4È ê Þ4" á u 9*ó ä URL Ä

"¼ ã Ö

x F ø ° » + ¹ J i d â 5 J n ` g ' : ÷ X W ò · Z k CPU — Ü + ¹ J , e J 6 G L ñ F J L 1 z ¶ ' Í

x " ™ d w ë ú 0 Š L ; ? 0 ÿ $ 8 L O L 1 ÷ URL J X , URL Ø - . ± ` « À L 1 ñ F ' w Í 3 1 J Þ X ‰ d J J 6 â ^ ÷ ´ * Í

4.5.4 Ò(,XL ,«JÒ

ZL !6,«JÒ,X ™ ‰ Î)„ È Ã ¹ S*ü ngx_http_referer_moduleœ + Ä!8 õ + Nginx ,X Y 5B õ + È áLÔ?UG¡ „4êA¥ Ä

L ,«JÒ/ _ V ß Ö

```
location ~* \.(gif|jpg|png|webp)$ {
    valid_referers none blocked server_names
            *.testnginx.com   ~\.baidu\.
            ~\.google\.;

    if ($invalid_referer) {
      return 403;
    }
}
```

ÿ Ö/ _ ,X referer *.testnginx.comÃbaiduÃgoogle,X ³ á ÈE- o ³ áFÑ Ã ¹!7 A"

location referer $invalid_referer 1
403

注意： CDN CDN
CDN
CDN

Nginx Nginx

Nginx

5

HTTP varnish squid proxy_cache
Nginx proxy_cache
Nginx

proxy_cache ngx_http_proxy_module 3 proxy
proxy_cache Nginx

proxy_cache

```
    upstream test_servers {
        server 127.0.0.1:81  max_fails=5 fail_timeout=10s weight=10;
        server 127.0.0.1:82  max_fails=5 fail_timeout=10s weight=10;
    }
    #
    proxy_cache_path  /data1/nginxcache  levels=1:2  keys_zone=cachedata:100m
inactive=7d max_size=50g use_temp_path=off;
    server {
        listen 80;
        location / {
```

```
        #
        proxy_cache cachedata;
        #          HTTP
        proxy_cache_valid        200 304  10s;
        proxy_cache_valid        301 302 100s;

        #
        proxy_cache_min_uses 2;

        #          key
        proxy_cache_key $scheme$host$uri$is_args$args;
        #       HTTP
        proxy_cache_methods GET HEAD
        #
        add_header N-Cache-Status $upstream_cache_status;

        #       on               HEAD        GET
        proxy_cache_convert_head on;

        sendfile on

        proxy_set_header     Host $host:$server_port;
        proxy_set_header     X-Real-IP $remote_addr;
        proxy_set_header     X-Forwarded-For $proxy_add_x_forwarded_for;

        #
        proxy_pass http://test_servers;
    }
}
```

proxy_cache 5-1

表 5-1 proxy_cache 相关指令说明

指 令	用 途	支持的配置环境
proxy_cache	keys_zone off	http server location
proxy_cache_valid		http server location
proxy_cache_key	key key URL $scheme$proxy_hosturiis_args$args	http server location

续表

指　令	用　途	支持的配置环境
proxy_cache_path	1　levels　levels=1　2 2　keys_zone keys_zone=cachedata:100m　cachedata 100MB 3　Inactive　Inactive=7d　7 4　max_size 5　use_temp_path　on proxy_temp_path I/O use_temp_path　off	http　server　location
proxy_cache_convert_head	on　HEAD　GET off　key　$request_method $host$request_uri$request_method	http　server　location
proxy_cache_methods	GET HEAD	http　server　location
proxy_cache_min_uses	key　1	http　server　location

proxy_cache_valid

proxy_cache

Nginx

1　X-Accel-Expires　60s　0

“ X-Accel-Expires:60 ”

2　Cache-Control　Expires　X-Accel-Expires

60s　0　“ Cache-Control:60 ”

3　proxy_cache_valid　10s

“ proxy_cache_valid 10s; ”

Set-Cookie　Vary

*　Set-Cookie　Vary

指令：proxy_ignore_headers

proxy_ignore_headers field ...;

http server location

X-Accel-Redirect X-Accel-Expires X-Accel-Limit-Rate X-Accel-Buffering X-Accel-Charset Expires Cache-Control Set-Cookie Vary Set-Cookie Vary

```
proxy_ignore_headers    Vary    Set-Cookie ;
```

proxy_cache 5-2

表 5-2 不使用缓存的配置指令

指 令	用 途	支持配置的环境
proxy_no_cache	a b 0 proxy_no_cache $arg_a $arg_b	http server location
proxy_cache_bypass	Cookie dd 0 proxy_cache_bypass $cookie_dd	http server location

Nginx I/O

“ ” “ ”

5.3.1 缓存未命中的最佳实践

- URL

-

5xx　500　502　503　504　ngx_http_proxy_module

proxy_cache_lock

指令：proxy_cache_lock

proxy_cache_lock on | off;

proxy_cache_lock off;

http　server　location

on　URL　Nginx　Nginx　proxy_cache_lock_timeout

指令：proxy_cache_lock_timeout

proxy_cache_lock_timeout time;

proxy_cache_lock_timeout 5s;

http　server　location

proxy_cache_lock

URL

指令：proxy_cache_background_update

proxy_cache_background_update on | off;

proxy_cache_background_update off;

http server location

proxy_cache_use_stale updating

指令：proxy_cache_use_stale

proxy_cache_use_stale error | timeout | invalid_header | updating | http_500 | http_502 | http_503 | http_504 | http_403 | http_404 | http_429 | off ...;

proxy_cache_use_stale off;

http server location

“ ”

500 timeout proxy_cache_use_stale updating

```
   location / {
       proxy_cache cachedata;
       proxy_cache_valid        200 304  5m;
       proxy_cache_valid        301 302 2m;

       proxy_cache_key $scheme$host$uri$is_args$args;

       proxy_set_header         Host $host;
       proxy_set_header         X-Real-IP $remote_addr;
       proxy_set_header         X-Forwarded-For $proxy_add_x_forwarded_for;
       #
       proxy_cache_background_update on;
       #
       proxy_cache_use_stale     error   timeout   invalid_header   updating
http_502  http_503   http_504;
       #
       #
       proxy_cache_lock on;
       #
       #
```

```
        proxy_cache_lock_timeout 20;

        proxy_ignore_headers    Vary    Set-Cookie ;
        proxy_pass http://test_servers;
    }
```

5.1　　$upstream_cache_status

$upstream_cache_status

5-3

表 5-3　$upstream_cache_status 状态值和标识说明

状　态　值	标　　识
MISS	
BYPASS	proxy_cache_bypass
EXPIRED	
STALE	proxy_cache_use_stale
REVALIDATED	proxy_cache_revalidate　　if-modified-since
UPDATING	proxy_cache_background_update　　on
HIT	

5.3.2　横向扩展最佳实践

URL　　I/O

key　Nginx　key　　1MB　　8000　key

key

500MB　　400　　key

value　proxy_cache　　I/O

proxy_cache_path　　proxy_cache data

Nginx　　split_clients

http{}

```
    http {
    # proxy_cache_path          $disk
```

```
   proxy_cache_path   /data1/nginxcache   levels=1:2   keys_zone=data_1:100m
inactive=7d max_size=1000g use_temp_path=off;
   proxy_cache_path   /data2/nginxcache   levels=1:2   keys_zone=data_2:100m
inactive=7d max_size=1000g use_temp_path=off;
   proxy_cache_path   /data3/nginxcache   levels=1:2   keys_zone=data_3:100m
inactive=7d max_size=1000g use_temp_path=off;
   proxy_cache_path   /data4/nginxcache   levels=1:2   keys_zone=data_4:100m
inactive=7d max_size=1000g use_temp_path=off;

   #     URL     hash                      hash                          $disk
   split_clients $request_uri $disk {
       20%     1;
       20%     2;
       30%     3;
       *       4;
   }

   }
```

proxy_cache

```
   proxy_cache data_$disk;
```

I/O

5.3.3 避免硬盘 I/O 阻塞

Nginx worker Nginx

1.7.11 Nginx

thread_pool main error.log worker_processes

“ thread_pool default threads=32 max_queue=65536; ”

```
   #
   # thread_pool
   thread_pool pool_1 threads=32 max_queue=65536;
   thread_pool pool_2 threads=32 max_queue=65536;
   thread_pool pool_3 threads=32 max_queue=65536;
   thread_pool pool_4 threads=32 max_queue=65536;
```

proxy_cache

```
   proxy_cache data_$disk;
   aio threads=pool_$disk;
```

Nginx

注意：　　gzip　　I/O　　I/O

5.3.4　集群模式

Nginx　hash　　proxy_cache

proxy_cache　　Nginx

```
upstream proxy_cache_servers  {
    hash $request_uri;
    server 172.18.1.5:8083  weight=20 max_fails=10 fail_timeout=30s;
    server 172.18.1.6:8083  weight=20 max_fails=10 fail_timeout=30s;
    server 172.18.1.7:8083  weight=20 max_fails=10 fail_timeout=30s;
    keepalive 300;
}
```

proxy_cache　　HTTP

hash

注意：　　hash

proxy_cache_key　　key　　URL

Cookie

1．多级缓存

proxy_cache　　URL

varnish　　varnish

varnish　　proxy_cache

2．硬盘同步备份数据

I/O

rsync

I/O

3．使用网络 I/O

I/O I/O

Nginx

Lua-Nginx-Module

I/O I/O

I/O CPU

```
user  webuser webuser;
worker_processes  8;

thread_pool pool_1 threads=32;
thread_pool pool_2 threads=32;
thread_pool pool_3 threads=32;
thread_pool pool_4 threads=32;

error_log  logs/error.log  error;

pid /data/nginx.pid;

worker_rlimit_nofile 65535;
```

```
    events {
        worker_connections  65535;
        multi_accept on;
        use epoll;
    }

    http {

        include       mime.types;
        default_type  application/octet-stream;

        log_format  main  '$remote_addr - $remote_user [$time_local] "$request"
"$status $body_bytes_sent "$http_referer" "disk:data_$disk" "cache_status:$upstream_
cache_status"';

        access_log /data/access.log main;

        keepalive_requests 1000;
        keepalive_timeout 60;
        client_max_body_size 300m;
        client_body_buffer_size 512k;
        reset_timedout_connection on;

        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;


        gzip on;
        gzip_min_length  1k;
        gzip_buffers     16 8k;
        gzip_comp_level 7;
        gzip_types        text/plain text/css application/x-javascript text/
xml  application/xml  application/xml+rss  text/javascript  application/json
application/javascript;
        gzip_vary on;

        proxy_connect_timeout   5;
        proxy_send_timeout      30;
        proxy_read_timeout      60;
        proxy_buffering         on;
        proxy_next_upstream error http_500 http_502  http_504 timeout;
        proxy_next_upstream_tries 2;
        proxy_next_upstream_timeout 0;
```

```
       upstream test_servers {
          server 127.0.0.1:81  max_fails=5 fail_timeout=10s weight=10;
          server 127.0.0.1:82  max_fails=5 fail_timeout=10s weight=10;
          keepalive 100;

      }

       proxy_cache_path  /data1/nginxcache  levels=1:2  keys_zone=data_1:100m
inactive=7d max_size=1000g use_temp_path=off;
       proxy_cache_path  /data2/nginxcache  levels=1:2  keys_zone=data_2:100m
inactive=7d max_size=1000g use_temp_path=off;
       proxy_cache_path  /data3/nginxcache  levels=1:2  keys_zone=data_3:100m
inactive=7d max_size=1000g use_temp_path=off;
       proxy_cache_path  /data4/nginxcache  levels=1:2  keys_zone=data_4:100m
inactive=7d max_size=1000g use_temp_path=off;

       split_clients $request_uri $disk {
           20%    1;
           20%    2;
           30%    3;
           *   4;
       }

       server {
           listen       80;
           location / {
               proxy_cache data_$disk;
               aio threads=pool_$disk;
               proxy_cache_valid        200 304  5m;
               proxy_cache_valid        301 302 2m;

               proxy_cache_key $scheme$host$uri$is_args$args;

               proxy_set_header Host $host;
               proxy_set_header X-Real-IP $remote_addr;
               proxy_set_header X-Forwarded-For $proxy_add_x_forwar ded_for;
               proxy_http_version 1.1;
               proxy_set_header Connection "";

               proxy_cache_background_update on;
               proxy_cache_use_stale  error timeout invalid_header updating
http_502  http_503   http_504;
```

```
            proxy_cache_lock on;
            proxy_cache_lock_timeout 10;

            proxy_ignore_headers   Vary    Set-Cookie ;
            proxy_pass http://test_servers;
        }
    }
}
```

Nginx　proxy_cache　10　I/O　I/O　CPU

6

2 5 Nginx

“ ”

Nginx

1 Nginx if…elseif…else if

2 Nginx

3 Nginx Nginx

4 Nginx
Nginx

5 Nginx API

6 Nginx

7 Nginx C Nginx

2013　　Lua　　Nginx
Lua　Nginx

1　Tengine　　Web　　2011　　Nginx+Lua

2　　Nginx　　OpenResty　　Lua-Nginx-Module
Nginx　　Lua-Nginx-Module　　Nginx

3　Lua　　1~2　　Nginx

4　Lua　　Nginx
Nginx　　Lua　“　　”

注意：　　Lua　　Lua-Nginx-Module　　Lua
Lua

Lua　　Pontifical Catholic University of Rio de Janeiro
Luiz Henrique de Figueiredo　Roberto Ierusalimschy　Waldemar Celes
1993

- Lua　　C
- Lua　　C　C++
- Lua 5.1　　200KB

LuaJIT　　C　　Lua　　Lua 5.1　　Lua
LuaJIT　　LuaJIT　Lua　　Just-In-Time　　CPU
Lua　　Lua
LuaJIT

LuaJIT　　LuaJIT 2.0.5　LuaJIT 2.1.0-beta3

LuaJIT 2.1.0-beta3 Lua-Nginx-Module LuaJIT 2.1.0-beta3 Nginx+Lua LuaJIT 2.1.0-beta3

LuaJIT 2.1.0-beta3

```
# wget -S https://codeload.github.com/openresty/luajit2/tar.gz/v2.1-20181029 -O LuaJIT-OpenResty-2.1.0-beta3.tar.gz
# tar -zxvf LuaJIT-OpenResty-2.1.0-beta3.tar.gz
# cd luajit2-2.1-20181029/
# make && make install
```

lib /etc/profile

```
# vim /etc/profile
export LUAJIT_LIB=/usr/local/lib
export LUAJIT_INC=/usr/local/include/luajit-2.1
# source /etc/profile
```

```
# luajit -v
LuaJIT 2.1.0-beta3 -- Copyright (C) 2005-2017 Mike Pall. http://luajit.org/
```

LuaJIT

“ Hello, World ”

```
# echo 'print("Hello, World")' >test.lua
# cat test.lua
print("Hello, World")
# luajit test.lua
Hello, World
```

注意： Lua -- # Shell Python Nginx

Lua Lua 8

nil boolean number string table function thread userdata

6.4.1 类型说明

Lua 6-1

表 6-1　Lua 的数据类型说明

数据类型	描述说明
nil	nil　false
boolean	true/false
number	LuaJIT　number　long int　long long int Lua　number
string	[[]]
table	table　nil　Lua
function	Lua　C
thread	Lua
userdata	C　Lua

tpye

```
print(type("test lua"))          -->      string
print(type(233+2))               -->      number
print(type(true))                -->      boolean
print(type(nil))                 -->      nil
print(type(type))                -->      function
```

6.4.2　类型示例

8　Linux　luajit -i

```
# luajit -i
```

- nil

```
> print(x)
nil
> print(type(a))
nil
```

x　a　nil

- boolean

test.lua

```
local   str   =    ''
local   n     =   nil

if      str     then
```

```
    print('str')
else
    print("not str")
end

if      n       then
    print(n)
else
    print("not n")
end
```

```
# luajit test.lua
str
not n
```

- number

```
local a = 3
local b = '3'
print(type(a))    -->      number
print(type(b))    -->      string
```

LuaJIT long long int Lua 5.1 Lua long long int

```
# luajit -i
LuaJIT 2.1.0-beta3 -- Copyright (C) 2005-2017 Mike Pall.
http://luajit.org/
JIT: ON SSE2 SSE3 SSE4.1 BMI2 fold cse dce fwd dse narrow loop abc sink
fuse
> print(922337203685423420 1LL-1)
922337203685423420 0LL

# lua
Lua 5.1.4  Copyright (C) 1994-2008 Lua.org, PUC-Rio
> print(922337203685423420 1LL-1)
stdin:1: malformed number near '922337203685423420 1LL'
```

- string

[[]] [[]]

```
# vim test.lua
str1 = "str1"
str2 = 'str2'
str3 = [[
-----------------
this
is
str3
-----------------
]]

print(str1)
print(str2)
print(str3)
```

```
# luajit test.lua
str1
str2
-----------------
this
is
str3
-----------------
```

Lua　　number

```
# luajit
LuaJIT 2.1.0-beta3 -- Copyright (C) 2005-2017 Mike Pall.
http://luajit.org/
JIT: ON SSE2 SSE3 SSE4.1 BMI2 fold cse dce fwd dse narrow loop abc sink
fuse
> print("45" - 2)
43
> print("3" * "2")
6
> print("a" + 1)
stdin:1: attempt to perform arithmetic on a string value
stack traceback:
    stdin:1: in main chunk
    [C]: at 0x00404180
> print("a22" + 1)
stdin:1: attempt to perform arithmetic on a string value
```

```
stack traceback:
    stdin:1: in main chunk
    [C]: at 0x00404180
```

- table

```
--             table
local n_table = {}

--                        table
local n_table  = {"apple", "pear", "orange", "grape"}
```

“ ” table

```
local   n_table =
{
test =  "testnginx", --
city = {"shanghai","wuhan","chengdu","beijing"},
                      --                    table

"sada",               --                                          1
[17]    =  360,       --                            17
12312                 --                                          2
}

print(n_table.test)     -->    testnginx
print(n_table.city[2])  -->    wuhan
print(n_table[1])       -->    sada
print(n_table[2])       -->    12312
print(n_table[17])      -->    360
```

注意：Lua table 1 0
table 6.10

- function

```
# vim test.lua
function res(st)
    if st == 'test' then
        return 'this is test'
```

```
    else
        return 'this is not test'
    end
end
print(res("test"))  --
local a = res       --                          local
print(a("123"))     --     a
```

```
# luajit test.lua
this is test
this is not test
```

- thread

Lua　　coroutine　　thread

　　yield

- userdata

　　Lua　　C

Lua　　C

6.5.1　算术运算符

　　6-2

表 6-2　算术运算符及其说明

算术运算符	说　　明
+	
-	
*	
/	
%	
^	

```
# vim test.lua
print("    1+2=", 1+2 )
print("    1-2=", 1-2 )
print("    1*2=", 1*2 )
print("    1/2=", 1/2 )
print("    1%2=", 1%2 )
print("    2^2=", 2^2 )
```

Lua

```
# luajit test.lua
    1+2= 3
    1-2= -1
    1*2= 2
    1/2= 0.5
    1%2= 1
    2^2= 4
```

6.5.2 关系运算符

6-3

表 6-3 关系运算符及其说明

关系运算符	说　明
>	
<	
>=	
<=	
~=	
==	

```
# vim test.lua
a = 14
b = 7

if( a == b )
then
    print("a      b" )
else
    print("a       b" )
```

```
end

print('a>b',a>b)
print('a~=b',a~=b)
print('a<b',a<b)
```

```
# luajit test.lua
a       b
a>b true
a~=b true
a<b false
```

true　false

6.5.3　逻辑运算符

6-4

表 6-4　逻辑运算符及其说明

逻辑运算符	说　明
and	
or	
not	

```
# vim test.lua
a = "test"
b = nil
c = ''  --              nil

if (a and b) then
    print("a and b is true")
else
    print("a and b is false")
end

if (a or b)  then
    print("a or b is true")
else
    print("a or b is false")
end
```

```
if ( not( a and b) )  then
    print("not( a and b) is true")
else
    print("not( a and b) is false")

end

print('a and c : ',a and c)
print('a or c : ',a or c )
print('not(a and c): ',not(a and c))
```

```
a and b is false
a or b is true
not( a and b) is true
a and c :     --               c
a or c : test
not(a and c): false
```

- a and b　　a　false　　a　　b
- a or b　　a　true　　a　　b
- not(a and b)　　a and b　true　　false

6.5.4　字符串连接和字符串长度计算

Lua　　..

```
# vim test.lua
print("a" .. "b")
print(1 .. "x")
print(1 .. 2)
```

```
luajit test.lua
ab
1x
12
```

注意：

Lua　　#

```
# vim test.lua
print(#("a" .. "b"))
print(#"sxjk19dasdas")
print(#"")
print(#"12 12 12")
local t1 = {"a",  "c" , "b" , nil}  --      table
print(#t1)
```

```
# luajit test.lua
2           --
12          --
0           --          0
8           --
3           --                          nil
```

6.5.5　运算符优先级

1　^

2　not　　#　　-　　“ - ”

3　*　/　%

4　+　-　　“ - ”

5　..

6　<　>=　==　~=

7　and

8　or

nil　　Lua

3

6.6.1 全局变量

local

table

table table CPU

6.6.2 局部变量

local

1 “ ”

2 table

3

```
# vim test.lua
function test()
    local a = 'local var'
         b = 'global'
    print('test() output:',a)
end

test()  --     test

print('a:',a)
print('b',b)
```

```
# luajit test.lua
test() output: local var   --   test
a: nil          -- a                test              a                nil
b: global       -- b
```

6.6.3 变量赋值

```
local a = 1
```

```
local a = a + 2
local b = a .. "1"  --

print("a:" ,a)      --       3
print("b:" ,b)      --      31
```

Lua

```
# vim test.lua
local a, b, c = 1, 'test', 'x'
print("a:", a)
print("b:", b)
a, b = b, a  --
print("a:", a)
print("b:", b)
```

```
# luajit test.lua
a: 1
b: test
a: test
b: 1
```

nil

6.7.1　if-else

if-else

```
# vim test.lua
local a = 70
if (a > 100) then
  print(a .. '>100')
elseif ( a <= 100 and  a >= 50)  then
  print(a,'Between 50 and 100')
else
  print(a .. '<50')
end
```

```
# luajit test.lua
70 Between 50 and 100
```

6.7.2 for 循环

for

```
for var=exp1,exp2,exp3 do
    something
end
```

for var exp1 exp2 exp3 exp3
1

```
# vim test.lua
for i=5,1,-1 do
    print(i)
end
```

```
# luajit test.lua
5
4
3
2
1
```

for Lua

```
# vim test.lua
local a = {'1', '2', '3', '4', 'test', 'test1'}

for i,v in ipairs(a) do
 print("index:" .. i , "value:" .. v)
end
```

index value

```
# luajit test.lua
index:1 value:1
index:2 value:2
index:3 value:3
index:4 value:4
index:5 value:test
index:6 value:test1
```

6.7.3　while 循环

while　　false

```
# vim test.lua
local a=6
while( a < 10 )
do
    a = a+1
    print("a:", a)
end
```

```
# luajit test.lua
a: 7
a: 8
a: 9
a: 10  -- 10<10   false
```

6.7.4　break 和 return

break

```
# vim test.lua

local a=6
while( a < 12 )
do
    a = a+1
    print("a:", a)
    if (a == 9) then
       print("it is:",a)
       break
    end
end
```

```
# luajit test.lua
a: 7
a: 8
a: 9
it is: 9  --    a     9
```

return　　return

return

```
# vim test.lua

local function sum_num(a)

    local  b = a + 1
    if (b == 3)  then
        return a .. "---match"
    else
        return "no match"
    end
   print('123')

end

local num = sum_num(2)
print(num)
```

```
# luajit test.lua
2---match
```

print('123') return

6.4 Lua

6.8.1 函数格式

function

```
local function function_name(argument1,argument2,argument3...)
    function_body
    return function_result
end
```

- function function_name function

local

- (argument1,argument2,argument3...)　　Lua

- function_body
- return function_result

```
# vim test.lua
local function sum_num(a,b)
    local c = a - b
    local d = a + b
    return c,d
end

local c,d = sum_num(2,1)
print(c,d)
```

```
# luajit test.lua
1 3
```

6.8.2　传参方式

```
# vim test.lua

local function sum_num()
    local a = 2
    local b = 1
    local c = a - b
    local d = a + b
    return c,d
end

local c,d = sum_num()  --
print(c,d)  --         1  3
```

…

```
# vim test.lua
```

```
local function sum_num(...)
    local a = "a"
    for i, v in ipairs{...} do
        a = a .. v
    end
    return a
end
local a = sum_num("b","c","d","1")
print(a)
```

```
# luajit test.lua
abcd1
```

6.8.3 函数的创建位置

nil

```
# vim test.lua
--     5.7.1
local c,d = sum_num(2,1)
print(c,d)

local function sum_num(a,b)  --
    local c = a - b
    local d = a + b
    return c,d
end
```

```
# luajit test.lua
luajit: test.lua:1: attempt to call global 'sum_num' (a nil value)
stack traceback:
    test.lua:1: in main chunk
    [C]: at 0x00404180
```

Lua 5.1 Lua table

6.9.1　模块格式

Lua　　table　table

table

```
# vim md.lua
--
local m = {}
--
m.str1 = "a"
--
local function func_local()
    print("Are you ok!")
end
--
function m.func()
    print("I am here !")
    func_local()
end
return m
```

6.9.2　加载模块

require

```
require("     ")
```

- 　　.lua
-

```
# luajit test.lua
luajit: test.lua:1: module 'mdx' not found:
    no field package.preload['mdx']
    no file './mdx.lua'
    no file '/usr/local/share/luajit-2.1.0/mdx.lua'
    no file '/usr/local/share/lua/5.1/mdx.lua'
    no file '/usr/local/share/lua/5.1/mdx/init.lua'
    no file './mdx.so'
    no file '/usr/local/lib/lua/5.1/mdx.so'
    no file '/usr/local/lib/lua/5.1/loadall.so'
```

```
stack traceback:
    [C]: in function 'require'
    test.lua:1: in main chunk
    [C]: at 0x00404180
```

6.9.1 md.lua

表 6-5　table 常见的操作

常见操作	使用说明
local tabel = {}	table
table.insert(table,value)	table
table.insert(table,pos,value)	pos Lua 1
table.getn(table)	table #
table.concat(table, sep, start, end)	table sep sep start end start end start end table
table.maxn (table)	table
table.remove (table)	
table.remove (table, pos)	pos
table.sort(table)	
table.sort(table, func)	func func local func_sort = function(a, b) return b < a end

```
local a = {1, 8}
print(table.getn(a))      --       2
table.insert(a, 1, 5)     --          1         5    table    1           5
table.insert(a,123124)    --          123124
print(table.concat(a, ","))   --          a               5,1,8,123124
print(table.getn(a))          --       4
print(table.maxn(a))          --              4
print(#a)                     --       4
--table.remove(a,2)
table.sort(a)                 --          a
print(table.concat(a, ","))   --          1,5,8,123124

                              --
local func_sort = function(a, b) return b < a end
table.sort(a,func_sort)
print(table.concat(a, ",",1,3))   --          1   3      123124,8,5
```

6.10.2　定义字符串

''　　" "

[[]]　　[[]]

[]　　[=[]=]

```
local a = [["a","d","s"]]
```

```
print(a)
--
local b = [=[[[0-9]+
"x a"\n \\  ]]] [[]] [[]]  ]]=]
print(b)
```

```
"a","d","s"
[[0-9]+
"x a"\n \\  ]]] [[]] [[]]  ]
```

6.10.3 字符串连接

Lua .. ..

table.concat

```
local tabel_s = {};
for i = 1,10, 1 do
    tabel_s [i] = "a";
end
print(table.concat(tabel_s,","))  --          a,a,a,a,a,a,a,a,a,a
```

Lua 2013

URL a b 6-8 Cookie

a

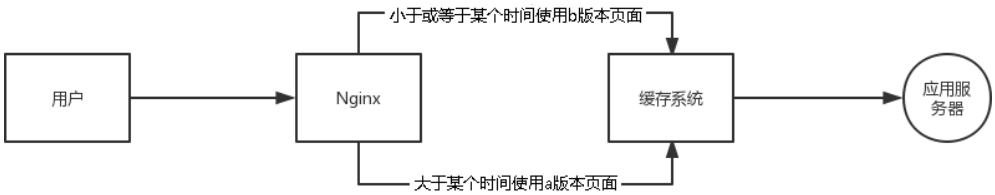
b URL Cookie

Cookie

Nginx

version a version b

key URL+version



6-1 URL a b

Nginx　　Nginx

Lua　　Lua

```
location  = /ab {
            -- Lua
        rewrite_by_lua '
            --
            local ngx = require "ngx"
            --        Cookie   frist_time                    0
            local frist_time = tonumber(ngx.var.cookie_frist_time) or 0
            if frist_time > 12392 then
            --     12392         vesion   a
                 ngx.req.set_header("Version", "a")
            else
            --          12392         vesion   b
                 ngx.req.set_header("Version", "b")

            end
        ';
            -- Lua
        proxy_pass http://cache_servers;
}
```

Nginx+Lua

Lua　　Lua

Lua-Nginx-Module

7

6 Lua LuaJIT Lua-Nginx-Module Nginx

注意： Lua-Nginx-Module 0.10.13 Nginx API for Lua Lua API Lua-Nginx-Module Ngx_Lua Lua API “ ? ” ngx.req.get_headers (max_headers?, raw?) max_headers raw

OpenResty Nginx Lua Web Web Web

Nginx Ngx_Lua OpenResty Nginx OpenResty

1．使用 Nginx 编译 Ngx_Lua 的场景

HTTP

Nginx OpenResty Nginx

2．OpenResty 的使用场景

API

Web

Nginx　　Lua　　OpenResty
OpenResty

LuaJIT 2.1.0-beta3　　6.2　　ngx_devel_kit

Nginx　　OpenResty

```
# wget 'http://nginx.org/download/nginx-1.12.2.tar.gz'
# git clone https://github.com/simplresty/ngx_devel_kit.git
# git clone https://github.com/openresty/lua-nginx-module.git
# tar -xzvf nginx-1.12.2.tar.gz
# cd nginx-1.12.2/
# ./configure --prefix=/usr/local/nginx_1.12.2 \
     --add-module=../ngx_devel_kit \
     --add-module=../lua-nginx-module
     --with-ld-opt="-Wl,-rpath,$LUAJIT_LIB"

# make && make install
```

Nginx　　Ngx_Lua　　Ngx_Lua　　Nginx

```
1.13.x (last tested: 1.13.6)
1.12.x
1.11.x (last tested: 1.11.2)
1.10.x
1.9.x (last tested: 1.9.15)
1.8.x
1.7.x (last tested: 1.7.10)
1.6.x
```

https://github.com/openresty/lua-nginx-module#
nginx-compatibility

Ngx_Lua API Nginx

指令：ngx.var.VARIABLE

ngx.var.VAR_NAME

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* ody_filter_by_lua* log_by_lua*

Context Ngx_Lua

Ngx_Lua

“ Hello, World! ” Nginx server

```
server {
    listen       80;
    server_name  testnginx.com;
    charset koi8-r;
    location = /test {
     #               MIME-Type,          Content-Type:text/plain
     default_type 'text/plain';
     -- content_by_lua_block
     content_by_lua_block {
         ngx.say('Hello,World!')
     }
    }
}
```

server

```
#  curl -I http://testnginx.com/test
Hello,World!
```

ngx.say

content_by_lua_block HTTP

8.6

Nginx　　Lua

```
lua_package_path "${prefix}conf/lua_modules/?.lua;/opt/lua/?.lua;;";
```

注意：“ ;; ” LuaJIT

7.6.2 定义 C 模块的搜索路径

lua_package_cpath C http lua_package_path

```
lua_package_cpath "${prefix}conf/c_md/?.so;/opt/c/?.so;;";
```

2 Nginx Nginx ngx.var.VARIABLE

ngx.var.VAR_NAME

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* og_by_lua*

/ Nginx HTTP Nginx URL Nginx $1 $2 ngx.var[1] ngx.var[2]

```
server {
    listen       80;
    server_name  testnginx.com;
    location ~ ^/([a-z]+)/var.html {
        set $a '';
        set $b '';
        set $c '';
        set $d '';
        rewrite_by_lua_block {
           local ngx = require "ngx"
           --   1         a
           ngx.var.a = '1'
           --     HTTP        User_Agent              b
           ngx.var.b = ngx.var.http_user_agent
           --        test            c
```

```
            ngx.var.c = ngx.var.arg_test
            --     location               $1              d
            ngx.var.d = ngx.var[1]
        }
        echo $a;
        echo $b;
        echo $c;
        echo $d;
    }
```

```
# curl -i 'http://testnginx.com/nginx/var.html?test=12132&a=2&b=c&dd'
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Thu, 07 Jun 2018 07:22:32 GMT
Content-Type: text/html
Transfer-Encoding: chunked
Connection: keep-alive
1
curl/7.19.7  (x86_64-redhat-linux-gnu)  libcurl/7.19.7  NSS/3.19.1  Basic
ECC zlib/1.2.3 libidn/1.18 libssh2/1.4.2
12132
nginx
```

Nginx　　Lua

$query_string　$arg_PARAMETER　$http_NAME

4.1　　Nginx　　Lua API

7.8.1　添加请求头

指令：ngx.req.set_header

ngx.req.set_header(header_name, header_value)

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*

HTTP

Test_Ngx_Ver 1.12.2

```
ngx.req.set_header("Test_Ngx_Ver", "1.12.2")
```

ngx.req.set_header

```
ngx.req.set_header("Test", {"1", "2"})
```

```
Test: 1
Test: 2
```

7.8.2 清除请求头

指令：ngx.req.clear_header

ngx.req.clear_header(header_name)

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua*

```
ngx.req.clear_header("Test_Ngx_Ver")
```

```
ngx.req.set_header("Test_Ngx_Ver", nil)
```

7.8.3 获取请求头

指令：ngx.req.get_headers

headers = ngx.req.get_headers(max_headers?, raw?)

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua*

Lua table

```
server {
    listen       80;
    server_name  testnginx.com;
```

```
        location  / {

           content_by_lua_block {
              local ngx = require "ngx";
              local h = ngx.req.get_headers()
              for k, v in pairs(h) do
                  ngx.say('Header name: ',k,' value:',v)
              end
              --       table
              ngx.say(h["host"])
           }
       }
   }
```

```
   # curl -i 'http://testnginx.com/test?=12132&a=2&b=c&dd'
   HTTP/1.1 200 OK
   Server: nginx/1.12.2
   Date: Fri, 08 Jun 2018 07:46:38 GMT
   Content-Type: application/octet-stream
   Transfer-Encoding: chunked
   Connection: keep-alive

   Header name:host value: testnginx.com
   Header name:accept value: */*
   Header  name:user-agent  value:  curl/7.19.7  (x86_64-redhat-linux-gnu)
libcurl/7.19.7 NSS/3.19.1 Basic ECC zlib/1.2.3 libidn/1.18 libssh2/1.4.2
   testnginx.com
```

HTTP　　CDN　　set-cookie　　Lua　API

7.9.1　获取响应头

指令：ngx.resp.get_headers

headers = ngx.resp.get_headers(max_headers?, raw?)

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_

by_lua*　body_filter_by_lua*　log_by_lua*　balancer_by_lua*

Lua　table

```
server {
    listen       80;
    server_name  testnginx.com;
     location  / {
        content_by_lua_block {
           local ngx = require "ngx";
           local h = ngx.resp.get_headers()
           for k, v in pairs(h) do
               ngx.say('Header name:',k,' value: ',v)
           end
           --       table
           ngx.say(h["content-type"])
        }
    }
}
```

```
# curl -i 'ttp://testnginx.com/test?=12132&a=2&b=c&dd'
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Fri, 08 Jun 2018 07:36:35 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

Header name:content-type value: application/octet-stream
Header name:connection value: keep-alive
application/octet-stream
```

7.9.2　修改响应头

指令：ngx.header.HEADER

ngx.header.HEADER = VALUE

value = ngx.header.HEADER

rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　og_by_lua*

API　　“_”　　“-”

```
server {
    listen       80;
    server_name  testnginx.com;

    location / {
        content_by_lua_block {
            local ngx = require "ngx"
            ngx.header.content_type = 'text/plain';
            --          “_”          “-”
            ngx.header.Test_Nginx = 'Lua';
            --            ngx.header.A_Ver = 'aaa'
            ngx.header["A_Ver"] = 'aaa';
            --                    a
            local a = ngx.header.Test_Nginx;
        }

    }
}
```

“_”　　“-”

```
# curl -i 'http://testnginx.com/?test=12132&a=2&b=c&dd'
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Fri, 08 Jun 2018 03:18:16 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive
Test-Nginx: Lua
A-Ver: aaa
```

/test　　set-cookie

Cookie

```
location = /test {
    content_by_lua_block {
       local ngx = require "ngx"
       --          Cookie
       ngx.header['Set-Cookie']   =   {'test1=1;   path=/test',   'test2=2;
path=/test'}
```

```
        }
    }
```

```
    # curl -i 'http://testnginx.com/test?=12132&a=2&b=c&dd'
    HTTP/1.1 200 OK
    Server: nginx/1.12.2
    Date: Fri, 08 Jun 2018 03:21:59 GMT
    Content-Type: application/octet-stream
    Transfer-Encoding: chunked
    Connection: keep-alive
    Set-Cookie: test1=1; path=/test
    Set-Cookie: test2=2; path=/test
```

7.9.3 清除响应头

nil

```
    ngx.header["X-Test"] = nil;
```

$request_body proxy_pass fastcgi_pass uwsgi_pass scgi_pass Nginx Lua ngx.var.request_body

7.10.1 强制获取请求体

指令：lua_need_request_body

lua_need_request_body <on|off>

off

http server location location if

off on ngx.var.request_body $request_body Nginx client_body_buffer_size Nginx $request_body

client_body_buffer_size　client_max_body_size

Ngx_Lua

7.10.2　用同步非阻塞方式获取请求体

指令：ngx.req.read_body

ngx.req.read_body()

rewrite_by_lua*　access_by_lua*　content_by_lua*

Nginx

ngx.req.get_body_data

ngx.req.get_body_file

指令：ngx.req.get_body_data

data = ngx.req.get_body_data()

rewrite_by_lua*　access_by_lua*　content_by_lua*　log_by_lua*

ngx.req.read_body

Lua　Lua　table

ngx.req.get_post_args

指令：ngx.req.get_post_args

args, err = ngx.req.get_post_args(max_args?)

rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*

ngx.req.read_body

POST　Lua　table　max_args

100

max_args　0

max_args　0　10　ngx.req.get_post_args(10)

指令：ngx.req.get_body_file

file_name = ngx.req.get_body_file()

rewrite_by_lua*　access_by_lua*　content_by_lua*

ngx.req.read_body

nil

Nginx

7.10.3　使用场景示例

1．获取 string 类型的请求体

string

```
   server {
       listen       80;
       server_name  testnginx.com;
       location / {
         client_max_body_size 10k;
         client_body_buffer_size 1k;

         content_by_lua_block {
          local ngx = require "ngx"
            --
            ngx.req.read_body()
            --
            local data = ngx.req.get_body_data()
            if data then
                ngx.print('ngx.req.get_body_data: ',data, ' ---- type is ',
type(data))
                return
            else
            --
                local file = ngx.req.get_body_file()
                if file then
                    ngx.say("body is in file ", file)
                else
                    ngx.say("no body found")
                end
            end
         }
       }
```

Nginx　　HUP　　reload

1KB　　Nginx　　client_body_buffer_size　1k

string

```
# curl -i http://testnginx.com/ -d 'test=12132&a=2&b=c&dd'
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Wed, 06 Jun 2018 11:03:35 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

ngx.req.get_body_data: test=12132&a=2&b=c&dd ---- type is string
```

2．获取 table 类型的请求体

table

```
server {
    listen       80;
    server_name  testnginx.com;

    location / {
      client_max_body_size 10k;
      client_body_buffer_size 1k;

      content_by_lua_block {
         --
         ngx.req.read_body()
         --                              Lua   table
         local args, err = ngx.req.get_post_args()
         if args then
            for k, v in pairs(args) do
                if type(v) == "table" then
                    --
                    ngx.say(k, ": ", table.concat(v, ", "))
                else
                    ngx.say(k, ": ", v)
                end
             end
         else
             --
             local file = ngx.req.get_body_file()
             if file then
                 ngx.say("body is in file ", file)
             else
```

```
                    ngx.say("no body found")
                end
            end
        }
    }
}
```

a 2 c d

```
#  curl -i http://testnginx.com/ -d 'test=12132&a=2&b=c&dd=1&a=354&c=&d'

b: c
dd: 1
d: true
c:
test: 12132
a: 2, 354
```

a c d

true

3．获取临时文件中的请求体

1KB~10KB

```
# curl -i http://testnginx.com/ -d 'test=12132&a=2&b=kls204120312saldkk12
easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12i3j12io
3jeioq2jesklS204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312s
aldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12
i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2je2041203
12saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3e
j12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS
204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdi
asasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12i3j12io3jeioq
2je204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12eas
jdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12i3j12io3je
ioq2jesklS204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312sald
kk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12i3j
12io3jeioq2je204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312s
aldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej12
i3j12io3jeioq2jesk20312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312
saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120312saldkk12easjdiasasd3ej1
2i3j12io3jeioq2je204120312saldkk12easjdiasasd3ej12i3j12io3jeioq2jesklS204120
312saldkk12easjdiasasd3ej11'
```

```
HTTP/1.1 100 Continue

HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Wed, 06 Jun 2018 10:14:32 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

body is in file /usr/local/nginx_1.12.2/client_body_temp/0000000051
```

client_body_buffer_size

注意：

io.open

7.10.4　使用建议

- lua_need_request_body
- ngx.req.read_body()
- client_body_buffer_size　client_max_body_size
  Nginx
  Lua
- Nginx
- ngx.req.get_post_args

Lua　ngx.print　ngx.say

7.11.1　异步发送响应体

指令：ngx.print

ok, err = ngx.print(...)

rewrite_by_lua*　access_by_lua*　content_by_lua*

```
location  / {

    content_by_lua_block {
        local ngx = require "ngx";
        local h = ngx.req.get_headers()
        for k, v in pairs(h) do
            ngx.print('Header name: ',k,' value: ',v)
        end

    }
}
```

```
# curl -i 'http://testnginx.com/test?=12132&a=2&b=c&dd'
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Fri, 08 Jun 2018 08:11:40 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

Header name:host value: testnginx.comHeader name:accept value: */*Header name:user-agent value: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.19.1 Basic ECC zlib/1.2.3 libidn/1.18 libssh2/1.4.
```

指令：ngx.say

ok, err = ngx.say(...)

rewrite_by_lua*　access_by_lua*　content_by_lua*

ngx.print　　1

7.11.2　同步发送响应体

ngx.print　　ngx.say
ngx.flush(true)

指令：ngx.flush

ok, err = ngx.flush(wait?)

rewrite_by_lua*　access_by_lua*　content_by_lua*

wait　　true

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';
    location /test1 {
        content_by_lua_block {
           ngx.say("test ")
           ngx.say("nginx ")
           ngx.sleep(3)
           ngx.say("ok!")
           ngx.say("666!")
        }
    }

    location /test2 {
        content_by_lua_block {
           ngx.say("test ")
           ngx.say("nginx ")
           ngx.flush(true)
           ngx.sleep(3)
           ngx.say("ok!")
           ngx.say("666!")
        }
    }

}
```

/test1　/test2　　ngx.flush(true)

“ test nginx ”　　3s　　“ ok! 666! ”　　ngx.flush(true)

3s

注意：　ngx.flush　　HTTP 1.0

```
# curl -i 'http://testnginx.com/test2'  --http1.0
```

Lua Nginx Ngx-Lua

7.12.1 单一捕获

指令：ngx.re.match

captures, err = ngx.re.match(subject, regex, options?, ctx?, res_table?)

init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

Perl subject

nil nil err

```
location / {
    content_by_lua_block {
        local ngx = require "ngx";
        --           +aaa
    local m, err = ngx.re.match(ngx.var.uri, "([0-9]+)(aaa)");
        if m then
           --
           ngx.say(ngx.var.uri,'---match success---','its type: ',type(m))
           ngx.say(ngx.var.uri,'---m[0]--- ', m[0])
           ngx.say(ngx.var.uri,'---m[1]--- ', m[1])
           ngx.say(ngx.var.uri,'---m[2]--- ', m[2])
        else
           if err then
               ngx.log(ngx.ERR, "error: ", err)
               return
           end
           ngx.say("match not found")
        end

    }
}
```

```
# curl  'http://testnginx.com/test/a123aaa/b456aaa/c'
/test/a123aaa/b456aaa/c---match success---its type: table
/test/a123aaa/b456aaa/c---m[0]---123aaa
/test/a123aaa/b456aaa/c---m[1]---123
/test/a123aaa/b456aaa/c---m[2]---aaa
```

1　ngx.re.match　　456aaa

2　ngx.re.match　　table

3　ngx.re.match　　m[0]　　m[1]　m[2]

7.12.2　全部捕获

ngx.re.match

ngx.re.gmatch

指令：ngx.re.gmatch

iterator, err = ngx.re.gmatch(subject, regex, options?)

init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

ngx.re.match　　Lua

```
location / {
   content_by_lua_block {
      local ngx = require "ngx";
      --     i
      local m_table, err = ngx.re.gmatch(ngx.var.uri, "([0-9]+)(aaa)",
"i");
      if not m_table then
          ngx.log(ngx.ERR,  err)
          return
      end
      while true do
         local m, err = m_table()
         if err then
            ngx.log(ngx.ERR,  err)
```

```
            return
        end
        if not m then
            break
        end
        ngx.say(m[0])
        ngx.say(m[1])
    end

  }
}
```

```
# curl  'http://testnginx.com/test/a123aaa/b456AAA/c'
123aaa
123
456AAA
456
```

ngx.re.match ngx.re.gmatch options options 7-1

表 7-1 options 常用参数说明

参 数	说 明
i	
o	worker
m	Perl /m
j	PCRE JIT PCRE Perl Compatible Regular Expressions C 8.21 Nginx --enable-jit o

7.12.3 更高效的匹配和捕获

ngx.re.match ngx.re.gmatch Lua table

指令：ngx.re.find

from, to, err = ngx.re.find(subject, regex, options?, ctx?, nth?)

init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_

lua*　ssl_certificate_by_lua*　ssl_session_fetch_by _lua*　ssl_session_store_by_lua*

ngx.re.match

ngx.re.find　table　ngx.re.match　ngx.re.gmatch　Lua　string.sub

```
location / {
   content_by_lua_block {
      local ngx = require "ngx";
      local uri = ngx.var.uri
      --     o  j
      local    find_begin,find_end,err    =    ngx.re.find(uri,    "([0-
9]+)(aaa)","oj");
      if find_begin then
          ngx.say('begin: ',find_begin)
          ngx.say('end: ',find_end)
       --     Lua   string.sub
          ngx.say('find it: ' ,string.sub(uri, find_begin,find_end))
          return
      end
     }
}
```

```
# curl  'http://testnginx.com/test/a123aaa/b456AAAa/c'
begin:8
end:13
find it: 123aaa
```

ngx.re.match　ngx.re.gmatch　ngx.re.find　ctx　ctx

- ctx　Lua table　4　5　nth　nil
- ctx　pos　pos=1　ngx.re.find　pos　1
- ctx　ngx.re.find　ctx　ctx

nth　ngx.re.find　5　Lua-Nginx-Module 0.9.3　ngx.re.match　m[1]　m[2]　nth　1　ngx.re.match　m[1]

```
   location / {
      content_by_lua_block {
         local ngx = require "ngx";
         local uri = ngx.var.uri

         --   uri      10                          1
         --       nth   1          ([0-9]+)
         local ctx = { pos = 10 }
         local    find_begin,find_end,err    =    ngx.re.find(uri,    "([0-
9]+)(aaa)","oji",ctx,1);
         if find_begin then
             ngx.say('begin: ',find_begin)
             ngx.say('end: ',find_end)
             ngx.say('find it: ' ,string.sub(uri, find_begin,find_end))
             return
         end
       }
   }
```

```
   # curl  'http://testnginx.com/test/a123aaa/b456AAAa/c'
   begin:10
   end:10
   find it: 3
```

ctx 10 uri “ /test/a12 ” 9
3aaa nth 1 3

7.12.4 替换数据

Lua API

指令：ngx.re.sub

newstr, n, err = ngx.re.sub(subject, regex, replace, options?)

init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

subject regex replace options
newstr n

```
location / {
    content_by_lua_block {
        local ngx = require "ngx";
        local uri = ngx.var.uri
        local n_str, n, err = ngx.re.sub(uri,"([0-9]+)",'zzzz')
        if n_str then
            ngx.say(uri)
            ngx.say(n_str)
            ngx.say(n)
        else
            ngx.log(ngx.ERR, "error: ", err)
            return
        end
    }
}
```

```
# curl   'http://testnginx.com/test188/x2/1231'
/test188/x2/1231
/testzzzz/x2/1231
1
```

1　　n

1　　ngx.re.gsub

```
local n_str, n, err = ngx.re.gsub(uri,"([0-9]+)",'zzzz')
```

3

```
# curl   'http://testnginx.com/test188/x2/1231'
/test188/x2/1231
/testzzzz/xzzzz/zzzz
3
```

7.12.5　转义符号

\d　\s　\w　　Ngx_Lua　　\　　Lua

\

\\\\d　\\\\s　\\\\w　　Nginx　Lua　　\\\\

\\　　\

[[]]

```
local find_regex = [[\d+]]
local m = ngx.re.match("xxx,43", find_regex)
ngx.say(m[0])   --     43
```

[[]]

Nginx subrequest
Nginx Nginx HTTP

location

Nginx GET POST PUT DELETE

7.13.1 请求方法

Lua API Lua API 7-2

表 7-2 Lua API 常见的请求方法说明

Nginx 的请求方法	Lua API 中的请求方法	说 明
GET	ngx.HTTP_GET	GET
HEAD	ngx.HTTP_HEAD	GET
PUT	ngx.HTTP_PUT	
DELETE	ngx.HTTP_DELETE	
POST	ngx.HTTP_POST	
OPTIONS	ngx.HTTP_OPTIONS	

7.13.2 单一子请求

指令：ngx.location.capture

ngx_fastcgi　ngx_memc　ngx_postgres　ngx_drizzle　　Ngx_Lua　Nginx　c

cosockets　　cosockets　ngx.socket.tcp　API

location　internal

res　table　4　res.status　res.header　res.body　res.truncated　res　7-3

表 7-3　res 的元素名及其用途

元 素 名	用　途
res.status	HTTP
res.header	table
res.body	table
res.truncated	

ngx.location.capture　2　options

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';

    location = /main {
        set $m 'hello';
        content_by_lua_block {
            local ngx = require "ngx";
            --              /test          GET         test nginx
            --        URL      a=1&b=2       copy_all_vars
            --          Nginx      $m
              local res = ngx.location.capture(
              '/test',  { method = ngx.HTTP_GET , body = 'test nginx',
               args = { a = 1, b = 2 },copy_all_vars = true }
            )
            ngx.say(res.status)
            ngx.say(res.body)
            ngx.say(type(res.header))
            ngx.say(type(res.truncated))
        }
    }
```

```
        location = /test
    {
        #       Nginx
            internal;
            content_by_lua_block {
                local ngx = require "ngx";
                --
                ngx.req.read_body()
                local body_args = ngx.req.get_body_data()
                --                          m             world
                ngx.print('request_body:  '  ,body_args,  '  capture_args:  ',
ngx.var.args, '---  copy_all_vars : ', ngx.var.m .. 'world!')
            }
        }
    }
```

```
    # curl   'http://testnginx.com/main'
    200
    request_body:test   nginx   capture_args:a=1&b=2---      copy_all_vars   :
helloworld!
    table
    boolean
```

- ngx.location.capture 2 options table
- mthod GET
- body
- args URL args args table
- copy_all_vatrue
-

```
    local res = ngx.location.capture('/test?a=1&b=2')
    local res = ngx.location.capture('/test , args = { a = 1, b = '2' }')
```

ngx.location.capture

- vars table Nginx

copy_all_vars = true　　vars　　vars
vars

- share_all_vars

- always_forward_body　　false　　body
  PUT　POST　　always_forward_body
  true　　body

- ctx　　table　　ngx.ctx

vars

```
location = /main {
    set $m 'hello';
    set $mm '';
    content_by_lua_block {
        local ngx = require "ngx";
        local res = ngx.location.capture(
            '/test',
            { method = ngx.HTTP_POST ,
            vars = {mm = 'MMMMM',m = 'hhhh'}}
        )
        ngx.say(res.body)
    }
}
location = /test {
    content_by_lua_block {
        local ngx = require "ngx";
        ngx.print(ngx.var.m .. ngx.var.mm )
    }
}
```

```
# curl   'http://testnginx.com/main'
hhhhMMMMM
```

/test

注意： ngx.location.capture

Accept-Encoding:gzip

ngx_proxy

Lua proxy_pass_request_headers off

7.13.3 并发子请求

指令：ngx.location.capture_multi

res1, res2, ... = ngx.location.capture_multi({ {uri, options?}, {uri, options?}, ... })

rewrite_by_lua* access_by_lua* content_by_lua*

ngx.location.capture

```
    server {
        listen       80;
        server_name  testnginx.com;
        default_type 'text/plain';
        location = /main {
            set $m 'hello';
            set $mm '';
            content_by_lua_block {
                local ngx = require "ngx";
                --
                local res1, res2 = ngx.location.capture_multi{
                    { "/test1?a=1&b=2" },
                    {  "/test2",{  method  =  ngx.HTTP_POST},body  =  "test
nginx" },
                }
                --      body        ngx.location.capture
                if res1.status == ngx.HTTP_OK then
                    ngx.say(res1.body)
                end

                if res2.status == ngx.HTTP_OK then
                    ngx.say(res2.body)
```

```
            end
        }
    }
    location = /test1 {
        echo 'test1';
    }
    location = /test2 {
        echo 'test2';
    }

}
```

```
# curl   'http://testnginx.com/main'
test1

test2
```

注意：Nginx　　Nginx 1.1
200　　50

Lua API　　Nginx
worker　　worker
ID　　ID　　Nginx　worker　　shutdown

7.14.1　获取环境所在的模块

指令：ngx.config.subsystem

subsystem = ngx.config.subsystem

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　init_by_lua*　init_worker_by_lua*

Nginx　　http　stream　　http

http stream stream

7.14.2 确认调试模式

指令：ngx.config.debug

```
debug = ngx.config.debug
```

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* init_by_lua* init_worker_by_lua*

Debug Debug

7.14.3 获取 prefix 路径

指令：ngx.config.prefix

```
prefix = ngx.config.prefix()
```

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* init_by_lua* init_worker_by_lua*

Nginx --prefix= Nginx -p -p

7.14.4 获取 Nginx 的版本号

指令：ngx.config.nginx_version

```
ver = ngx.config.nginx_version
```

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* init_by_lua* init_worker_by_lua*

Nginx Nginx 1.12.2

7.14.5 获取 configure 信息

指令：ngx.config.nginx_configure

```
str = ngx.config.nginx_configure()
```

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_

by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　init_by_lua*

Nginx　./configure

7.14.6　获取 Ngx_Lua 的版本号

指令：ngx.config.ngx_lua_version

ver = ngx.config.ngx_lua_version

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　init_by_lua*

Ngx_Lua　Ngx_Lua

7.14.7　判断 worker 进程是否退出

指令：ngx.worker.exiting

exiting = ngx.worker.exiting()

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　init_by_lua*　init_worker_by_lua*

Nginx　worker

7.14.8　获取 worker 进程的 ID

指令：ngx.worker.id

count = ngx.worker.id()

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　init_by_lua*

worker　ID　worker　ID　0

worker　1

7.14.9　获取 worker 进程的数量

指令：ngx.worker.count

count = ngx.worker.count()

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* init_by_lua* init_worker_by_lua*

Nginx worker Nginx worker_processes

Nginx MySQL

Nginx

Lua-Nginx_Module 0.10.9 ngx.timer.at Lua-Nginx_Module 0.10.9 ngx.timer.every

ngx.timer.every

7.15.1 创建定时任务

指令：ngx.timer.every

hdl, err = ngx.timer.every(delay, callback, user_arg1, user_arg2, ...)

init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

delay 0.001s

0s callback Lua Nginx

```
   init_worker_by_lua_block {
       local delay = 3;
       local ngx = require "ngx";
       local check
       check = function(premature)
           if not premature then
            --        worker      PID   ID
               ngx.log(ngx.ERR,   '   ngx.worker.pid:   ',ngx.worker.pid(),'
ngx.worker.id: ',ngx.worker.id(),"------test nginx !!!")
           end
       end
       --     3s        check
```

```
        local ok, err = ngx.timer.every(delay, check)
        if not ok then
             ngx.log(ngx.ERR, "failed to create timer: ", err)
             return
        end
    }
```

Nginx　　worker　　7-1

7-1

7-1

- worker
- 3s
- Nginx

user_arg1　user_arg2

```
    init_worker_by_lua_block {
        local delay = 3;
        local ngx = require "ngx";
        local check
        --        u_arg1                    'test nginx'
        check = function(premature,u_arg1)
            if not premature then
                ngx.log(ngx.ERR,    '    ngx.worker.pid:    ',ngx.worker.pid(),'
ngx.worker.id: ',ngx.worker.id(),'------', u_arg1)
            end
        end
```

```
        --        'test nginx'
        local ok, err = ngx.timer.every(delay, check,'test nginx')
        if not ok then
            ngx.log(ngx.ERR, "failed to create timer: ", err)
            return
        end
    }
```

7.15.2 性能优化

Nginx 3 worker 3s 3 worker
worker
worker
worker
worker ID ID
worker

```
    init_worker_by_lua_block {
         local delay = 3;
         local ngx = require "ngx";
         local check
         check = function(premature)
             if not premature then
                 ngx.log(ngx.ERR,   '   ngx.worker.pid:   ',ngx.worker.pid(),'
ngx.worker.id: ',ngx.worker.id(),"------test nginx !!!")
             end
         end

       --     worker      ID   0
       if 0 == ngx.worker.id() then
             local ok, err = ngx.timer.every(delay, check)
             if not ok then
                 ngx.log(ngx.ERR, "failed to create timer: ", err)
                 return
             end
        end

    }
```

3s worker

worker Nginx master worker

worker　　ID

Nginx

```
local ok, err = ngx.timer.at(0,func)
```

func　　ngx.timer.at

注意：　　init_worker_by_lua*　　8.2

Nginx

Nginx

指令：lua_max_running_timers

lua_max_running_timers <count>

lua_max_running_timers 256

http

running timers

“ N lua_max_running_timers are not enough ”　　N

running timers

指令：lua_max_pending_timers

lua_max_pending_timers <count>

lua_max_pending_timers 1024

http

pending timers

“ too many pending timers ”

7.15.3　禁用的 Lua API

ngx.timer.every

API　　API

- ngx.location.capture
- Lua API　　ngx.say　ngx.print　　ngx.flush
- ngx.req.　　Lua API

Ngx_Lua Lua API

7.16.1 请求重定向

Nginx rewrite Ngx_Lua ngx.redirect ngx.req.set_uri Ngx_Lua ngx.exec

指令：ngx.redirect

ngx.redirect(uri, status?)

rewrite_by_lua* access_by_lua* content_by_lua*

HTTP 301 302 URI

status 301 302 303 307 308 302 ngx.redirect rewrite

```
location / {
    #       rewrite ^/ http://testnginx.com/test? redirect;
    rewrite_by_lua_block {
        return ngx.redirect("/test")
    }
}
```

302

```
location / {
    #       rewrite ^/ http://testnginx.com/test permanent;
    rewrite_by_lua_block {
        local ngx = require "ngx";
        return ngx.redirect("/test?" ..  ngx.var.args  ,301)
    }
}
```

```
return ngx.redirect("/test?test=1&a=2" ,301)
```

http://abc.testnginx.com

```
return ngx.redirect("http://abc.testnginx.com",301)
```

注意： return

指令：ngx.req.set_uri

ngx.req.set_uri (uri, jump?)

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua*

uri URL Nginx rewrite rewrite rewrite ^ /test last; ngx.req.set_uri("/test", true) rewrite ^ /test break; ngx.req.set_uri("/foo", false)

ngx.req.set_uri_args

```
ngx.req.set_uri_args("a=1&b=2&c=3")
ngx.req.set_uri("/test", true)
```

指令：ngx.exec

ngx.exec(uri, args?)

rewrite_by_lua access_by_lua* content_by_lua*

uri args echo-nginx-module echo_exec

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';
    location / {
        content_by_lua_block {
            return ngx.exec('/test');
        }
    }
    location /test {
        content_by_lua_block {
            ngx.say(ngx.var.args);
        }
    }
}
```

ngx_exec

- “ ngx.exec('/test',ngx.var.args); ”

- “ ngx.exec('/test' ,ngx.var.args .. 'd=4'); ”
- “ ngx.exec('/test' , 'd=4'); ”

注意：ngx_exec HTTP
return ngx.exec(…)

7.16.2 日志记录

Lua Lua API
ngx.log

指令：ngx.log

ngx.log(log_level, …)

init_by_lua* init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

log_level error.log

log_level 7-4 Nginx error.log

表 7-4 log_level 的级别及其说明

级 别	说 明
ngx.STDERR	
ngx.EMERG	
ngx.ALERT	
ngx.CRIT	
ngx.ERR	
ngx.WARN	
ngx.NOTICE	
ngx.INFO	
ngx.DEBUG	

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';
```

```
        location / {
            content_by_lua_block {
                ngx.say("test ")
                ngx.say("nginx ")
                ngx.log(ngx.ALERT,'Log Test Nginx')
                ngx.log(ngx.STDERR,'Log Test Nginx')
                ngx.log(ngx.EMERG,'Log Test Nginx')
                ngx.log(ngx.ALERT,'Log Test Nginx')
                ngx.log(ngx.CRIT,'Log Test Nginx')
                ngx.log(ngx.ERR,'Log Test Nginx')
                ngx.log(ngx.WARN,'Log Test Nginx')
                ngx.log(ngx.NOTICE,'Log Test Nginx')
                ngx.log(ngx.INFO,'Log Test Nginx')
                ngx.log(ngx.DEBUG,'Log Test Nginx')
            }
        }

    }
```

```
    curl -i 'http://testnginx.com/'
```

error.log　　logs/error.log

```
    2018/06/11  11:18:26  [alert]  1180#1180:  *34  [lua]  content_by_lua
(nginx.conf:66):4:  Log  Test  Nginx,  client:  10.19.48.161,  server:
testnginx.com, request: "GET / HTTP/1.1", host: "testnginx.com"
    2018/06/11  11:18:26  []  1180#1180:  *34  [lua]  content_by_lua
(nginx.conf:66):5:  Log  Test  Nginx,  client:  10.19.48.161,  server:
testnginx.com, request: "GET / HTTP/1.1", host: "testnginx.com"
    2018/06/11  11:18:26  [emerg]  1180#1180:  *34  [lua]  content_by_lua
(nginx.conf:66):6:  Log  Test  Nginx,  client:  10.19.48.161,  server:
testnginx.com, request: "GET / HTTP/1.1", host: "testnginx.com"
    2018/06/11  11:18:26  [alert]  1180#1180:  *34  [lua]  content_by_lua
(nginx.conf:66):7:  Log  Test  Nginx,  client:  10.19.48.161,  server:
testnginx.com, request: "GET / HTTP/1.1", host: "testnginx.com"
    2018/06/11  11:18:26  [crit]  1180#1180:  *34  [lua]  content_by_lua
(nginx.conf:66):8:  Log  Test  Nginx,  client:  10.19.48.161,  server:
testnginx.com, request: "GET / HTTP/1.1", host: "testnginx.com"
    2018/06/11  11:18:26  [error]  1180#1180:  *34  [lua]  content_by_lua
(nginx.conf:66):9:  Log  Test  Nginx,  client:  10.19.48.161,  server:
testnginx.com, request: "GET / HTTP/1.1", host: "testnginx.com"
```

error.log Nginx error.log
Lua error.log

```
error_log /usr/local/nginx_1.12.2/logs/error.log info;
```

Lua info debug Nginx
Debug

ngx.log

```
ngx.log(ngx.ERR, 'Log Test Nginx', 'a', 'b', 'c')
```

ngx.log Nginx 2048 2KB

Lua print info

```
print("Log Test Nginx ")
ngx.log(ngx.INFO, 'Log Test Nginx')
```

注意：ngx.print print

7.16.3 请求中断处理

Lua

-
-

ngx.exit

指令：ngx.exit

ngx.exit(status)

rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

status HTTP status ≥ 200
status Nginx status=0

init_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_ lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_ lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

Ngx_Lua HTTP　　7-5

表 7-5　Ngx_Lua HTTP 状态码清单

赋值方式	HTTP 状态码
value = ngx.HTTP_CONTINUE	100
value = ngx.HTTP_SWITCHING_PROTOCOLS	101
value = ngx.HTTP_OK	200
value = ngx.HTTP_CREATED	201
value = ngx.HTTP_ACCEPTED	202
value = ngx.HTTP_NO_CONTENT	204
value = ngx.HTTP_PARTIAL_CONTENT	206
value = ngx.HTTP_SPECIAL_RESPONSE	300
value = ngx.HTTP_MOVED_PERMANENTLY	301
value = ngx.HTTP_MOVED_TEMPORARILY	302
value = ngx.HTTP_SEE_OTHER	303
value = ngx.HTTP_NOT_MODIFIED	304
value = ngx.HTTP_TEMPORARY_REDIRECT	307
value = ngx.HTTP_PERMANENT_REDIRECT	308
value = ngx.HTTP_BAD_REQUEST	400
value = ngx.HTTP_UNAUTHORIZED	401
value = ngx.HTTP_PAYMENT_REQUIRED	402
value = ngx.HTTP_FORBIDDEN	403
value = ngx.HTTP_NOT_FOUND	404
value = ngx.HTTP_NOT_ALLOWED	405
value = ngx.HTTP_NOT_ACCEPTABLE	406
value = ngx.HTTP_REQUEST_TIMEOUT	408
value = ngx.HTTP_CONFLICT	409
value = ngx.HTTP_GONE	410
value = ngx.HTTP_UPGRADE_REQUIRED	426
value = ngx.HTTP_TOO_MANY_REQUESTS	429
value = ngx.HTTP_CLOSE	444
value = ngx.HTTP_ILLEGAL	451
value = ngx.HTTP_INTERNAL_SERVER_ERROR	500
value = ngx.HTTP_METHOD_NOT_IMPLEMENTED	501
value = ngx.HTTP_BAD_GATEWAY	502
value = ngx.HTTP_SERVICE_UNAVAILABLE	503

续表

赋值方式	HTTP 状态码
value = ngx.HTTP_GATEWAY_TIMEOUT	504
value = ngx.HTTP_VERSION_NOT_SUPPORTED	505
value = ngx.HTTP_INSUFFICIENT_STORAGE	507

HTTP 0

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';
    location / {
        set $a  '0';
        rewrite_by_lua_block {
             ngx.var.a = '1';
             --       ngx.exit(0), 0   HTTP
             ngx.exit(ngx.OK)
        }
        echo $a;  #          1
    }
}
```

ngx.exit(ngx.OK) Nginx rewrite_by_lua_block
echo

- 200 300
- 500

0

```
location / {
    set $a  '0';
    rewrite_by_lua_block {
         ngx.var.a = '1';
         ngx.exit(200)  --          500
    }
         echo $a;  #
}
```

200 ngx.exit echo

return

```
return ngx.exit(ngx.OK)
```

Lua　　Lua　　Nginx

lua_code_cache

lua_code_cache on | off

lua_code_cache on

http　server　location　location if

*_by_lua_file　　Lua　　Lua　　off

*_by_lua_file　　Nginx

注意：　　*_by_lua_file　　*_by_lua_block　　*_by_lua

Nginx　　reload

lua_code_cache　　on

7.17.1　断开客户端连接

API

指令：ngx.eof

ok, err = ngx.eof()

rewrite_by_lua*　access_by_lua*　content_by_lua*

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';
    location / {
```

```
            set $a '0';
            content_by_lua_block {
                ngx.var.a = '1';
                --
                ngx.eof()
                ngx.sleep(3);  --          3s
                ngx.log(ngx.ERR,'Test Nginx---',ngx.var.a)
            }
        }
    }
```

curl -i 'http://testnginx.com/ 200

3s error.log

注意： ngx.eof

Nginx proxy_ignore_client_abort proxy_ignore_client_abort on ngx.eof

7.17.2 请求休眠

指令：ngx.sleep

ngx.sleep(seconds)

rewrite_by_lua* access_by_lua* content_by_lua* ngx.timer.* ssl_certificate_by_lua* ssl_session_fetch_by_lua*

ngx.sleep Nginx worker

seconds seconds 0.001s

```
    location / {
        content_by_lua_block {
            -- 5s      ok
            ngx.sleep(5);
            ngx.say('ok')
        }
    }
```

7.17.3 获取系统时间

Ngx_Lua Nginx

Lua　　os.time

init_worker_by_lua*　set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　balancer_by_lua*　ssl_certificate_by_lua*　ssl_session_fetch_ by_lua*　ssl_session_store_by_lua*

Ngx_Lua

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';
    location / {
       content_by_lua_block {
            ngx.say('ngx.today: ',ngx.today())
            ngx.say('ngx.time: ',ngx.time())
            ngx.say('ngx.now: ',ngx.now())
            ngx.say('ngx.localtime: ',ngx.localtime())
            ngx.say('ngx.utctime: ',ngx.utctime())
            ngx.say('ngx.cookie_time: ',ngx.cookie_time(1528721405))
            ngx.say('ngx.parse_http_time:  ',ngx.parse_http_time('Mon,  11-
Jun-18 12:50:05 GMT'))
            ngx.say('ngx.update_time: ',ngx.update_time())
       }
    }
}
```

```
ngx.today: 2018-06-11    #
ngx.time: 1528721734     #             UNIX
ngx.now: 1528721734.775 #            UNIX
ngx.localtime: 2018-06-11 20:55:34    #
ngx.utctime: 2018-06-11 12:55:34      #     UTC  Coordinated Universal
                                      # Time
ngx.cookie_time: Mon, 11-Jun-18 12:50:05 GMT  #             Cookie
                                              #             UNIX
ngx.http_time: Mon, 11 Jun 2018 12:50:05 GMT  #             HTTP
                                    #        Expires   Last-Modified
ngx.parse_http_time: 1528721405   #     UNIX          ngx.http_time
                                    #
```

```
    ngx.update_time:      #                    Nginx
                          #
```

7.17.4 编码与解码

Ngx_Lua API

指令：ngx.escape_uri

newstr = ngx.escape_uri(str)

init_by_lua* init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua* ngx.quote_sql_str

str URI

指令：ngx.unescape_uri

newstr = ngx.unescape_uri(str)

init_by_lua* init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua*

str URI

指令：ngx.encode_args

str = ngx.encode_args(table)

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua*

URI Lua table

指令：ngx.decode_args

table, err = ngx.decode_args(str, max_args?)

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

URI　　Lua　table

指令：ngx.md5

digest = ngx.md5(str)

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　balancer_by_lua*　ssl_certificate_by_lua*　ssl_session_fetch_by_lua*　ssl_session_store_by_lua*

str　　md5

指令：ngx.md5_bin

digest = ngx.md5_bin(str)

set_by_lua*　rewrite_by_lua*　access_by_lua*　content_by_lua*　header_filter_by_lua*　body_filter_by_lua*　log_by_lua*　ngx.timer.*　balancer_by_lua*　ssl_certificate_by_lua*　ssl_session_fetch_by_lua*　ssl_session_store_by_lua*

str　　md5

注意：ngx.escape_uri　ngx.unescape_uri　　ngx.encode_args　ngx.decode_args

```
   server {
       listen       80;
       server_name  testnginx.com;
       default_type 'text/plain';
       location / {
          content_by_lua_block {
               local ngx = require "ngx";
               --   URI
               ngx.say(ngx.var.uri,'---ngx.escape_uri---', ngx.escape_uri
(ngx.var.uri))

               --              URI
               ngx.say('%2Ftest%2Fa%2Fb%2Fc','---ngx.unescape_uri---',
ngx.unescape_uri('%2Ftest%2Fa%2Fb%2Fc'))

               --   Lua   table
               local  args_table_new =  ngx.encode_args({a = 1, b = 2, c =
```

```
3 })
                ngx.say('{a = 1, b = 2, c = 3 }', '---ngx.encode_args---' ,
args_table_new)

                --   URI                          table
                local args = ngx.var.args
                local args_table = ngx.decode_args(args)
                ngx.say(args, '---ngx.decode_args---', 'a=',args_table["a"])
                --     table    a
                --   URI    md5
                ngx.say(ngx.var.uri, '---ngx.md5---',ngx.md5(ngx.var.uri))
                --   URI    md5
                ngx.say(ngx.var.uri, '---ngx.md5_bin---',
ngx.md5_bin(ngx.var.uri))
            }
        }
    }
```

```
    # curl  'http://testnginx.com/test/a/b/c?a=1&b=2&c=3'
    /test/a/b/c---ngx.escape_uri---%2Ftest%2Fa%2Fb%2Fc
    %2Ftest%2Fa%2Fb%2Fc---ngx.unescape_uri---/test/a/b/c
    {a = 1, b = 2, c = 3 }---ngx.encode_args---b=2&a=1&c=3
    a=1&b=2&c=3---ngx.decode_args---a=1
    /test/a/b/c---ngx.md5---dfa371a9a8f52c9aadd016bda535fa43
    /test/a/b/c---ngx.md5_bin---▪q©¨□Z½¥5
```

7.17.5 防止 SQL 注入

Lua API ngx.quote_sql_str MySQL SQL Nginx MySQL SQL Nginx 9

指令：ngx.quote_sql_str

quoted_value = ngx.quote_sql_str(raw_value)

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

7.17.6　判断是否为子请求

指令：ngx.is_subrequest

value = ngx.is_subrequest

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua*

Nginx true false

```
server {
    listen       80;
    server_name  testnginx.com;
    default_type 'text/plain';
    location /main1 {
         echo_location /test;
    }

    location /main2 {
       content_by_lua_block {
         return ngx.exec('/test');
       }
    }

    location /main3 {
       content_by_lua_block {
          local res = ngx.location.capture("/test")
          ngx.say(res.body)
       }
    }

    location /test {
      content_by_lua_block {
          ngx.say('test');
          ngx.say('is_subrequest: ',ngx.is_subrequest);
      }
    }
}
```

```
[root@testnginx ~]# curl  'http://testnginx.com/main1'
```

```
test
is_subrequest: true
[root@testnginx ~]# curl  'http://testnginx.com/main2'
test
is_subrequest: false
[root@testnginx ~]# curl  'http://testnginx.com/main3'
test
is_subrequest: true
```

ngx.exec echo_location ngx.location.capture

7.17.7 设置 MIME 类型

Ngx_Lua Nginx MIME JSON HTML

Default_Type Lua JSON

```
default_type application/json;
```

default_type mime-type;

default_type text/plain;

http server location

建议： Lua Ngx_Lua Content-Type

Ngx_Lua Nginx Ngx_Lua Ngx_Lua Wiki openresty.org Nginx “ ”

8

7 Ngx_Lua Ngx_Lua

- Lua Nginx
- Lua Nginx

Ngx_Lua

3.9 Nginx 11

Ngx_Lua Nginx 11 Lua

Ngx_Lua

init_by_lua_block init_by_lua OpenResty 1.9.3.1 Lua-Nginx-Modulev 0.9.17 init_by_lua init_by_lua_block init_by_lua init_by_lua_block

*_block

8.1.1 阶段说明

init_by_lua_block {lua-script-str}

http

loading-config

Nginx master Nginx Nginx Lua VM Virtual Machine <lua-script-str> Nginx HUP Hangup

8.1.2 初始化配置

loading-config

1 Lua master CPU

2

3 lua_shared_dict 10

```
    user  webuser webuser;
    worker_processes  1;
    worker_rlimit_nofile 10240;

    events {
        use epoll;
        worker_connections  10240;
    }

    http {
        include       mime.types;
        default_type  application/octet-stream;
        log_format main '$remote_addr-$remote_user[$time_local] "$request" '
                          '$status $body_bytes_sent "$http_referer" '
                          '"$http_user_agent"          "$http_x_forwarded_for"
"$request_time"  "$upstream_addr  $upstream_status  $upstream_response_time"
"upstream_time_sum:$upstream_time_sum"  "jk_uri:$jk_uri"';
        access_log  logs/access.log  main;
        sendfile        on;
        keepalive_timeout  65;


        lua_package_path "/usr/local/nginx_1.12.2/conf/lua_modules/?.lua;;";
```

```
        lua_package_cpath
"/usr/local/nginx_1.12.2/conf/lua_modules/c_package/?.so;;";

        lua_shared_dict dict_a 100k;  --        Lua         dict_a   100KB
    init_by_lua_block {
    -- cjson.so                lua_package_cpath
    --       OpenResty
              cjson = require "cjson";
              local dict_a = ngx.shared.dict_a;
              dict_a:set("abc", 9)
        }

        server {
            listen       80;
            server_name  testnginx.com;
            location / {
               content_by_lua_block {
                   ngx.say(cjson.encode({a = 1, b = 2}))
                   local dict_a = ngx.shared.dict_a;
                   ngx.say("abc=",dict_a:get("abc"))
               }
            }
        }
```

```
    # curl -I http://testnginx.com/
    {"a":1,"b":2}
    abc=9
```

8.1.3　控制初始值

init_by_lua_block　　Nginx　　Nginx

```
    init_by_lua_block {
          local cjson = require "cjson";
          local dict_a = ngx.shared.dict_a;
          local v = dict_a:get("abc");  --                set
          if not v then                 --
```

8.1.4 init_by_lua_file

init_by_lua_file init_by_lua_block init_by_lua_block
Lua Nginx

init_by_lua_file Nginx -p PATH
Nginx -p PATH --prefix
Nginx $prefix ${prefix} init_by_lua_file Nginx include

```
init_by_lua_file conf/lua/init.lua;                    --
init_by_lua_file /usr/local/nginx/conf/lua/init.lua; --
```

init.lua

```
cjson = require "cjson"
local dict_a = ngx.shared.dict_a
local v = dict_a:get("abc")
if not v then
   dict_a:set("abc", 9)
end
```

8.1.5 可使用的 Lua API 指令

init_by_lua* Nginx Lua API
Lua API ngx.log ngx.shared.DICT print

注意：init_by_lua* * init_by_lua* init_by_lua API

8.2.1 阶段说明

init_worker_by_lua_block {lua-script-str}

http

starting-worker

master worker Lua Nginx

master　　init_by_lua*

8.2.2　启动 Nginx 的定时任务

init_worker_by_lua_block

```
user  webuser webuser;
worker_processes  3;
worker_rlimit_nofile 10240;

events {
    use epoll;
    worker_connections  10240;
}

http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;

    upstream  test_12 {
        server 127.0.0.1:81 weight=20 max_fails=300000 fail_timeout=5s;
        server 127.0.0.1:82 weight=20 max_fails=300000 fail_timeout=5s;
    }

    lua_package_path "${prefix}conf/lua_modules/?.lua;;";
    lua_package_cpath "${prefix}conf/lua_modules/c_package/?.so;;";

    init_worker_by_lua_block  {
        local delay = 3  --3s
        local cron_a
    --
        cron_a = function (premature)
    --
            if not  premature then
                ngx.log(ngx.ERR, "Just do it !")
            end

        end
    --     delay                     cron_a
        local ok, err = ngx.timer.every(delay, cron_a)
        if not ok then
            ngx.log(ngx.ERR, "failed to create the timer: ", err)
```

```
            return
        end
    }
```

8.2.3 动态进行后端健康检查

init_worker_by_lua_block HTTP

Nginx health_check

OpenResty 1.9.3.2 lua-upstream-nginx-module

Nginx

```
# git clone https://github.com/openresty/lua-upstream-nginx-module.git
# cd nginx-1.12.2
# ./configure --prefix=/opt/nginx \
    --with-ld-opt="-Wl,-rpath,$LUAJIT_LIB" \
    --add-module=/path/to/lua-nginx-module \
    --add-module=/path/to/lua-upstream-nginx-module
# make && make install
```

注意： Nginx

lua-resty-upstream-healthcheck lib lua_package_path

```
# git clone https://github.com/openresty/lua-resty-upstream-
healthcheck.git
# cp lua-resty-upstream-healthcheck/lib/resty/upstream/healthcheck.lua
/usr/local/nginx_1.12.2/conf/lua_modules/resty/
```

```
user  webuser webuser;
worker_processes  3;
worker_rlimit_nofile 10240;

events {
    use epoll;
    worker_connections  10240;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
```

```
    upstream  test_12 {
        server 127.0.0.1:81  weight=20  max_fails=10 fail_timeout=5s;
        server 127.0.0.1:82  weight=20  max_fails=10 fail_timeout=5s;
        server 127.0.0.1:8231  weight=20  max_fails=10 fail_timeout=5s;
    }
    lua_shared_dict healthcheck 1m;  #     upstream servers
                                     #
    lua_socket_log_errors off;   #   TCP                error
                                 # error.log
                                 #
                                 #                  ngx.log
    lua_package_path "${prefix}conf/lua_modules/?.lua;;";
    lua_package_cpath "${prefix}conf/lua_modules/c_package/?.so;;";

    init_worker_by_lua_block  {
        local hc = require "resty.upstream.healthcheck"
        local ok, err = hc.spawn_checker{
            shm = "healthcheck",  --
            upstream = "test_12", --             upstream
            type = "http",        --          http
            http_req      =      "GET      /status      HTTP/1.0\r\nHost:
testnginx.com\r\n\r\n",
            --        HTTP
            interval = 3000,  --              3s/
            timeout = 1000,   --                 1s
            fall = 3,  --           3                down
            rise = 2,  --           2                up
        valid_statuses = {200, 302},  --                       200   302
            concurrency = 10,  --
        }
        if not ok then
            ngx.log(ngx.ERR, "failed to spawn health checker: ", err)
            return
        end
    }
    server {
        listen       80;
        server_name  testnginx.com;
        location / {
           proxy_pass http://test_12;
        }
        # /status
        location = /status {
```

```
            default_type text/plain;
            content_by_lua_block {
                local hc = require "resty.upstream.healthcheck"
                --                    worker
                ngx.say("Nginx Worker PID: ", ngx.worker.pid())
                -- status_page()
                ngx.print(hc.status_page())
            }
        }
    }
```

http://testnginx.com/status 8-1

8-1

upstream

```
local ok, err = hc.spawn_checker{
    shm = "healthcheck",
    upstream = "test_12",
    type = "http",

    http_req = "GET /status HTTP/1.0\r\nHost: testnginx.com\r\n\r\n",

    interval = 3000,
    timeout = 1000,
    fall = 3,
    rise = 2,
    valid_statuses = {200, 302},
    concurrency = 10,
}
local ok, err = hc.spawn_checker{
    shm = "healthcheck",
    upstream = "test_34",
    type = "http",
```

```
        http_req = "GET /test HTTP/1.0\r\nHost: testnginx.com\r\n\r\n",
        interval = 3000,
        timeout = 1000,
        fall = 3,
        rise = 2,
        valid_statuses = {200, 302},
        concurrency = 10,
```

lua_socket_log_errors　　on

error.log　　8-2

```
2018/05/28 16:08:36 [error] 32374#32374: *64 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:38 [error] 32375#32375: *84 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:39 [error] 32375#32375: *84 lua tcp socket read timed out, context: ngx.timer
2018/05/28 16:08:39 [error] 32375#32375: *84 lua tcp socket read timed out, context: ngx.timer
2018/05/28 16:08:39 [error] 32375#32375: *84 lua tcp socket read timed out, context: ngx.timer
2018/05/28 16:08:40 [error] 32376#32376: *104 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:43 [error] 32374#32374: *124 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:45 [error] 32376#32376: *144 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:47 [error] 32375#32375: *164 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:49 [error] 32374#32374: *184 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:50 [error] 32374#32374: *184 lua tcp socket read timed out, context: ngx.timer
2018/05/28 16:08:52 [error] 32375#32375: *204 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:54 [error] 32376#32376: *224 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:55 [error] 32376#32376: *224 lua tcp socket read timed out, context: ngx.timer
2018/05/28 16:08:56 [error] 32374#32374: *244 connect() failed (111: Connection refused), context: ngx.timer
2018/05/28 16:08:57 [error] 32374#32374: *244 lua tcp socket read timed out, context: ngx.timer
2018/05/28 16:08:59 [error] 32375#32375: *264 connect() failed (111: Connection refused), context: ngx.timer
```

8-2

lua-resty-upstream-healthcheck　　Nginx

upstream

8.3.1　阶段说明

set_by_lua_block $res {lua-script-str}

server　server if　location　location if

rewrite

<lua-script-str>　　$res

8.3.2　变量赋值

$res　　$res

```
server {
    listen       80;
    server_name  testnginx.com;
    location / {
    set $a '';
    set_by_lua_block $a {
         local t = 'tes'
         return t
    }
    return 200 $a;
}
```

```
#  curl http://testnginx.com/
test
```

ngx.var.VARIABLE

```
server {
    listen       80;
    server_name  testnginx.com;
    location / {
    #     ngx.var.VARIABLE
    set $b '';
    set_by_lua_block $a {
        local t = 'test'
        ngx.var.b = 'test_b'
        return t
    }
    return 200 $a,$b;
    }
}
```

```
#  curl http://testnginx.com/test
test,test_b
```

8.3.3 rewrite 阶段的混用模式

set_by_lua_block rewrite ngx_http_rewrite_module set-misc-nginx-module array-var-nginx-module

```
server {
    listen       80;
    server_name  testnginx.com;
    location / {

    set $a '123';
    set_by_lua_block $b {
        local t = 'bbb'
        return t
    }
    set_by_lua_block $c {
        local t = 'ccc'  .. ngx.var.b
        return t
    }
    set $d "456$c";
    return 200 $a,$b,$c,$d;
    }

}
```

```
# curl  http://testnginx.com/test
123,bbb,cccbbb,456cccbbb
```

8.3.4　阻塞事件

set_by_lua_block　　Nginx

set_by_lua_block　　I/O

yield　　Lua

8.3.5　被禁用的 Lua API 指令

set_by_lua_block　　Lua API

- 　　API　　ngx.say　ngx.send_headers
- 　　API　　ngx.exit
- 　　API　　ngx.location.capture　ngx.location.capture_multi
- Cosocket API　　ngx.socket.tcp　ngx.req.socket
- 　　API　　ngx.sleep

8.4.1 阶段说明

rewrite_by_lua_block {lua-script-str}

http server location location if

rewrite tail

<lua-script-str> Lua

Lua Lua API

Lua API

URL MySQL Redis

8.4.2 利用 rewrite_by_lua_no_postpone 改变执行顺序

rewrite_by_lua_block ngx_http_rewrite_module

```
server {
    listen       80;
    server_name  testnginx.com;
    location / {
     set $b  '1';
     rewrite_by_lua_block { ngx.var.b = '2'}
     set $b  '3';
     echo $b;
    }
}
```

3 2

```
#  curl  http://testnginx.com/test
2
```

rewrite_by_lua_block rewrite

rewrite_by_lua_no_postpone

rewrite_by_lua_no_postpone on|off

rewrite_by_lua_no_postpone off

http

rewrite rewrite_by_lua*

off　　on

```
rewrite_by_lua_no_postpone on;   #       http
server {
    listen       80;
    server_name  testnginx.com;
    location / {
        set $b  '1';
        rewrite_by_lua_block { ngx.var.b = '2'}
        set $b  '3';
        echo $b;
    }
}
```

```
#  curl -i http://testnginx.com/test
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Mon, 28 May 2018 12:47:37 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

3
```

注意：if　　rewrite_by_lua_block　　if

8.4.3　阶段控制

rewrite_by_lua_block　　ngx.exit(ngx.OK)

ngx.exit　　7.16.3

8.5.1　阶段说明

access_by_lua_block {lua-script-str}

http　server　location　location if

access tail

Nginx access <lua-script-str> rewrite_by_lua_block Lua Lua API

8.5.2 利用 access_by_lua_no_postpone 改变执行顺序

access_by_lua_block ngx_http_access_module access_by_lua_no_postpone on access_by_lua_no_ postpone rewrite_by_lua_no_postpone

8.5.3 阶段控制

access_by_lua_block rewrite_by_lua_block ngx.exit

8.5.4 动态配置黑白名单

Nginx allow deny IP IP Nginx access_by_lua_block

- Redis Nginx+Lua Redis Redis I/O
- Ngx_Lua

Nginx

8.6.1 阶段说明

content_by_lua_block {lua-script-str}

location　location if

content

content_by_lua_block　　<lua-script-str>
rewrite_by_lua_block　　Lua　　Lua API

content_by_lua_block　　echo　return　proxy_pass

```
server {
    listen       80;
    server_name  testnginx.com;

    location / {
        content_by_lua_block {
             ngx.say("content_by_lua_block")
        }
        echo 'ok';
    }
}
```

ok　　content_by_lua_block

8.6.2　动态调整执行文件的路径

content_by_lua_file　　Lua
URL

```
location / {
    # content_by_lua_file           URL      file_name
    content_by_lua_file conf/lua/$arg_file_name;
}
```

8.7.1　阶段说明

balancer_by_lua_block { lua-script }

upstream

content

upstream upstream

```
upstream foo {
    server 127.0.0.1;  #
    balancer_by_lua_block {
        #       IP
    }
}
```

注意： 11.4

8.7.2 被禁用的 Lua API 指令

balancer_by_lua_block Lua yield
yield Lua API cosockets light threads ngx.ctx
rewrite_by_lua* upstream

8.8.1 阶段说明

header_filter_by_lua_block { lua-script }

http server location location if

output-header-filter

header_filter_by_lua_block <lua-script-str>

test nginx-lua

```
location / {
    header_filter_by_lua_block {
       ngx.header.test = "nginx-lua";
    }
    echo 'ok';
}
```

8.8.2　被禁用的 Lua API 指令

header_filter_by_lua_block　　Lua API

- 　　API　　ngx.say　ngx.send_headers
- 　　API　　ngx.redirect　ngx.exec
- 　　API　　ngx.location.capture　ngx.location.capture_multi
- cosocket API　　ngx.socket.tcp　ngx.req.socket

8.9.1　阶段说明

body_filter_by_lua_block { lua-script-str }

http　server　location　location if

output-body-filter

body_filter_by_lua_block　　<lua-script-str>

8.9.2　控制响应体数据

ngx.arg[1]　Lua　　eof

ngx.arg[2]　Lua

Nginx chain

eof　　Nginx chain　　last_buf　　last_in_chain

Nginx　　Lua

```
    server {
        listen       80;
        server_name  testnginx.com;

        location / {
            #
            body_filter_by_lua_block   {   ngx.arg[1]   =   string.upper(ngx.
arg[1]) }
            echo 'oooKkk';
```

```
        echo 'oooKkk';
        echo 'oooKkk';
    }
}
```

```
#  curl -i http://testnginx.com/?file_name=1.lua
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Tue, 29 May 2018 11:36:54 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

OOOKKK
OOOKKK
OOOKKK
```

return ngx.ERROR

```
location / {
    body_filter_by_lua_block {
        ngx.arg[1] = string.upper(ngx.arg[1]);
        return ngx.ERROR
    }
    echo 'oooKkk';
    echo 'oooKkk';
    echo 'oooKkk';
}
```

```
#  curl -i http://testnginx.com/
curl: (52) Empty reply from server
```

ngx.arg[2] true return ngx.ERROR

```
server {
    listen       80;
    server_name  testnginx.com;
    location / {
        body_filter_by_lua_block {
           local body_chunk = ngx.arg[1]
           --                2ooo      ngx.arg[2]=true
```

```
            if string.match(body_chunk, "2ooo") then
                ngx.arg[2] = true
                return
            end
            --     ngx.arg[1] = nil
            ngx.arg[1] = nil

        }
        echo '1oooKkk';
        echo '2oooKkk';
        echo '3oooKkk';
    }
}
```

```
#  curl -i http://testnginx.com/?file_name=1.lua
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Tue, 29 May 2018 11:48:52 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

2oooKkk
```

- 1oooKkk　　if　　ngx.arg[1] = nil
- 3oooKkk　　2oooKkk　　ngx.arg[2] = true

8.9.3　被禁用的 Lua API 指令

body_filter_by_lua_block　　Lua API

- API　　ngx.say　　ngx.send_headers
- API　　ngx.redirect　　ngx.exec
- API　　ngx.location.capture　　ngx.location.capture_multi
- cosocket API　　ngx.socket.tcp　　ngx.req.socket

8.10.1 阶段说明

log_by_lua_block { lua-script }

http server location location if

log

{ lua-script } access

access

log_by_lua_block

12 log Ngx_Lua

```
server {
    listen       80;
    server_name  testnginx.com;

    location / {
        log_by_lua_block {
            local ngx = require "ngx";
            xxxsada  --
            ngx.log(ngx.ERR, 'x');
        }
        echo 'ok';
    }
}
```

error_log “ ok ”

8.10.2 被禁用的 Lua API 指令

log_by_lua_block Lua API

- API ngx.say ngx.send_headers
- API ngx.redirect ngx.exec
- API ngx.location.capture ngx.location.capture_multi
- cosocket API ngx.socket.tcp ngx.req.socket
- API ngx.sleep

Lua API　　HTTPS　　ssl_certificate_by_lua_block
ssl_session_fetch_by_lua_block　ssl_session_store_by_lua_block　3　　3
lua-resty-core　　ngx.ssl
lua-resty-core

Ngx_Lua

8-3　　Ngx_Lua　　Lua-Nginx-Module　　Wiki
Ngx_Lua　Nginx

8-3　Ngx_Lua

1 *_by_lua_file test.lua

```
# vim /usr/local/nginx_1.12.2/conf/lua/test.lua
 local ngx  = require "ngx"
-- ngx.get_phase()       Lua
 local phase = ngx.get_phase()
 ngx.log(ngx.ERR, phase, ':   Hello,world!')
```

2 Nginx

```
# init_by_lua_file    init_worker_by_lua_file      http
init_by_lua_file  /usr/local/nginx_1.12.2/conf/lua/test.lua;
init_worker_by_lua_file  /usr/local/nginx_1.12.2/conf/lua/test.lua;

server {
    listen       80;
    server_name  testnginx.com;

    location / {
        #
        body_filter_by_lua_file
/usr/local/nginx_1.12.2/conf/lua/test.lua;
        header_filter_by_lua_file
/usr/local/nginx_1.12.2/conf/lua/test.lua;
        rewrite_by_lua_file  /usr/local/nginx_1.12.2/conf/lua/test.lua;
        access_by_lua_file  /usr/local/nginx_1.12.2/conf/lua/test.lua;
        set_by_lua_file $test /usr/local/nginx_1.12.2/conf/lua/test.lua;
        log_by_lua_file  /usr/local/nginx_1.12.2/conf/lua/test.lua;
        # content_by_lua_file    balancer_by_lua_block
        # location         content_by_lua_file
        content_by_lua_file  /usr/local/nginx_1.12.2/conf/lua/test.lua;
    }
}
```

3 Nginx Nginx error.log

“ init ” init_by_lua_block “ init_worker ” init_worker_by_lua_block

init_worker 3 Nginx worker worker Lua Nginx 3 worker

```
    2018/06/04  19:00:23  [error]  12034#12034:  [lua]  test.lua:3:  init:
Hello,world!
    2018/06/04  19:00:23  [error]  21019#21019:  *914  [lua]  test.lua:3:
init_worker:  Hello,world!, context: init_worker_by_lua*
    2018/06/04  19:00:23  [error]  21020#21020:  *915  [lua]  test.lua:3:
init_worker:  Hello,world!, context: init_worker_by_lua*
    2018/06/04  19:00:23  [error]  21018#21018:  *916  [lua]  test.lua:3:
init_worker:  Hello,world!, context: init_worker_by_lua*
```

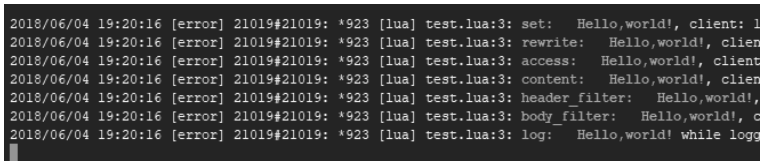
4

```
    curl -i http://testnginx.com/
    HTTP/1.1 200 OK
    Server: nginx/1.12.2
    Date: Mon, 04 Jun 2018 11:00:29 GMT
    Content-Type: application/octet-stream
    Transfer-Encoding: chunked
    Connection: keep-alive
```

5　　access.log

```
    2018/06/04 19:16:20 [error] 21019#21019: *920 [lua] test.lua:3: set:
Hello,world!, client: 10.19.64.210, server: testnginx.com, request: "GET
/test?ss HTTP/1.1", host: "testnginx.com"
    2018/06/04 19:16:20 [error] 21019#21019: *920 [lua] test.lua:3: rewrite:
Hello,world!, client: 10.19.64.210, server: testnginx.com, request: "GET
/test?ss HTTP/1.1", host: "testnginx.com"
    2018/06/04 19:16:20 [error] 21019#21019: *920 [lua] test.lua:3: access:
Hello,world!, client: 10.19.64.210, server: testnginx.com, request: "GET
/test?ss HTTP/1.1", host: "testnginx.com"
    2018/06/04 19:16:20 [error] 21019#21019: *920 [lua] test.lua:3: content:
Hello,world!, client: 10.19.64.210, server: testnginx.com, request: "GET
/test?ss HTTP/1.1", host: "testnginx.com"
    2018/06/04  19:16:20  [error]  21019#21019:  *920  [lua]  test.lua:3:
header_filter:  Hello,world!, client: 10.19.64.210, server: testnginx.com,
request: "GET /test?ss HTTP/1.1", host: "testnginx.com"
    2018/06/04  19:16:20  [error]  21019#21019:  *920  [lua]  test.lua:3:
body_filter:  Hello,world!, client: 10.19.64.210, server: testnginx.com,
request: "GET /test?ss HTTP/1.1", host: "testnginx.com"
    2018/06/04 19:16:20 [error] 21019#21019: *920 [lua] test.lua:3: log:
Hello,world!  while  logging  request,  client:  10.19.64.210,  server:
testnginx.com, request: "GET /test?ss HTTP/1.1", host: "testnginx.com"
```

access.log　　8-4



8-4 access.log

Ngx_Lua

- Ngx_Lua

```
init_by_lua_block
init_worker_by_lua_block
set_by_lua_block
rewrite_by_lua_block
access_by_lua_block
content_by_lua_block
header_filter_by_lua_block
body_filter_by_lua_block
log_by_lua_block
```

- Nginx
- http init_by_lua_block init_worker_by_lua_block
  HTTP

注意：init_by_lua_file lua_code_cache off HTTP
Lua

Ngx_Lua

9

Python PHP Java

Nginx

Nginx “ ” MySQL Redis

Nginx

注意： Nginx OpenResty

OpenResty Web

JSON Ngx_Lua JSON

C cjson

```
    #     wget     https://www.kyne.com.au/~mark/software/download/lua-cjson-
2.1.0.tar.gz
    # tar -zxvf lua-cjson-2.1.0.tar.gz
    # cd lua-cjson-2.1.0
    # make
    # cp cjson.so /usr/local/nginx_1.12.2/conf/c_package/
```

Ngx_Lua LuaJIT Lua make

```
    # make
    cc  -c  -O3  -Wall  -pedantic  -DNDEBUG    -I/usr/local/include  -fpic  -o
lua_cjson.o lua_cjson.c
```

```
lua_cjson.c:43:17: error: lua.h: No such file or directory
lua_cjson.c:44:21: error: lauxlib.h: No such file or directory
lua_cjson.c:192: error: expected ') ' before '*' token
lua_cjson.c:206: error: expected ') ' before '*' token
lua_cjson.c:218: error: expected ') ' before '*' token
lua_cjson.c:237: error: expected ') ' before '*' token
lua_cjson.c:266: error: expected ') ' before '*' token
lua_cjson.c:279: error: expected ') ' before '*' token
lua_cjson.c:288: error: expected ') ' before '*' token
lua_cjson.c:296: error: expected ') ' before '*' token
lua_cjson.c:304: error: expected ') ' before '*' token
```

lua-cjson-2.1.0 Makefile

“ LUA_INCLUDE_DIR= ” LuaJIT include

```
##### Build defaults #####
LUA_VERSION =       5.1
TARGET =            cjson.so
PREFIX =            /usr/local
#CFLAGS =            -g -Wall -pedantic -fno-inline
CFLAGS =            -O3 -Wall -pedantic -DNDEBUG
CJSON_CFLAGS =      -fpic
CJSON_LDFLAGS =     -shared
LUA_INCLUDE_DIR =   $(PREFIX)/include/luajit-2.1
LUA_CMODULE_DIR =   $(PREFIX)/lib/lua/$(LUA_VERSION)
LUA_MODULE_DIR =    $(PREFIX)/share/lua/$(LUA_VERSION)
LUA_BIN_DIR =       $(PREFIX)/bin
```

make csjon.so csjon.so

lua_package_cpath lua_package_cpath "${prefix}conf/c_package/?.so;;"

```
location = /test1 {
    content_by_lua_block {
        local  cjson = require "cjson"
        ngx.say(cjson.encode{a = 1, b = 2, c =3 })
    }
}
```

```
# curl  'http://testnginx.com/test1'
{"b":2,"a":1,"c":3}
```

Nginx　MySQL　lua-resty-mysql
worker

9.2.1　安装 lua-resty-mysql 模块

lua-resty-mysql releases　lua_package_path
resty　lib　mysql.lua　resty

lua-resty-mysql　resty　mysql.lua
local mysql = require "resty.mysql"

```
   #  wget  -S  https://codeload.github.com/openresty/lua-resty-mysql/tar.gz/
v0.21 -O lua-resty-mysql-0.21.tar.gz
   # tar -zxvf lua-resty-mysql-0.21.tar.gz
   #cp              lua-resty-mysql-0.21/lib/resty/mysql.lua              \
/usr/local/nginx_1.12.2/conf/lua_modules/resty/
```

9.2.2　读取 MySQL 数据

lua-resty-mysql　MySQL

```
   lua_package_path "${prefix}conf/lua_modules/?.lua;;";
   lua_package_cpath "${prefix}conf/c_package/?.so;;";

   server {
       listen       80;
       server_name  testnginx.com;
       default_type 'text/plain';

       location = /test {
           content_by_lua_block {
               local mysql = require "resty.mysql";
               local db, err = mysql:new();
               if not db then
                 ngx.say("failed to instantiate mysql: ", err);
                 return
               end
               --             1s
               db:set_timeout(1000) ;
               --     MySQL
               local ok, err, errcode, sqlstate = db:connect{
```

```
                    host = "10.19.40.113",
                    port = 3306,
                    database = "clairvoyant",
                    user = "ngx_test",
                    password = "ngx_test",
                    charset = "utf8",
                    max_packet_size = 2048 * 2048
                }
                --
                if not ok then
                  ngx.say("failed to connect: ", err, ": ", errcode, " ",
sqlstate);
                  return
                end

                --     SQL
                local res, err, errcode, sqlstate =
                  db:query("select id,host from  nginx_resource limit 2")

                --
                if not res then
                  ngx.say("bad result: ", err, ": ", errcode, ": ",
                          sqlstate, ".")
                  return
                end

                --        res   table              cjson           JSON
                local cjson = require "cjson";
                ngx.say("result: ", type(res));
                ngx.say("result: ", cjson.encode(res));

                --
                --                50               10s
                --               10s
                local ok, err = db:set_keepalive(10000, 50)
                if not ok then
                  ngx.say("failed to set keepalive: ", err);
                  return
                end
            }
        }
    }
```

```
    # curl 'http://testnginx.com/test'
    result: table
    result:
[{"host":"shop.zhe800.com","id":703},{"host":"shop.zhe800.com","id":705}]
```

指令：syntax: db, err = mysql:new()

MySQL nil

err

指令：syntax: ok, err, errcode, sqlstate = db:connect(options)

MySQL

- host MySQL IP
- port MySQL
- path MySQL UNIX socket
- database MySQL
- user MySQL
- password MySQL
- charset lua-resty-mysql MySQL MySQL
  big5 dec8 cp850 hp8 koi8r latin1 latin2 swe7 ascii ujis sjis hebrew tis620 euckr koi8u gb2312 greek cp1250 gbk latin5 armscii8 utf8 ucs2 cp866 keybcs2 macce macroman cp852 latin7 utf8mb4 cp1251 utf16 utf16le cp1256 cp1257 utf32 binary geostd8 cp932 eucjpms gb18030

- max_packet_size lua-resty-mysql MySQL 1MB
- compact_arrays true array-of-arrays

compact_arrays = true Nginx result

```
    # curl   'http://testnginx.com/test'
    result: table
    result: [[703,"shop.zhe800.com"],[705,"shop.zhe800.com"]]
```

指令：set_timeout

MySQL connect

MySQL set_timeout

指令：syntax: ok, err = db:set_keepalive(max_idle_timeout, pool_size)

1 2 worker

pool_size=20 2 worker

20×2=40 err

指令：syntax: times, err = db:get_reused_times()

0 0

指令：syntax: bytes, err = db:send_query(query)

MySQL

err

指令：syntax: res, err, errcode, sqlstate = db:read_result()

MySQL SQL err again db:read_result again err errcode MySQL

SQL read_result() db:read_result(3) 3 SQL

指令：syntax: res, err, errcode, sqlstate = db:query(query)

res err errcode MySQL

2

```
   local res, err, errcode, sqlstate = db:query("select id,host from nginx_resource " 2)
```

MySQL limit

指令：syntax: ok, err = db:close()

MySQL

```
    local ok, err = db:set_keepalive(10000, 50)
```

9.2.3　执行多条 SQL 语句

db:read_result　　SQL　　MySQL

```
    location = /test {
        content_by_lua_block {
            local mysql = require "resty.mysql";
            local db, err = mysql:new();
            if not db then
              ngx.say("failed to instantiate mysql: ", err);
              return
            end
            --              3s
            db:set_timeout(3000) ;
            local ok, err, errcode, sqlstate = db:connect{
               host = "10.19.40.113",
               port = 3306,
               database = "clairvoyant",
               user = "ngx_test",
               password = "ngx_test",
               charset = "utf8",
               max_packet_size = 2048 * 2048
            }
            --
            if not ok then
              ngx.say("failed  to  connect:  ",  err,  ":  ",  errcode,  "  ",
sqlstate);
              return
            end

            --     SQL        4   SQL
            sql1 = 'select id,host from  nginx_resource limit 1;';
            sql2 = 'select id from  nginx_resource limit 1;';
            sql3 = 'select host from  nginx_resource limit 1;';
            sql4 = 'select host from  nginx_resource limit 2;';

            -- SQL      ..
            local res, err, errcode, sqlstate =
            db:query(sql1 .. sql2 .. sql3 .. sql4)

            --                                                    SQL
            --     error.log
```

```
            if not res then
               ngx.log(ngx.ERR, "bad result #1: ", err, ": ", errcode, ": ",
sqlstate, ".")
               return ngx.exit(500)
            end
            local cjson = require "cjson";
            --
            ngx.say("result sql: " , cjson.encode(res))
            --                      err      again
            while err == "again" do
                res, err, errcode, sqlstate = db:read_result()
                if not res then
                ngx.log(ngx.ERR, "bad result sql",  ": ", err, ": ",
                errcode, ": ", sqlstate, ".")
                return ngx.exit(500)
                end
             --          SQL
                ngx.say("result sql", ": ", cjson.encode(res))
            end

            --
            --                 50            10000ms
            local ok, err = db:set_keepalive(10000, 50)
            if not ok then
              ngx.say("failed to set keepalive: ", err);
              return
            end
        }
    }
```

```
    # curl 'http://testnginx.com/test'
    result sql: [{"host":"shop.zhe800.com","id":703}]
    result sql: [{"id":703}]
    result sql: [{"host":"shop.zhe800.com"}]
    result sql: [{"host":"shop.zhe800.com"},{"host":"shop.zhe800.com"}]
```

SQL

SQL

db:set_timeout(3000) SQL

sql2 “ select sleep(4); ”

9.2.4　防止 SQL 注入

Ngx_Lua　　ngx.quote_sql_str　　SQL

指令：ngx.quote_sql_str

quoted_value = ngx.quote_sql_str(raw_value)

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

MySQL

URL　　a

```
local a = ngx.quote_sql_str(ngx.var.arg_a)
loal sql = 'select id,host from  nginx_resource where test= ' .. a
local res, err, errcode, sqlstate = db:query(sql)
```

Nginx　Redis　　lua-resty-redis　redis2-nginx-module　Ngx_Lua　　C　　Nginx　Nginx　worker　　lua-resty-redis　Ngx_Lua　　redis2-nginx-module

9.3.1　安装 lua-resty-redis

lua-resty-redis　lua-resty-mysql　　lib　　lua_package_path

```
# git clone https://github.com/openresty/lua-resty-redis.git
# cp lua-resty-redis/lib/resty/redis.lua \ /usr/local/nginx_1.12.2/conf/lua_modules/resty/
```

9.3.2　读/写 Redis

lua-resty-redis　Redis

```
location = /test {
    content_by_lua_block {
```

```
        local redis = require "resty.redis"

        --        Redis
        local red = redis:new()

    --             1s         Redis                  lua-resty-mysql
        red:set_timeout(1000)

        --     Redis
        local ok, err = red:connect("127.0.0.1", 6379)
        if not ok then
            ngx.say("failed to connect: ", err)
            return
        end
        --     set      Redis
        ok, err = red:set("test", "nginx")
        if not ok then
            ngx.say("failed to set test: ", err)
            return
        end
        local key_1 = 'test'
        --     get      Redis                  res
        local res, err = red:get(key_1)
        if not res then
            ngx.say("failed to get test: ", err)
            return
        end
        --             key              ngx.null
        if res == ngx.null then
            ngx.say(key_1," not found.")
            return
        end
        --
        ngx.say("test: ", res)

        --                100              10s    lua-resty-mysql
        local ok, err = red:set_keepalive(10000, 100)
        if not ok then
            ngx.say("failed to set keepalive: ", err)
            return
        end
        }
    }
```

```
# curl 'http://testnginx.com/test'
test: nginx
```

lua-resty-redis　lua-resty-mysql
Ngx_Lua　cosocket API

lua-resty-redis

指令：syntax: red, err = redis:new()

Redis　nil
err

指令：syntax: ok, err = red:connect(host, port, options_table?)

syntax: ok, err = red:connect("unix:/path/to/unix.sock", options_table?)

Redis

- host　Redis　IP
- port　Redis
- options_table
- unix:/path/to/unix.sock　Redis　socket

指令：syntax: ok, err = red:close()

Redis
local ok, err = red:set_keepalive(10000, 100)

指令：syntax: times, err = red:get_reused_times()

0　0

9.3.3　管道命令

Redis　TCP

```
location = /test {
    content_by_lua_block {
        local redis = require "resty.redis"
        --        Redis
```

```
local red = redis:new()

--             1s,
red:set_timeout(1000)

--     Redis
local ok, err = red:connect("127.0.0.1", 6379)
if not ok then
    ngx.say("failed to connect: ", err)
    return
end
--     pipeline
red:init_pipeline()
red:set("a", "1")
red:set("b", "2")
--
red:hset("a s s")
red:get("a")
red:get("b")
red:get("c")

--               pipeline
local results, err = red:commit_pipeline()
if not results then
    ngx.say("failed to commit the pipelined requests: ", err)
    return
end

--            table
for i, res in ipairs(results) do
    if type(res) == "table" then
        --               res[1]      false
        if res[1] == false then
            ngx.say("failed to run command ", i, ": ", res[2])
        else
        end
    else
      --
      ngx.say('type:',type(res),'---res: ',res)
    end
end

local ok, err = red:set_keepalive(10000, 100)
if not ok then
```

```
            ngx.say("failed to set keepalive: ", err)
            return
        end

        }
    }
}
```

red:init_pipeline()　　pipeline
red:commit_pipeline()

```
# curl   'http://testnginx.com/test'
type:string---res: OK
type:string---res: OK
failed to run command 3: ERR wrong number of arguments for 'hset' command
type:string---res: 1
type:string---res: 2
type:userdata---res: null
```

- 　　set　　ok
- 　　hset
- 　get　　null

9.3.4　密码登录

Redis

```
--     Redis
local ok, err = red:connect("127.0.0.1", 6379)
if not ok then
    ngx.say("failed to connect: ", err)
    return
end
--   red:connect
local res, err = red:auth("testpasswd")
if not res then
    ngx.say("failed to authenticate: ", err)
    return
end
```

9.3.5 其他执行命令

Redis lua-resty-redis
lua-resty-redis redis_cli Redis 9-1

表 9-1 lua-resty-redis 与 redis_cli 指令格式的对比

lua-resty-redis 指令格式	redis_cli 指令格式

```
--     Redis
local ok, err = red:connect("127.0.0.1", 6379)
if not ok then
    ngx.say("failed to connect: ", err)
    return
end
--           Redis
local res, err = red:auth("testpasswd")
if not res then
    ngx.say("failed to authenticate: ", err)
    return
end
--
local count, err = red:get_reused_times()
ngx.say('get_reused_times: ',count)

red:init_pipeline()
red:get("a")
red:get("b")
local results, err = red:commit_pipeline()
if not results then
    ngx.say("failed to commit the pipelined requests: ", err)
    return
end
for i, res in ipairs(results) do
    if type(res) == "table" then
        --              res[1]     false
        if res[1] == false then
            ngx.say("failed to run command ", i, ": ", res[2])
        else
        end
    else
      --
      ngx.say('type: ',type(res),'---res: ',res)
    end
end
local ok, err = red:set_keepalive(10000, 10)
if not ok then
    ngx.say("failed to set keepalive: ", err)
    return
end

}
```

```
    }
```

```
# curl   'http://testnginx.com/test'
get_reused_times: 0
type:string---res: 1
type:string---res: 2
[root@testnginx ~]# curl   'http://testnginx.com/test'
get_reused_times: 1
type:string---res: 1
type:string---res: 2
[root@testnginx ~]# curl   'http://testnginx.com/test'
get_reused_times: 2
type:string---res: 1
type:string---res: 2
[root@testnginx ~]# curl   'http://testnginx.com/test'
get_reused_times: 3
type:string---res: 1
type:string---res: 2
[root@testnginx ~]# curl   'http://testnginx.com/test'
get_reused_times: 4
type:string---res: 1
type:string---res: 2
```

get_reused_times()

10s 0 set_keepalive

10s 10s

10s

Redis get_reused_times 0

Redis get_reused_times=0

```
# curl   'http://testnginx.com/test'
failed to authenticate: ERR invalid password
```

9.4.2　读/写分离

/

Redis /

```
local cjson = require "cjson"
local redis = require "resty.redis"
--
local red = redis:new()
red:set_timeout(1000)
local ok, err = red:connect("127.0.0.1", 6379)
if not ok then
    ngx.say("failed to connect: ", err)
    return
end
--
local red_slave1 = redis:new()
red_slave1:set_timeout(1000)
local ok, err = red_slave1:connect("127.0.0.1", 6380)
if not ok then
    ngx.say("failed to connect: ", err)
    return
end
```

/ Redis

9.4.3　分离配置文件和代码

require

db_config.lua Lua

```
--        table      _M
local _M = {}
--   MySQL          _M
local _M.mysql_config = {
    host = "10.19.40.113",
    port = 3306,
    database = "clairvoyant",
    user = "ngx_test",
```

```
        password = "ngx_test",
        charset = "utf8",
        max_packet_size = 2048 * 2048
    }
    return _M
```

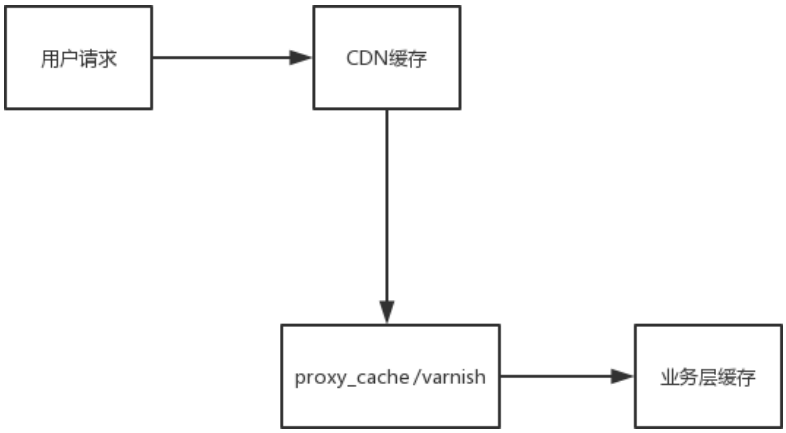
require

```
    --
    local db_config = require "db_config"
    --        mysql_config
    local ok, err, errcode, sqlstate = db:connect(db_config.mysql_config)
```

Nginx MySQL Redis Ngx_Lua
Nginx Couchbase MongoDB Memcached

10

10-1



10-1

CDN proxy_cache

Redis Memcached Couchbase

Ngx_Lua

key/value TCP

Ngx_Lua

worker Nginx worker
worker worker

10.1.1 创建共享内存区域

1

lua_shared_dict

lua_shared_dict <name> <size>

http

name size Lua

shared_test 1MB

```
lua_shared_dict shared_test 1M;
```

lua_shared_dict size KB MB 12KB
12KB size 12KB 8KB

```
nginx: [crit] ngx_slab_alloc() failed: no memory
```

size 8KB

```
nginx: [emerg] invalid lua shared dict size "2k" in /usr/local/nginx_
1.12.2/conf/nginx.conf
```

lua_shared_dict Nginx
500MB Nginx
Nginx worker 500MB 500MB +Nginx
500MB

1 worker worker
worker

2 LRU Least

Recently Used

3　Nginx

10.1.2　操作共享内存

/

```
--
lua_shared_dict shared_test 1m;
server {
    listen 80;
    server_name testnginx.com;
    location /set {
        content_by_lua_block {
            --                Lua
            local shared_test = ngx.shared.shared_test
            --              URL    a          Redis   set
            shared_test:set("a",ngx.var.arg_a)
            ngx.say("STORED")
        }
    }
    location /get {
        content_by_lua_block {
            --                Lua
            local shared_test = ngx.shared.shared_test
            --                      a          Redis   get
            ngx.say(shared_test:get("a"))
        }
    }
}
```

a=123　　/set　location　　set

/get　　a

```
# curl   'http://testnginx.com/set?a=123'
STORED
# curl   'http://testnginx.com/get?a'
123
```

Ngx_Lua

注意：　　DICT　　lua_shared_dict

指令：ngx.shared.DICT

dict = ngx.shared.DICT　dict = ngx.shared[name_var]

init_by_lua* init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_ lua* ssl_certificate_by_lua* ssl_session_fetch_by_lua* ssl_session_store_by_lua*

Lua

指令：ngx.shared.DICT.set

success, err, forcible = ngx.shared.DICT:set(key, value, exptime?, flags?)

key/value value Lua nil table

3

- success set
- err success false err
- forcible

LRU forcible true

key/value set

- exptime key/value
- flags set 0

number flags set

指令：ngx.shared.DICT.safe_set

ok, err = ngx.shared.DICT:safe_set(key, value, exptime?, flags?)

set

“no memory” nil

指令：ngx.shared.DICT.get

value, flags = ngx.shared.DICT:get(key)

key value value key nil

flags set flags 0

指令：ngx.shared.DICT.get_stale

value, flags, stale = ngx.shared.DICT:get_stale(key)

get

flags　　set　　0

stale　　value　　true

指令：ngx.shared.DICT.add

success, err, forcible = ngx.shared.DICT:add(key, value, exptime?, flags?)

set　　key　　add　　err

“exist”　　key　　add　　key/value

指令：ngx.shared.DICT.safe_add

ok, err = ngx.shared.DICT:safe_add(key, value, exptime?, flags?)

set

“no memory”　　nil

指令：ngx.shared.DICT.replace

success, err, forcible = ngx.shared.DICT:replace(key, value, exptime?, flags?)

set　　key　　key/value　　key

指令：ngx.shared.DICT.delete

ngx.shared.DICT:delete(key)

key

指令：ngx.shared.DICT.incr

newval, err, forcible? = ngx.shared.DICT:incr(key, value, init?)

key　　value　　newval

key　　init　　“not found”

err　　key　　init　　number　　init+value　　key

注意：　　key　value　init　　number

指令：ngx.shared.DICT.flush_all

ngx.shared.DICT:flush_all()

get_stale

指令：ngx.shared.DICT.flush_expired

flushed = ngx.shared.DICT:flush_expired(max_count?)

max_count 0
0

指令：ngx.shared.DICT.get_keys

keys = ngx.shared.DICT:get_keys(max_count?)

table

max_count 1024 0
0
worker

1 key/value

set forcible = true safe_set err = "no memory"

2 key/value

get get_stale flags set
flags

3 incr

init_by_lua* Nginx

4 key

get_keys 1024 key get_keys(0)
key MySQL limit select * from table

```
location /set {
    content_by_lua_block {
        local shared_test = ngx.shared.shared_test
        --
        local newval, err = shared_test:incr("incr_init_n",1,2)
        ngx.say("newval:",newval, " err:",err)  --      newval:3 err:nil
        shared_test:set("a",ngx.var.arg_a,100,2001)
        shared_test:set("b",ngx.var.arg_a,100,2001)
        local  success, err, forcible = shared_test:set("d",
        ngx.var.arg_a, 100,2001)
        --     success:true err:nil forcible:false
        ngx.say("success:",success, " err:",err, " forcible:",forcible)
        local  ok, err = shared_test:safe_set("c",ngx.var.arg_a,100,
        2001)ngx.say("ok:",ok," err:",err)       --     ok:true err:nil
        local value, flags, stale = shared_test:get_stale("a")
        --     value:123s flags:2001 stale:false
        --                   stale   false
        ngx.say("value:",value," flags:",flags," stale:",stale)
        --             2   key
        local getall_key = shared_test:get_keys(2)
        if type(getall_key) == 'table' then
           ngx.say(#getall_key)                  --     key
           for _, k in ipairs(getall_key) do
                                                 --       2
             -- key:b value:123s2001  key:d value:123s2001
             ngx.say("key:",k ," value:",shared_test:get(k))
           end
        end
        shared_test:flush_all()                  --
        local getall_key = shared_test:get_keys()
        --          key       type   table  llen_list: 0
        ngx.say("type  "  type(getall_key),  " llen_list: ",#getall_key)
    }
}
```

10.1.3　制造消息队列

ngx.shared.DICT　　10-1

表 10-1　与消息队列相关的指令及其使用方式

指　令	说　明
length, err = ngx.shared.DICT:lpush(key, value)	string　number　value　key key　key　key nil　err　"value not a list"

续表

指　令	说　明
length, err = ngx.shared.DICT:rpush(key, value)	lpush
val, err = ngx.shared.DICT:lpop(key)	key 1 val key nil key nil err "value not a list"
val, err = ngx.shared.DICT:rpop(key)	lpop
syntax: len, err = ngx.shared.DICT:llen(key)	key key 0 key nil err "value not a list"

```
lua_shared_dict shared_test_1 1m;
server {
    listen 80;
    server_name testnginx.com;
    location /lpush {
        content_by_lua_block {
            local shared_test_1 = ngx.shared.shared_test_1
            -- 1 lpush
            local newval, err = shared_test_1:incr("incr_init_n",1,0)
            -- length
            local length, err = shared_test_1:lpush("push_abc", newval)
            --
            ngx.say("length:",length," lpush_value:",newval)
        }
    }
    location /lpop {
        content_by_lua_block {
            local shared_test_1 = ngx.shared.shared_test_1
            -- val
            local val, err = ngx.shared.shared_test_1:lpop("push_abc")
            -- key
            local len, err = ngx.shared.shared_test_1:llen("push_abc")
            --
            ngx.say("length:",len, " val:" ,val)
        }
    }
}
```

HTTP

```
# curl   'http://testnginx.com/lpush'
length:1 lpush_value:1
```

```
[root@testnginx ~]# curl   'http://testnginx.com/lpush'
length:2 lpush_value:2
[root@testnginx ~]# curl   'http://testnginx.com/lpush'
length:3 lpush_value:3
[root@testnginx ~]# curl   'http://testnginx.com/lpop'
length:2 val:3
[root@testnginx ~]# curl   'http://testnginx.com/lpop'
length:1 val:2
[root@testnginx ~]# curl   'http://testnginx.com/lpop'
length:0 val:1
[root@testnginx ~]# curl   'http://testnginx.com/lpop'
length:0 val:nil
```

key

1 3

2 4 val

3 2 1

3 1 nil

pop HTTP

- pop HTTP
-

for

10.1.4 lua-resty-core

key lua-resty-core

lua-resty-core

OpenResty

lua-resty-core

指令：ngx.shared.DICT.ttl

ttl, err = ngx.shared.DICT:ttl(key)

key

指令：ngx.shared.DICT.expire

success, err = ngx.shared.DICT:expire(key, exptime)

key

指令：ngx.shared.DICT.free_space

free_page_bytes = ngx.shared.DICT:free_space()

10.1.5 配置环境

ngx.shared.DICT.*

1 dict = ngx.shared.DICT

init_by_ lua* init_worker_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch _by_lua* ssl_session_ store_by_lua*

2 ngx.shared.DICT:get_stale ngx.shared.DICT:get

init_by_lua*, init_worker_by_lua*

set_by_lua* rewrite_by_lua* access_by_lua* content_by_lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_lua* ssl_certificate_by_lua* ssl_session_fetch _by_lua* ssl_session_store_by_lua*

3 ngx.shared.DICT.*

init_worker_by_lua*

init_by_lua* set_by_lua* rewrite_by_lua* access_by_lua* content_by_ lua* header_filter_by_lua* body_filter_by_lua* log_by_lua* ngx.timer.* balancer_by_ lua* ssl_certificate_by_lua* ssl_session_fetch _by_lua* ssl_session_store_by_lua*

Redis

I/O

10.1　　worker

1　worker

2　　Lua　　nil　　table

3　　CPU

Ngx_Lua

lua_shared_dict

lua-resty-lrucache

10.2.1　安装 lua-resty-lrucache

lua-resty-lrucache　　Ngx_Lua

1　　table　　value

2　　key

3　　worker　　worker　　key

lua_shared_dict

1　　worker

worker

2

3　　Nginx　　lua_shared_dict

lua-resty-lrucache

lua-resty

```
# git clone https://github.com/openresty/lua-resty-lrucache.git
# cp -r lua-resty-lrucache/lib/resty/lrucache* \
 /usr/local/nginx_1.12.2/conf/lua_modules/resty/
```

10.2.2　使用 lua-resty-lrucache 进行缓存的方法

lua-resty-lrucache

```
local lrucache = require "resty.lrucache"
local lrucache = require "resty.lrucache.pureffi"
```

lua-resty-lrucache 2 lua_package_path

resty.lrucache

resty.lrucache.pureffi key

test_m.lua lua_package_path

```
local _M = {}
local lrucache = require "resty.lrucache"
--                  1000   key
local cache, err = lrucache.new(1000)
if not cache then
    return error("failed to create the cache: " .. (err or "unknown"))
end
--                    key/value
local function mem_set()
    -- set()                        key  value         2s
    cache:set("a", 19, 2)
    cache:set("b", {"1","2","3"},0.001)  --        table
    return
end
--                    value   a   value          a   nil
--       value                   stale_data           value
local function mem_get(key)
    local a,stale_data = cache:get(key)
    return a,stale_data
end
function _M. fromcache ()
    --     a
local a,stale_data  =  mem_get("a")
    --     a           a
    if a then
       ngx.say("a: ", a)
    --     a        stale_data                value
    --                               value
    elseif stale_data then
       ngx.say("a        : " , stale_data)
       mem_set()
```

```
        local a_again  =  mem_get("a")
        ngx.say("a: ", a_again )
    --     a    stale_data                              value
    else
        ngx.say("no found a")
        mem_set()
        local a_again  =  mem_get("a")
        ngx.say("a: ", a_again )

    end
end
return _M
```

nginx.conf

```
location / {
    content_by_lua_block {
        --
        require("test_m").fromcache()
    }
}
```

Nginx

```
# curl   'http://testnginx.com/'
no found a
a: 19
[root@testnginx ~]# curl   'http://testnginx.com/'
a: 19
[root@testnginx ~]# curl   'http://testnginx.com/'
a: 19
[root@testnginx ~]# curl   'http://testnginx.com/'
a        : 19
a: 19
```

1　1　a　“ no found a ”

2　2　value

3　3　value

4　4

value　value

Nginx　restart　a

lua-resty-lrucache

指令：new

cache, err = lrucache.new(max_items [, load_factor])

nil err

max_items key

key

load_factor resty.lrucache.pureffi FFI Foreign Function Interface hash 0.1~1 0.5 hash

指令：set

cache:set(key, value, ttl)

key/value ttl 0

0.001s

指令：get

data, stale_data = cache:get(key)

key key nil

stale_data

指令：delete

cache:delete(key)

key

指令：flush_all

cache:flush_all(key)

10.1　　10.2

rewrite　　content

Ngx_Lua　　Lua API

Lua API　　Lua API

ngx.ctx

10.3.1　ngx.ctx 的使用

指令：ngx.ctx

init_worker_by_lua*　　set_by_lua*　　rewrite_by_lua*　　access_by_lua*

5 ?) r - content_by_lua* 5 . m i t . (□ 2□ g□ n

```
            --
            ngx.header["test"] = ngx.ctx.test .. ' world!'
        }
    }
```

```
    header_filter_by_lua_block {
        ngx.header["test"] =  ' world!'
    }
    content_by_lua_block {
        ngx.ctx.test = "nginx"
        --
        ngx.exec("/subq")
    }
}
```

```
# curl   -i 'http://testnginx.com/'
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Mon, 18 Jun 2018 05:36:38 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive
test: not test world!

nil
```

ngx.ctx.test

注意：　ngx.ctx

ngx.ctx　　ngx.ctx

16

8.3

```
lua_shared_dict white_list_ip 1m;   #
server {
    listen       80;
    server_name  testnginx.com;
    location / {
        access_by_lua_block {
            local ngx = require "ngx";
            local white_list_ip = ngx.shared.white_list_ip
            --           IP         Nginx                  CDN
            --          real_ip              IP
            local ip  = ngx.var.remote_addr
```

```
            --                    IP
            local value, flags = white_list_ip:get(ip)
            --          IP                    access_by_lua_block
            --
            if value then
                ngx.exit(ngx.OK)
            else
                ngx.exit(ngx.HTTP_FORBIDDEN)-                    403
            end
        }
        echo 'ok';
    }

  --    HTTP
     location = /white_list_ip_op {
         default_type 'text/plain';
         content_by_lua_block {
              local ngx = require "ngx"
              local white_list_ip = ngx.shared.white_list_ip
              local op = ngx.var.arg_op
              local ip = ngx.var.arg_ip
              --
              if op == 'add' then
                  white_list_ip:set(ip,'1')
              --
              elseif op == 'del' then
                  white_list_ip:delete(ip)
              end
              --
              local ds = white_list_ip:get_keys()
              if type(ds) == 'table' then
                  for _, k in ipairs(ds) do
                      ngx.say("      ip :  ",k)
                  end
              end
            }
      }

}
```

```
#  curl 'http://testnginx.com/white_list_ip_op?op=add&ip=10.19.64.210'
     ip :  10.19.64.210
```

IP　　10.19.64.210　　testnginx.com

```
#  curl -i http://testnginx.com/
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Tue, 29 May 2018 09:43:30 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

ok
```

```
#  curl -i 'http://testnginx.com/white_list_ip_op?op=del&ip=10.19.64.210'
HTTP/1.1 200 OK
Server: nginx/1.12.2
Date: Tue, 29 May 2018 09:44:11 GMT
Content-Type: text/plain
Transfer-Encoding: chunked
Connection: keep-alive
```

IP　　10.19.64.210　　testnginx.com

```
#  curl -i http://testnginx.com/
HTTP/1.1 403 Forbidden
Server: nginx/1.12.2
Date: Tue, 29 May 2018 09:45:21 GMT
Content-Type: text/html
Content-Length: 169
Connection: keep-alive

<html>
<head><title>403 Forbidden</title></head>
<body bgcolor="white">
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx/1.12.2</center>
</body>
</html>
```

Nginx　　master

- A　　IP　　MySQL　　/　MySQL　Lua
  MySQL　　Nginx　　master

init_worker_by_lua MySQL

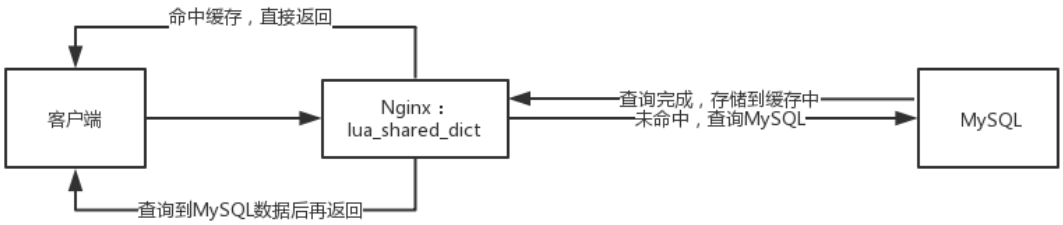
- B Nginx
  master init_by_lua_block IP

NoSQL MySQL
Ngx_Lua MySQL

10.5.1 从数据库获取数据

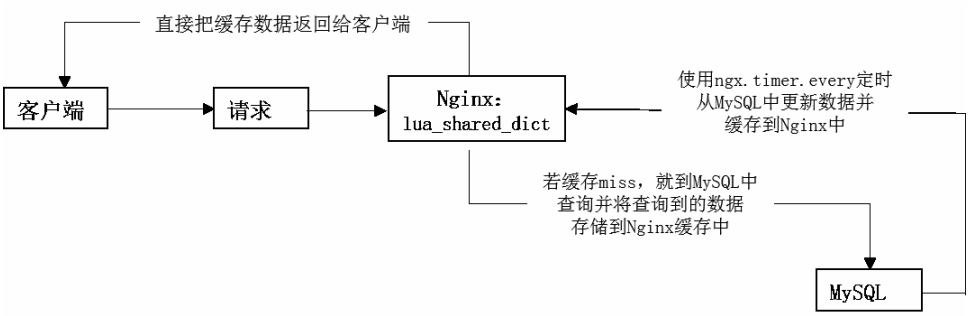
MySQL Ngx_Lua 3

A key 10-2 key



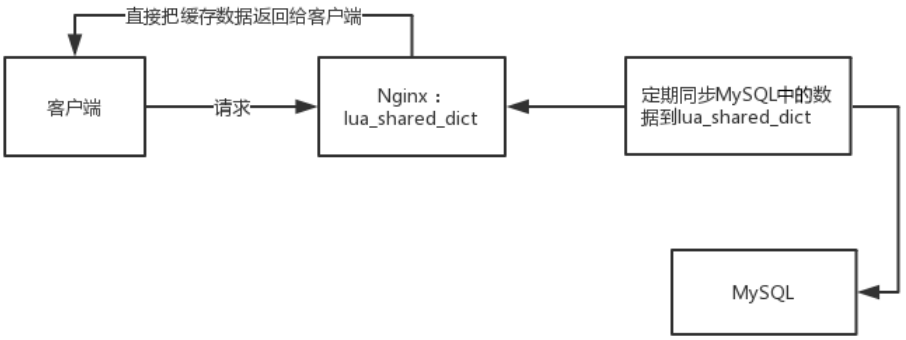
10-2 key

B key 10-3 key



10-3 key

C　　key　　Nginx
miss　　Nginx
MySQL　　10-4　　key



10-4　　key

A　　B　　B
miss　　C　　B
C　　MySQL

3　　lua_shared_dict　　lua-resty-lrucache
lua-resty-lrucache

db_op　　MySQL　　db_op.lua
lua_package_path

```
local _DB =  {}

--                      SQL
function _DB.getMySQL(sql)
            local MySQL = require "resty.MySQL";
            local db, err = MySQL:new();
            if not db then
              ngx.say("failed to instantiate MySQL: ", err);
              return
            end
            --              5s
            db:set_timeout(5000) ;
            --     MySQL
            local ok, err, errcode, sqlstate = db:connect{
               host = "10.19.10.113",
               port = 3306,
```

```
                    database = "clairvoyant",
                    user = "ngx_test",
                    password = "ngx_test",
                    charset = "utf8",
                    max_packet_size = 2048 * 2048

                }
                --
                if not ok then
                  ngx.say("failed to connect: ", err, ": ", errcode, " ",
sqlstate);
                  return
                end

                --     SQL
                local sql = sql
                local res, err, errcode, sqlstate =
                  db:query(sql)

                if not res then
                  ngx.say("bad result: ", err, ": ", errcode, ": ", sqlstate,
".")
                  return
                end

                ngx.log(ngx.ERR,db:get_reused_times(),err)
                local ok, err = db:set_keepalive(10000, 10)
                if not ok then
                  ngx.say("failed to set keepalive: ", err);
                  return
                end
                return res
    end

    return _DB
```

host_deny Ngx_Lua

host_deny.lua lua_package_path

```
    local _M = {}

    local lrucache = require "resty.lrucache"
    --     db_op             SQL
    local db_op = require("db_op")
    local cache, err = lrucache.new(1000) --              1000   key
```

```
    if not cache then
        return error("failed to create the cache: " .. (err or "unknown"))
    end

    local function mem_set(host)
        --     Lua                  host          SQL
        local  sql  =  string.format([[select  sleep(3),host  from    nginx_
resource where host =  '%s'  limit 1]] , host)
        --     SQL
        local res = db_op.getMySQL(sql)
        if type(res) == 'table' then
            for i, data  in ipairs(res) do
                --                          'find'
                --                          key   host
                cache:set(data["host"],'find',5)

            end
        end
        return
    end

    local function mem_get(host)
        local res_host,stale_data = cache:get(host)
        if res_host then
           return res_host
        elseif stale_data then
        --
        --      MySQL
           mem_set(host)
           res_host  = cache:get(host)
           return res_host

        else
        --             SQL
           mem_set(host)
           res_host  = cache:get(host)
           return res_host
        end
    end
    function _M.fromcache(host)
        --           URL   Host
        local res_host  =  mem_get(host)
        return res_host
    end
```

```
return _M
```

Nginx Host

```
server {
    listen 80;
    location / {
        access_by_lua_block {
            --     host_deny
            local host_deny = require "host_deny"
            local ngx = require "ngx"
            local host = ngx.var.host
            --     host_deny      fromcache        host
            local white_host = host_deny.fromcache(host)

            --                      403
            if not white_host then
                 ngx.exit(ngx.HTTP_FORBIDDEN)
            else
                 ngx.exit(ngx.OK)
            end

        }
        content_by_lua_block {
            ngx.say("hello world!!!")
        }
    }
}
```

403

200

```
# curl   -i 'http://testnginx.com/' -H 'Host: a.test.com'
HTTP/1.1 403 Forbidden
Server: nginx/1.12.2
Date: Mon, 18 Jun 2018 09:24:04 GMT
Content-Type: text/html
Content-Length: 169
Connection: keep-alive

[root@testnginx  ~]#  curl      -i  'http://testnginx.com/'  -H  'Host:
shop.zhe800.com'
HTTP/1.1 200 OK
```

```
Server: nginx/1.12.2
Date: Mon, 18 Jun 2018 09:23:31 GMT
Content-Type: application/octet-stream
Transfer-Encoding: chunked
Connection: keep-alive

hello world!!!
```

miss　　MySQL

10.5.2　避免出现因缓存失效引起的“风暴”

lua-resty-lock

ngx_http_proxy_module　　proxy_cache_lock

Nginx　　lua-resty-lock　　OpenResty

```
# wget -S https://codeload.github.com/openresty/lua-resty-lock/tar.gz/
v0.07 -O lua-resty-lock_0.07.tar.gz
# tar -zxvf lua-resty-lock_0.07.tar.gz
# cp lua-resty-lock-0.07/lib/resty/lock.lua \
  /usr/local/nginx_1.12.2/conf/lua_modules/resty
```

注意：　　Nginx　　resty.core　　lua-resty-lock 0.07　　lua-resty-lock　　resty.core

Wiki　　https://github.com/ openresty/ lua-resty-lock

10.5.1　　SQL

sleep(3)　　MySQL　　3s

db_op　　MySQL　　db_op　　10.5.1

host_deny.lua

```
local _M = {}

--     db_op             SQL
local db_op = require("db_op")

-- db_locks              key                    key
--           cache                            key/value
local db_locks=  ngx.shared.db_locks
```

```
    local cache=  ngx.shared.db_cache

    local function get_MySQL(host)
        --     MySQL                                          MySQL
        -- SQL        sleep 3s
        --     MySQL
        local  sql  =  string.format([[select  sleep(3),host  from    nginx_
resource where host =  '%s'  limit 1]] , host)
        local res = db_op.getMySQL(sql)
        if res[1] then
            local value = res[1]["host"] or nil
            return value
        end
        return nil
    end

    local function lock_db(key)
        --
    local resty_lock = require "resty.lock"
        --            db_locks                key
        local lock, err = resty_lock:new("db_locks")
        if not lock then
            ngx.log(ngx.ERR,err)
            return nil,"failed to create lock: " .. err
        end
        --          key
        local elapsed, err = lock:lock(key)
        if not elapsed then
            ngx.log(ngx.ERR,err)
            return nil,"failed to acquire the lock: " .. err
        end
        --
    ngx.log(ngx.ERR,elapsed)
        --
        --
        local val, err = cache:get(key)
    if val then
        --
            local ok, err = lock:unlock()
            if not ok then
            ngx.log(ngx.ERR,err)
                return nil,"failed to unlock: " .. err
            end
            return val
```

```
    end
    --   MySQL
    local val = get_MySQL(key)
if not val then
    --
        local ok, err = lock:unlock()
        if not ok then
        ngx.log(ngx.ERR,err)
            return nil,"failed to unlock: " .. err
        end
        --        key                    MySQL
        --             MySQL
        --                          null
        local ok,err = cache:set(key,'null',1)  -- 1            1s
        return 'null'  --        null
end
    --          val
    local ok, err = cache:set(key, val,3)
if not ok then
    --     set
        local ok, err = lock:unlock()
        if not ok then
            return nil,"failed to unlock: " .. err
        end
        return nil,"failed to update shm cache:  " .. err
    end
    --
    local ok, err = lock:unlock()
    if not ok then
        return nil,"failed to unlock: " .. err
    end
    return  val
end
local function mem_get(host)
    local res_host = cache:get(host)
    if res_host then
       return res_host
    else
    --
       local res_host = lock_db(host)
       return res_host
    end
end
function _M.fromcache(host)
```

```
    --               host
    local res_host  =  mem_get(host)
    return res_host
end

return _M
```

Host Host

MySQL key key

注 意： lua_shared_dict

lua_shared_dict lua-resty-lrucache

nginx.conf

```
--
lua_shared_dict db_locks 1m;
--
lua_shared_dict db_cache 5m;
server {
    listen 80;
    location / {
        access_by_lua_block {
            local host_deny = require "host_deny"
            local ngx = require "ngx"
            local host = ngx.var.host
            local white_host = host_deny.fromcache(host) or nil
            if not white_host then
                 ngx.exit(ngx.HTTP_FORBIDDEN)
            else
                 ngx.exit(ngx.OK)
            end

        }
        content_by_lua_block {
            ngx.say("hello world!!!")
        }
    }
}
```

5

```
# webbench -c 5 -t 10  'http://www.zhe800.com/'
```

error.log

```
   2018/06/19 19:17:56 [error] 8318#8318: *18671259 [lua] host_deny.lua:38:
lock_db(): 0, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:00 [error] 8318#8318: *18671262 [lua] host_deny.lua:38:
lock_db(): 3.511, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:00 [error] 8318#8318: *18671261 [lua] host_deny.lua:38:
lock_db(): 3.511, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:00 [error] 8318#8318: *18671263 [lua] host_deny.lua:38:
lock_db(): 3.511, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:00 [error] 8318#8318: *18671264 [lua] host_deny.lua:38:
lock_db(): 3.511, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:02 [error] 8318#8318: *18694720 [lua] host_deny.lua:38:
lock_db(): 0, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:05 [error] 8318#8318: *18694721 [lua] host_deny.lua:38:
lock_db(): 3.011, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:05 [error] 8318#8318: *18694722 [lua] host_deny.lua:38:
lock_db(): 3.011, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:05 [error] 8318#8318: *18694723 [lua] host_deny.lua:38:
lock_db(): 3.011, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
   2018/06/19 19:18:05 [error] 8318#8318: *18694724 [lua] host_deny.lua:38:
lock_db(): 3.011, client: 10.19.48.161, server: , request: "GET / HTTP/1.0",
host: "www.zhe800.com"
```

- lock_db()　　3s　　sleep(3)
- 　　1
- 　　3s　　key　　ngx.log(ngx.ERR,elapsed)

注意：　　MySQL
MySQL　　sleep

　　new

```
new
```

obj, err = lock:new(dict_name, opts?)

dict_name Nginx

opts table

- exptime 30s 0.001s
- timeout timeout
  exptime 0
- step 0.001s
  0.001s
  ratio
- ratio 2
  max_step
- max_step 0.5s

Ngx_Lua
“ ” “ ”

11

upstream Nginx Nginx
Nginx

1 Nginx

2

3

upstream

1 Nginx upstream
Nginx

2 upstream
upstream Nginx

3 upstream
upstream Nginx Nginx
Nginx
upstream

upstream

“ ”

1 upstream Nginx

2

3

4 Nginx

upstream

ngx_http_dyups_module API upstream
Nginx ip_hash keepalive upstream

11.2.1 安装 ngx_http_dyups_module

ngx_http_dyups_module C Nginx

```
# git clone git://github.com/yzprofile/ngx_http_dyups_module.git
# ./configure --add-module=/path/ngx_http_dyups_module
```

11.2.2 动态管理 upstream

ngx_http_dyups_module upstream nginx.conf
Nginx

```
upstream test_12 {
    server 127.0.0.1:8001;
}
upstream test_34 {
    server 127.0.0.1:8001;
}

server {
```

```
        listen 8000;
    location / {
            #     IP
            allow 127.0.0.1;
            deny  all;

            #         upstream
            dyups_interface;
        }
    }
    server {
        listen 80;
        location / {
            proxy_set_header     Host $host;
            proxy_set_header     X-Real-IP $remote_addr;
            proxy_set_header     X-Forwarded-For $proxy_add_x_forwarded_for;

            #   test_ser      $ups  test_ser   upstream   name
            #         $ups   proxy_pass      upstream
            set $ups test_ser;
            proxy_pass http://$ups;
        }
    }
```

upstream　　test_ser

```
    # curl 127.0.0.1:8000/list
    test_12
    test_34
```

test_ser　　upstream　　test_ser

upstream

```
    # curl -d "server 127.0.0.1:81 weight=10 max_fails=5 fail_timeout=10;
server 127.0.0.1:83 weight=20 max_fails=7 fail_timeout=10;" 127.0.0.1:8000/
upstream/test_ser

    success
```

upstream

```
    # curl 127.0.0.1:8000/detail
    test_12
    server  127.0.0.1:81  weight=20  max_fails=10  fail_timeout=5  backup=0
down=0

    test_34
```

```
    server  127.0.0.1:111  weight=20  max_fails=10  fail_timeout=5  backup=0
down=0
    server  127.0.0.1:821  weight=20  max_fails=10  fail_timeout=5  backup=0
down=0

    test_servers
    server  10.19.48.162:81  weight=10  max_fails=5  fail_timeout=10  backup=0
down=0

    test_ser
    server  127.0.0.1:81  weight=10  max_fails=5  fail_timeout=10  backup=0
down=0
    server  127.0.0.1:83  weight=20  max_fails=7  fail_timeout=10  backup=0
down=0
```

upstream

```
    # curl -i -X DELETE 127.0.0.1:8081/upstream/test_ser
```

Nginx upstream

upstream

dyups 11-1

表 11-1 dyups 支持的接口说明

请求方法	HTTP 接口	用 途
GET	/detail	upstream
GET	/list	upstream name
GET	/upstream/name	upstream IP
POST	/upstream/name	upstream IP
DELETE	/upstream/name	upstream

Ngx_Lua Ngx_Lua upstream

Wiki

11.2.3 确保 upstream 数据的完整性

ngx_http_dyups_module Nginx

Nginx

ngx_http_dyups_module /detail
upstream Nginx upstream upstream
upstream include Nginx
Nginx upstream Nginx upstream

ngx_http_dyups_module Ngx_Lua Ngx_Lua
Ngx_Lua https://github.com/
yzprofile/ngx_http_dyups_module/blob/master/README.md

Nginx

- upstream
- upstream
- upstream

nginx-upsync-module ngx_http_dyups_module
upstream Nginx nginx-upsync-module upstream
Consul Consul

11.3.1　安装 nginx-upsync-module 和 Consul

C Nginx

```
# git clone https://github.com/weibocom/nginx-upsync-module.git
# ./configure --add-module=/path/nginx-upsync-module
```

Consul Consul Go
Zookeeper
key/value API key/value Consul
upstream

Consul 1.9 Go yum

```
    # yum install golang -y
```

Consul Linux /bin

```
    #   wget   -S   https://releases.hashicorp.com/consul/1.1.0/consul_1.1.0_
linux_amd64.zip
    # unzip consul_1.1.0_linux_amd64.zip
    # cp consul /usr/local/bin/
```

Consul

```
    # consul agent -dev -client 10.19.48.161
```

Consul http:// IP:8500/ui/

11.3.2 Consul 的键值操作

Consul API

1

```
    curl   -X   PUT   -d   '{"weight":1,   "max_fails":2,   "fail_timeout":10}'
http://$consul_ip:$port/v1/kv/$dir1/$upstream_name/$backend_ip:$backend_port
```

```
    #curl -X PUT http://127.0.0.1:8500/v1/kv/upstreams/test_ser/127.0.0.1:82
-d '{"weight":1, "max_fails":20, "fail_timeout":20}'
```

2

```
    curl   -X   PUT   -d   '{"weight":1,   "max_fails":2,   "fail_timeout":10}'
http://$consul_ip:$port/v1/kv/$dir1/$upstream_name/$backend_ip:$backend_port
```

```
    # curl -X DELETE   http://127.0.0.1:8500/v1/kv/upstreams/test_ser/127.0.
0.1:82
```

3

```
    curl   -X   PUT   -d   '{"weight":2,   "max_fails":2,   "fail_timeout":10}'
http://$consul_ip:$port/v1/kv/$dir1/$upstream_name/$backend_ip:$backend_port
```

```
    #curl -X PUT   http://127.0.0.1:8500/v1/kv/upstreams/test_ser/127.0.0.1:
82 -d '{"weight":3 "max_fails":10, "fail_timeout":11}'
```

11.3.3　动态管理 upstream

upstream　　Consul　　Nginx

```
    http {
    upstream test_ser {
            #   Consul      test_ser             Nginx
            upsync  127.0.0.1:8500/v1/kv/upstreams/test_ser/  upsync_timeout=
3m upsync_interval=5s upsync_type=consul strong_dependency=off;

            #        upstream
            upsync_dump_path/usr/local/nginx/conf/servers/servers_test.conf;

            #           upstream           upsync_dump_path
            include /usr/local/nginx/conf/servers/servers_test.conf;

            #          least_conn   hash           upsync_lb
            least_conn
            upsync_lb least_conn
        }
        server {
            listen 80;
            location / {
                proxy_pass http://test_ser;
            }

            location = /upstream_show {
               #        allow  deny
               #
                upstream_show;
            }
        }
    }
```

指令：upsync

upstream

Consul　　Consul　　key　　upstream

127.0.0.1:8500/v1/kv/upstreams/test_ser/　　Consul　　IP　　+

127.0.0.1:8500 Consul key /v1/kv/nginx_upstream/test_ser/

upsync 11-2

表 11-2 关于 upsync 其他参数的说明

参数名	作 用
upsync_interval	Consul 5s
upsync_timeout	Consul 6ms
upsync_type	Consul Etcd
strong_dependency	on Nginx Nginx Consul Etcd off Nginx upsync_dump_path

注意： strong_dependency off Nginx

Consul 1 on

upsync_dump_path

指令：upsync_dump_path

upsync_dump_path $path

/tmp/servers_$host.conf

upstream

include /usr/local/nginx/conf/servers/servers_test.conf; strong_dependency off Nginx

指令：upsync_lb

upsync_lb $load_balance

round_robin/ip_hash/hash modula

upstream

upstream least_conn hash upsync_lb

指令：upstream_show

upstream_show

location

11.3.4　验证动态配置功能

Consul　　Consul　upstream

test_ser

```
# vim  /usr/local/nginx/conf/servers/servers_test.conf

server 1.0.0.1:1222 weight=10 max_fails=10 fail_timeout=10s;
```

Nginx　　test_ser　upstream

```
# curl http://127.0.0.1/upstream_show

Upstream name: test_ser; Backend server count: 1
        server 127.0.0.1:81 weight=10 max_fails=10 fail_timeout=10s;
```

```
# curl -X PUT http://127.0.0.1:8500/v1/kv/upstreams/test_ser/127.0.0.1:
82 -d '{"weight":1, "max_fails":20, "fail_timeout":20}'
```

upstream　　test_ser　　upstream

upsync_interval

11.3.5　高可用、高并发设计

Consul　　Nginx　　upstream

worker　　Nginx　　Consul

Consul　　Consul

- server　　client　　client　　Nginx
- 127.0.0.1:8500　　Consul
- Consul　　Consul
- Consul “　　”　　upstream

Consul

ngx_http_dyups_module nginx-upsync-module C
Ngx_Lua upstream
balancer_by_lua* Ngx_Lua upstream Nginx
Ngx_Lua upstream OpenResty

https://github.com/openresty/lua-resty-core/blob/master/lib/ ngx/balancer.md

```
http {
    upstream backend {
        server 0.0.0.1;   # just an invalid address as a place holder
        balancer_by_lua_block {
            local balancer = require "ngx.balancer"

            -- well, usually we calculate the peer's host and port
            -- according to some balancing policies instead of using
            -- hard-coded values like below
            local host = "127.0.0.2"
            local port = 8080

            local ok, err = balancer.set_current_peer(host, port)
            if not ok then
                ngx.log(ngx.ERR,"failed to set the current peer: ", err)
                return ngx.exit(500)
            end
        }

        keepalive 10;  # connection pool
    }

    server {
        # this is the real entry point
        listen 80;

        location / {
            # make use of the upstream named "backend" defined above:
            proxy_pass http://backend/fake;
        }
    }
```

```
        server {
            # this server is just for mocking up a backend peer here···
            listen 127.0.0.2:8080;

            location = /fake {
                echo "this is the fake backend peer···";
            }
        }
    }
```

Ngx_Lua

upstream

```
    local host = "127.0.0.2"
    local port = 8080
```

OpenResty

https://github.com/openresty/lua-resty-balancer

Ngx_Lua　　upstream　　upstream

nginx-upsync-module　ngx_http_dyups_module

3　　upstream

12

Nginx

1

2

3

4

5 IP User_Agent

6

Nginx

Nginx upstream

1 URI p90 p99

URI p99 99%

99%

2　　URI　　URI

URI　　800　　https://shop.zhe800.com/products/ze171126205509136896　　https://shop.zhe800.com/products/ze170814104348738286　　URI

https://shop.zhe800.com/products/[a-z0-9]+

URI　　URI

URI

3　　URI　　Web

URI　　CDN

Cache-Control

4　　Nginx

5

6

Nginx

ngxtop　　Python　　Nginx

ngxtop　　ngxtop　　Python　　Python　　pip　　ngxtop　　python2　　python3

```
# yum install python-pip
# pip install ngxtop
```

ngxtop　　Nginx　　ngxtop

ngxtop

```
log_format  main  '$remote_addr - $remote_user [$time_local]"$request" '
                  '$status $body_bytes_sent "$http_referer" '
                  '"$http_user_agent" "$http_x_forwarded_for"';
```

```
#  ngxtop --config  /usr/local/nginx/conf/nginx.conf -n  10
```

ngxtop access_log -n URI 10 ngxtop 12-1

```
running for 20 seconds, 1230 records processed: 61.19 req/sec

Summary:
|   count |   avg_bytes_sent |   2xx |   3xx |   4xx |   5xx |
|---------+------------------+-------+-------+-------+-------|
|    1230 |         2945.304 |  1175 |    46 |     9 |     0 |

Detailed:
| request_path                    |   count |   avg_bytes_sent |   2xx |   3xx |   4xx |   5xx |
|---------------------------------+---------+------------------+-------+-------+-------+-------|
| /app/cart/item/count            |      90 |           44.389 |    90 |     0 |     0 |     0 |
| /cn/f/status                |      70 |          558.414 |    70 |     0 |     0 |     0 |
| /check.json                     |      69 |           13.000 |    69 |     0 |     0 |     0 |
| /native/jump                    |      48 |           38.000 |    48 |     0 |     0 |     0 |
| /dx/pricebanner |      45 |          115.156 |    43 |     2 |     0 |     0 |
| /dx/promotion   |      43 |          176.791 |    40 |     3 |     0 |     0 |
| /dx/status      |  38 |          351.684 |    38 |     0 |     0 |     0 |
| /dx/comment     |      35 |          723.943 |    34 |     1 |     0 |     0 |
| /dx/shop        |      33 |         1044.606 |    33 |     0 |     0 |     0 |
| /dx/graph       |      32 |          903.500 |    31 |     1 |     0 |     0 |
```

12-1 ngxtop

12-1 URI HTTP

ngxtop

- “ ngxtop -l“Nginx ”; "
- IP “ ngxtop --config/usr/local/nginx/ conf/nginx. conf top remote_addr ”
- HTTP 502 “ ngxtop -l /data1/access.log --filter 'status == 502' ”

ngxtop

ngxtop

-
-
- Nginx　　Nginx

-
-
- 　　Nginx
-

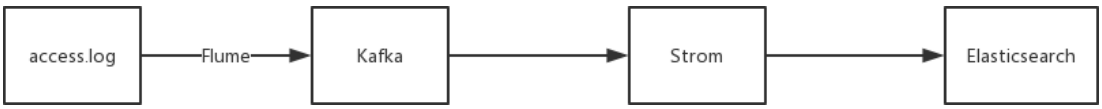
　　GoAccess

ngxtop

　　ngxtop

　　Flume　　Elasticsearch

　　Flume　　12-2



　　12-2　Flume

- Flume　　Kafka
- Kafka　　Storm　　Elasticsearch
- Elasticsearch

- 　　Flume
- Kafka
- Storm

- Elasticsearch SQL

-
- URI Storm
  Java
- Elasticsearch SQL
  SQL
- Flume Nginx

Flume

—— nginx_log_
analysis

12.4.1 架构重构

Nginx

Nginx

Ngx_Lua

- Nginx Nginx Ngx_Lua
- Ngx_Lua Nginx URI
- Ngx_Lua log_by_lua* log
- p90

-

GitHub　　https://github.com/leehomewl/nginx_ log_analysis

注意：

Ngx_Lua

12.4.2　日志远程传输

lua-resty-logger-socket　　I/O

Nginx　　log

lua-resty-logger-socket

12.5

12.4.3　时序数据库

Nginx　　PV　Page View

Nagios

Zabbix

InfluxDB　　Go

InfluxDB　　12.6

12.4.4　日志规则设计

Nginx　　$upstream_response_time

proxy_next_upstream

InfluxDB

Nginx

Nginx　URI　　$uri　　URL　　$uri

URI

Nginx

URI　　Ngx_Lua　　URI

- URI
- URI URI URI
- URI MySQL
- Ngx_Lua MySQL URI
- Nginx Ngx_Lua log URI URI URI

URI

URI Ngx_Lua URI
URI URI
URI POST
13 URI

URI MySQL
/a/b/[0-9]+ URI Nginx
Nginx URI /a/b/123 /a/b/345 Ngx_Lua /a/b/[0-9]+ /a/b/[0-9]+
InfluxDB

注意： MySQL URI
URI MySQL
MySQL 13

lua-resty-logger-socket log_by_lua*
Ngx_Lua
CPU I/O Nginx
16

12.5.1 安装 lua-resty-logger-socket

lua-resty-logger-socket syslog-ng syslog-ng
lua-resty-logger-socket lua-nginx-module lua-resty-logger-socket lua_package_path

```
# git clone https://github.com/cloudflare/lua-resty-logger-socket.git
# cp -pR lua-resty-logger-socket/lib/resty/* \
/usr/local/nginx_1.12.2/conf/lua_modules/resty/
```

12.5.2　远程传输配置

```
lua_package_path "/usr/local/nginx_1.12.2/conf/lua_modules/?.lua;;";

server {
    location / {
        log_by_lua_block {
            local ngx = require "ngx"
            local uri = ngx.var.uri
            local host = ngx.var.host
            --        UDP                  table
            local db = {
                host = '127.0.0.1',
                port = 8911,
                sock_type = udp
                flush_limit = 8096
                drop_limit = 2097152
            }
            local logger = require "resty.logger.socket"
            if not logger.initted() then
                --
                local ok, err = logger.init(db)
                if not ok then
                    ngx.log(ngx.ERR,"failed to initialize the logger: ",
                            err)
                    return
                end
            end
            local msg = host .. uri
            --   host   uri
            local bytes, err = logger.log(msg)
            if err then
                ngx.log(ngx.ERR, "failed to log message: ", err)
                return
            end
        }
```

```
            proxy_pass http://servers;
        }
    }
```

UDP　User Datagram Protocol

12.5.3　参数解读

lua-resty-logger-socket

1．初始化配置

ok, err = logger.init(user_config)

logger.init　　user_config

table　　12-1

表 12-1　user_config 支持的初始化参数

参　数	作　用
flush_limit	+　　≥ flush_limit 4096　　4KB
drop_limit	+　　≥ drop_limit drop_limit　　1048576　　1MB
timeout	
host	IP
port	
sock_type	TCP　UDP　　TCP　　sock_type = ‘tcp’　　TCP
path	UNIX　　path
max_retry_times	
retry_interval	retry_interval=100　　0.1s
pool_size	pool_size =10
max_buffer_reuse	
periodic_flush	nil
ssl	SSL　Secure Sockets Layer　　ssl=false
ssl_verify	true
sni_host	SNI　Server Name Indication

2．检查初始化配置

initted = logger.initted()

false

3．日志传输

bytes, err = logger.log(msg)

flush_limit　　periodic_flush

msg　　bytes

err

4．立即传输日志

bytes, err = logger.flush()

flush_limit

InfluxDB　　InfluxDB

InfluxDB　　InfluxDB

https://docs.influxdata.com/influxdb/v1.6/

12.6.1　安装 InfluxDB

CentOS　　InfluxDB

```
   #  wget  https://dl.influxdata.com/influxdb/releases/influxdb-1.5.4.x86_64.
rpm
   # sudo yum localinstall influxdb-1.5.4.x86_64.rpm
```

InfluxDB

```
   # /etc/init.d/influxdb start
```

https://portal. influxdata.com/downloads

12.6.2　基本概念和操作

InfluxDB　　InfluxDB

12-2

表 12-2　关于 InfluxDB 关键字的说明

关键字	说　明	示　例
database		create database nginx;
measurement	table	show measurements
point	1	1 insert nginx_log,host=testnginx.com uri= "/abc"

InfluxDB insert 1 point time tags fields

time nanosecond

tags group by order by

fields

注意：time tags time tags

12.6.3　数据分析之查询函数

InfluxDB 12-3

表 12-3　监控分析中常见的查询函数及其作用

函　数	作　用	在 Nginx 日志分析中的作用
count	field	URI PV
top	field top N N	URI
percentile	field	URI p90 p99
mean	field	URI
derivative	field	

Nginx InfluxDB URI

```
    >    select  top(upstream_time,3)  from  nginx  where  mysql_host_uri=
'"www.zhe800. com/"' and time > (now()-1m)  tz('Asia/Chongqing');
```

```
name: nginx
time                                top
----                                ---
2018-07-20T16:00:51.425490336+08:00 0.111
2018-07-20T16:01:02.480582679+08:00 0.102
2018-07-20T16:01:04.428871068+08:00 0.116
```

InfluxDB

https:// github.com/leehomewl/nginx_log_analysis

12.6.4　数据存放之保留策略

Nginx　　InfluxDB

InfluxDB

InfluxDB

InfluxDB　　InfluxDB　　Nginx

1h　　1h

Nginx

```
CREATE DATABASE nginx
CREATE RETENTION POLICY "1_hours" ON "nginx" DURATION 1h REPLICATION 1
DEFAULT
```

Nginx　　1　　1

```
CREATE RETENTION POLICY "1_month" ON "nginx" DURATION 30d REPLICATION  1
```

```
> SHOW RETENTION POLICIES ON nginx
name    duration shardGroupDuration replicaN default
----    -------- ------------------ -------- -------
autogen 0s       168h0m0s           1        false
1_hours 1h0m0s   1h0m0s             1        true
1_month 720h0m0s 24h0m0s            1        false
```

12.6.5　定时任务之连续查询

InfluxDB

URI p90

1 tp90_nginx

```
   >CREATE CONTINUOUS QUERY tp90 ON nginx BEGIN SELECT percentile
(upstream_time, 90) AS tptime, mysql_host_uri INTO nginx."1_month".
tp90_nginx FROM nginx."1_hours".nginx GROUP BY time(1m), mysql_host_uri END
```

```
   >show continuous queries
   name: _internal
   name query
   ---- -----

   name: nginxxx
   name query
   ---- -----

   name: mydb
   name query
   ---- -----

   name: nginx
   name  query
   ----  -----
   tp90   CREATE CONTINUOUS QUERY tp90 ON nginx BEGIN SELECT percentile
(upstream_time, 90) AS tptime, mysql_host_uri INTO nginx."2_month".tp90_
nginx FROM nginx."1_hours".nginx GROUP BY time(1m), mysql_host_uri END
```

12.6.6 客户端操作之 API

InfluxDB API Ngx_Lua API InfluxDB

```
   # curl -GET 'http://localhost:8086/query?pretty=true' --data-urlencode
"db=nginx"  --data-urlencode "q=select uri from nginx limit 1"
   {
       "results": [
           {
               "statement_id": 0,
               "series": [
                   {
                       "name": "nginx",
                       "columns": [
                           "time",
```

```
                            "uri"
                        ],
                        "values": [
                            [
                                "2018-07-20T07:00:00.10149097Z",
                                "/gateway/app/detail/graph"
                            ]
                        ]
                    }
                ]
            }
        ]
    }
```

12.6.7　使用 UDP 模式传输数据

InfluxDB　　TCP　UDP　　UDP　　InfluxDB UDP　　/etc/influxdb/influxdb.conf

　　UDP　　Nginx

```
    [[udp]]
      enabled = true
      bind-address = ":8911"
      database = "nginx"
      retention-policy = ""

     # These next lines control how batching works. You should have this
enabled
     # otherwise you could get dropped metrics or poor performance.
Batching
     # will buffer points in memory if you have many coming in.

     # Flush if this many points get buffered
      batch-size = 5000

     # Number of batches that may be pending in memory
      batch-pending = 10

     # Will flush at least this often even if we haven't hit buffer limit
      batch-timeout = "1s"

     # UDP Read buffer size, 0 means OS default. UDP listener will fail if
set above OS max.
      read-buffer = 0
```

12.6.6 InfluxDB API SQL Ngx_Lua API lua-resty-http

注意：Ngx_Lua HTTP “ ” ngx.timer.at API

12.7.1 安装 lua-resty-http

lib lua_package_path

```
# git clone https://github.com/pintsized/lua-resty-http.git
# cp lua-resty-http/lib/resty/* /usr/local/nginx/conf/lua_modules/resty/
```

12.7.2 使用方式

SQL select uri from nginx limit 1 lua-resty-http HTTP

```
location =  /q {
     content_by_lua_block {
          --
          local http = require "resty.http"
          local ngx = require "ngx"
          --     SQL
          local post_data = "db=nginx&q=select uri from nginx limit 1"
          --
          local hc = http:new()
          --
          local res, err=hc:request_uri('http://127.0.0.1:8086/query', {
              method = "POST", -- POST or GET
              headers  =  {["Content-Type"]  =  "application/x-www-form-
urlencoded" },
              body = post_data
          })
          if res.status ~= 200 then
             ngx.log(ngx.ERR,"SQL failed   " .. err)
             return
          end
          --
          ngx.say(res.body)
     }
}
```

```
   # curl  http://www.zhe800.com/q
   {"results":[{"statement_id":0,"series":[{"name":"nginx","columns":["time
","uri"],"values":[["2018-07-
20T08:00:00.101242877Z","/operation/abtest/pageconfig/v1"]]}]}]}
```

Ngx_Lua

InfluxDB InfluxDB

SQL

- 4
-
- InfluxDB
- UDP read-buffer

Nginx

https://github.com/leehomewl/nginx_log_analysis

13

-
-
-
-
- Bug Bug Bug
-
-

……

Nginx

注意：

80%

80%

3

1　　CDN

key　　Host+URL

proxy_cache　varnish　squid

5xx

200　304

2

CDN

Redis　Memcached　Couchbase

3

2

方案 1：提供两套数据，使用主备模式。

13-1

13-1

1

- “ ”
-
-

1

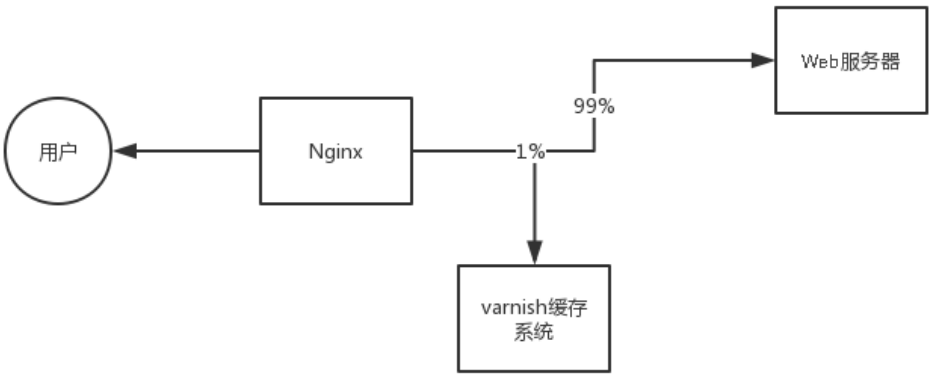
方案 2：将一部分客户端请求随机保存到缓存系统中，让缓存区拥有热数据，当出现异常时将请求切换到缓存系统。缓存系统没有业务逻辑，只和硬件资源有关，因此稳定性极高。

Cookie

Nginx error_page 5xx 4xx

Nginx

13-2



13-2

2

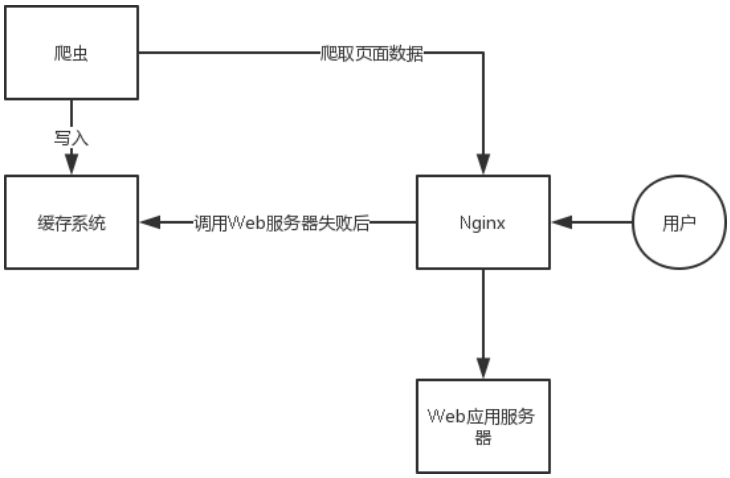
-
-

方案 3：自建爬虫模拟用户访问，将数据爬取后存放到缓存系统中，当出现异常时，将请求转发到缓存系统。

Nginx　error_page　5xx　4xx

Nginx

13-3



13-3

3

-
- JSON
- App　App
-

3.1　3

1　URL　——　MySQL

Ngx_Lua MySQL Ngx_Lua

2 12 nginx_log_analysis URL
URL
Ngx_Lua URL
URL p90

3
App
URL

4

5 Nginx
URL

6 MySQL URL
URL
Ngx_Lua URL

URL

HTTP

7

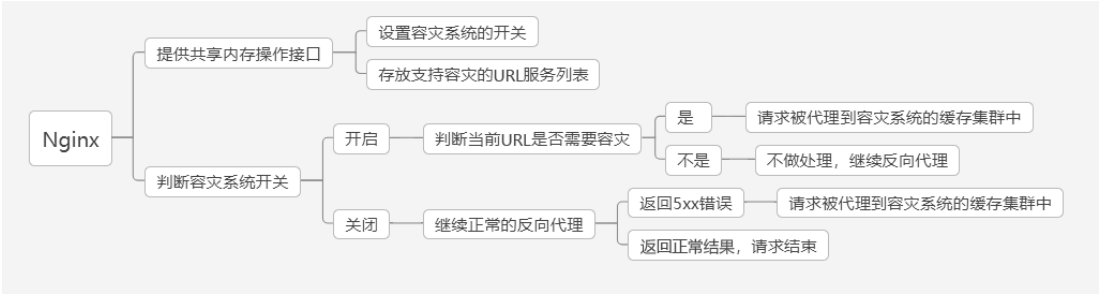
“ ”

13-4 URL

13-4 URL

13.3.1　反向代理系统

Ngx_Lua　　Ngx_Lua　　API　　13-5



13-5

13.3.2　日志分析系统

URI　　URI　　13-6

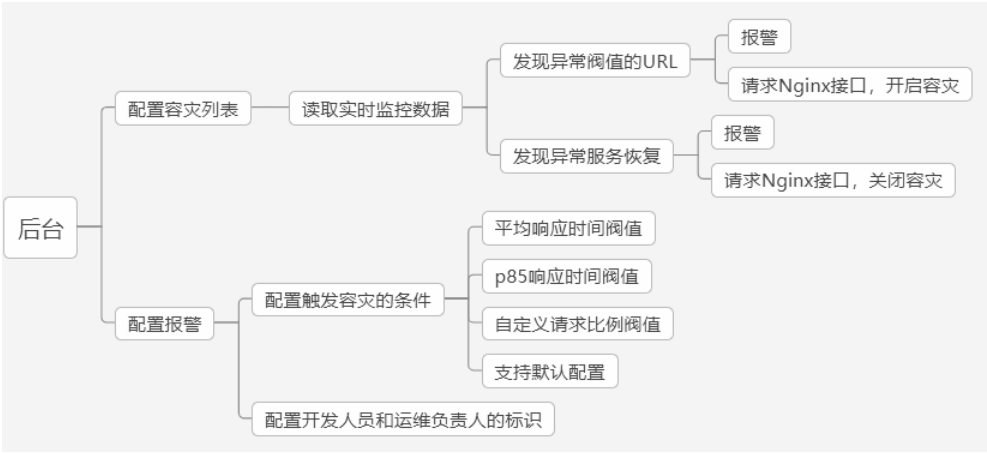
13-6　URI

13.3.3　后台系统

URL

URL　　URL

13-7



13-7

13.3.4 爬虫系统

URL URL

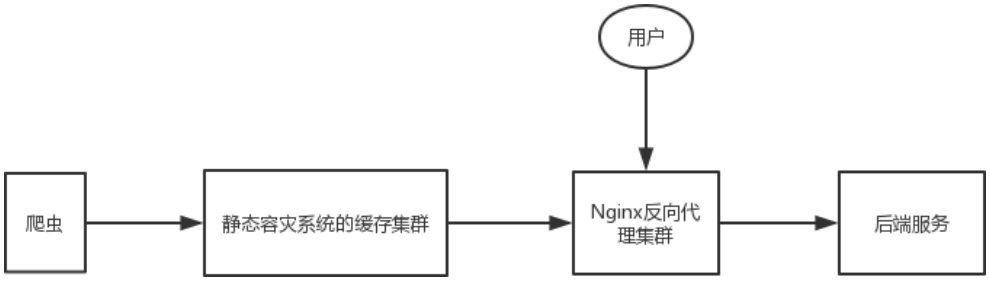
13-8

13-8

13.3.5 容灾的缓存系统

key Host+URL+

20min 20min 1 1h 3
1 72 13-9



13-9

13-9

13.3.6　时间版本的用途

200

URL

Ngx_Lua

13.3.7　异地容灾

“　　”

13-10

13-10

Ngx_Lua

nginx_log_analysis Wiki

13.4.1 Ngx_Lua 应用

Nginx Ngx_Lua Lua API

crash_status=0 0 URL

Nginx 1

URL URL

URL

20min

```
local ngx = require "ngx"
local ab_ver = tonumber(os.date('%H', ngx.time())) * 3
ab_ver = ab_ver + math.floor(tonumber(os.date('%M', ngx.time())) / 20)
```

URL　　URL　　URL　　URL 3
Ngx_Lua　　3　　3　URL
Ngx_Lua　　URL　　URL
URL　　key/value
URL　　URL　　CPU
CPU

URL
URL　　key/value
QPS　　8%
URL

URL　　URL
URL　　Lrucache　　Lrucache

- Key
- Value　　table　　URL　　{URL1,URL2,URL3}

for　　URL
URL

```
-- host           Host    ngx.var.host
-- uri           URI    ngx.var.uri
-- url_list                table       host            url_list
local find_uri = function(host,uri,url_list)
    local res_uri
    for key, value_uri in pairs(url_list) do
    --                          jo
        local m, err  = ngx.re.find(uri ,  value_uri[1]  ,"jo")
        if m then
    -- res_uri           URI
            res_uri =  host .. value_uri[1]
        end
    end
    return res_uri
end
```

URL　　location　　Nginx　　URL

URL $ngx_uri location
URL Ngx_Lua

```
    location ~ ^/a/[0-9]+$ {
#       $ngx_ur         URL               Ngx_Lua
#    $find_uri          MySQL         URL
        set $ngx_uri '/a/[0-9]+$';
        proxy_pass http://ups;
    }
```

Nginx

13.4.2 爬虫和日志系统的关系

12 nginx_log_analysis InfluxDB Nginx
Nginx

20min

- App
  URL
  20min
- URL 1200s
- 20min 30min 1h
  20min

13.4.3 全部容灾和部分容灾功能

- URI

- 35%　5xx　Nginx
  p=35　35%
- Lua　1~100　35
  35　lua-resty-random
-

3

3

13.5.1　从日志分析系统中复制请求

InfluxDB　URL
InfluxDB　Nginx
GET

QPS

13.5.2　利用 goreplay 复制流量

goreplay　goreplay　Go
goreplay
goreplay
goreplay

```
# wget \
https://github.com/buger/goreplay/releases/download/v0.16.1/gor_0.16.1_x
```

```
64.tar.gz
   # tar -zxvf gor_0.16.1_x64.tar.gz
   # cp  goreplay /usr/bin
```

GET IP 192.168.1.15

Host

```
   #  goreplay  --input-raw  :80  --output-http="http://192.168.1.15"  -http-
allow-method GET  --http-original-host
```

13.5.3 Nginx 的镜像功能

goreplay Nginx Nginx 1.13.4 ngx_http_mirror_module OpenResty 1.13.6.2 OpenResty 1.13.6.2

```
   location / {
       #
       mirror /crash_mirror;
       #
       mirror_request_body off;
       #
       proxy_cache crash_cache_store;
   }

   location /crash_mirror {
       internal;

       #
       if ($http_crw_rd = 'spider_static_crash')  {
          return 411 ;
       }
       proxy_pass_request_body off;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

       #               URL                Nginx
       proxy_pass http://nginx_servers$request_uri;
```

```
    #                Nginx
    proxy_set_header C-Mirror 1;
}
```

13.5.4　灰度验证容灾系统缓存

- Nginx upstream
  upstream
- Nginx URI
- DNS Nginx
- PC App DNS DNS
  Nginx
-
  URI

Ngx_Lua

https://github.com/leehomewl/nginx_log_analysis

14

Nginx

Nginx

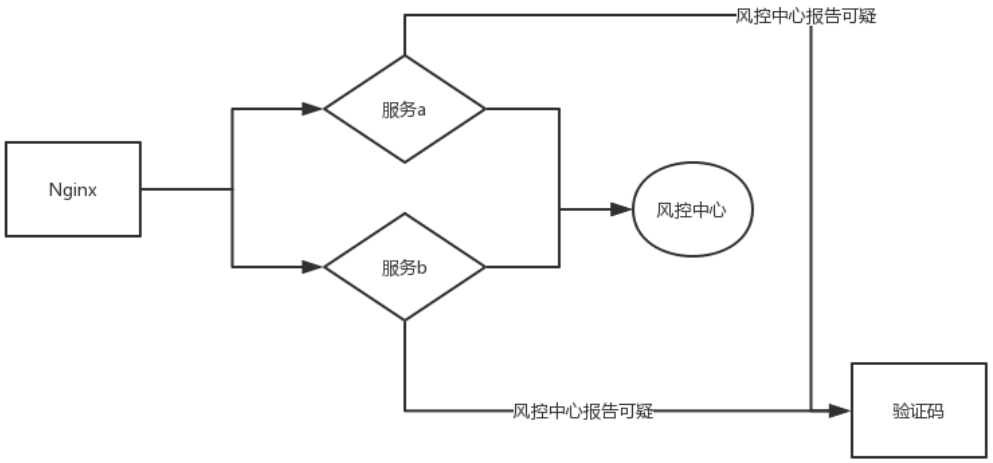
注意：

14-1 a b

Nginx Nginx

Nginx

- HTML
-
- JS



14-1

HTML

URL HTML

nginx_log_analysis URL

Content-Type Content-Type “ text/html ” “ text/css ”

URL JSON HTML JPG URL HTML

nginx_log_analysis MySQL URL Content-

Type

注意： Content-Type

Nginx

- Ngx_Lua A IP User_Agent Cookie
- Ngx_Lua B Content-Type HTML URI
- IP Ngx_Lua API A

IP

IP

- A Ngx_Lua B URI

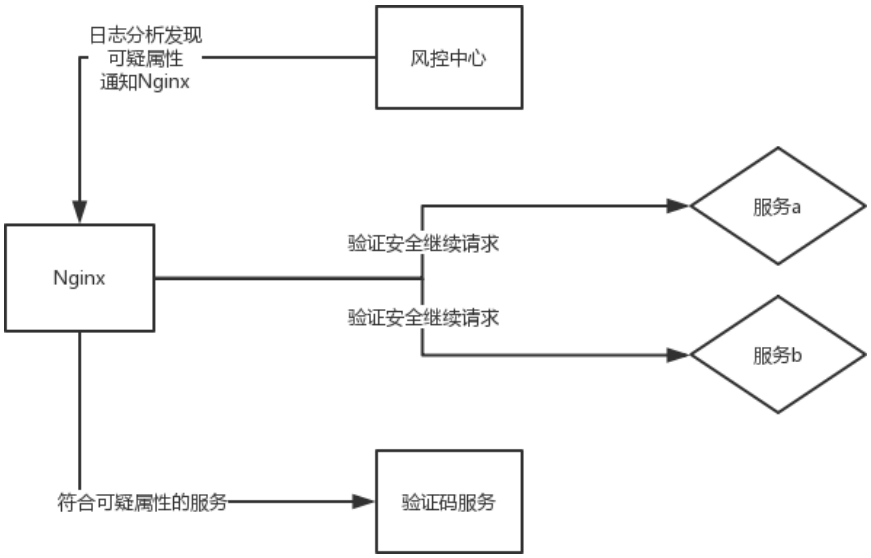
HTML

HTTP 421 HTTP

-

IP IP

Nginx 14-2



14-2 Nginx

Ngx_Lua

14.1

Nginx

14.2.1 利用 auth_request 管理鉴权

Nginx 1.5.4 ngx_http_auth_request_module

2xx

401　403

```
    #                          URI
        proxy_pass http://127.0.0.1:82/$request_uri;
        proxy_pass_request_body off;
        proxy_set_header Content-Length "";
        proxy_set_header X-Original-URI $request_uri;
    }
```

200 URI location
URI Nginx “ ”
URI

- access_by_lua_block location server http

注意： http http Nginx server

CSS JS JPG CPU Nginx
URI http

- URI 13
URI Ngx_Lua Nginx
URI
- URI
-
URI URI

7.13 Ngx_Lua ngx.location.capture
ngx.location.capture_multi
Ngx_Lua

14.3.1　轻线程的启动和终止

Ngx_Lua　　light thread　　Ngx_Lua

指令：ngx.thread.spawn

co = ngx.thread.spawn(func, arg1, arg2, …)

rewrite_by_lua*　access_by_lua*　content_by_lua*　ngx.timer.*　ssl_certificate_by_lua*　ssl_session_fetch_by_lua*

Lua　　func　　arg1　arg2　　func

ngx.exit　ngx.exec　ngx.redirect　ngx.req.set_uri(uri, true)

```
location / {
    content_by_lua_block {
        local ngx = require "ngx";
        local function func_say(a)
             --     a           worker
             ngx.sleep(a)
             ngx.say(a)
        end
        ngx.thread.spawn(func_say,'1')
        ngx.thread.spawn(func_say,'4')
        ngx.thread.spawn(func_say,'3')
        ngx.thread.spawn(func_say,'2')
     }
}
```

```
[root@testnginx ~]# curl http://172.19.3.41/
1
2
3
4
```

“1　4　3　2”

-
-

```
location / {
    rewrite_by_lua_block {
        local ngx = require "ngx";
        local function func_say(a)
             ngx.sleep(a)
             --     a    3
             if a == '3' then
                ngx.exit(0)
             end
             ngx.say(a)
        end
        ngx.thread.spawn(func_say,'1')
        ngx.thread.spawn(func_say,'4')
        ngx.thread.spawn(func_say,'3')
        ngx.thread.spawn(func_say,'2')
    }
}
```

```
[root@testnginx ~]# curl http://testnginx.com/
1
2
```

“ 1 ” “ 2 ” a==3 ngx.exit(0)

a==4

14.3.2 等待和终止轻线程

ngx.thread.spawn Ngx_Lua

指令：ngx.thread.wait

ok, res1, res2, ⋯ = ngx.thread.wait(thread1, thread2, ⋯)

rewrite_by_lua* access_by_lua* content_by_lua* ngx.timer.* ssl_certificate_by_lua* ssl_session_fetch_by_lua*

thread1 thread2 ngx.thread.wait

```
location / {
    rewrite_by_lua_block {
        local ngx = require "ngx";
        local function func_say(a)
            ngx.sleep(a)
            return a
        end

        local threads = {
           ngx.thread.spawn(func_say,'1'),
           ngx.thread.spawn(func_say,'4'),
           ngx.thread.spawn(func_say,'3'),
           ngx.thread.spawn(func_say,'2')
        }
        --
        for i = 1, #threads do
            local ok, res = ngx.thread.wait(threads[i])
            if not ok then
                ngx.say(i, ": failed to run: ", res)
            else
                ngx.say(i, " res: ",res)
            end
        end

    }
}
```

```
[root@testnginx ~]# curl http://testnginx.com/
1 res: 1
2 res: 4
3 res: 3
4 res: 2
```

指令：ngx.thread.kill

ok, err = ngx.thread.kill(thread)

rewrite_by_lua* access_by_lua* content_by_lua* ngx.timer.*

ngx.thread.spawn true

Nginx

ngx.location.capture

```
location / {
    rewrite_by_lua_block {
        local ngx = require "ngx";
        local function func_say(a)
            ngx.sleep(a)
            ngx.say(a)
        end

        local a = ngx.thread.spawn(func_say,'1')
        local b = ngx.thread.spawn(func_say,'4')
        local c = ngx.thread.spawn(func_say,'3')
        local d = ngx.thread.spawn(func_say,'2')
        --     b
        if coroutine.status(b) == 'running' then
           ngx.say(coroutine.status(b))
           local ok, err = ngx.thread.kill(b)
        --     b             b
           ngx.say('kill   b       b               ',coroutine.status(b))
        end
    }
}
```

```
[root@testnginx ~]# curl http://testnginx.com/
running
kill   b       b               dead
1
2
3
```

coroutine.status Lua

3 dead suspend running

14.3.3 URL 的外部合并和内部并发

HTTP 1.1 Web

URL

URL URL
URL URL

- TCP
-
- URL

URL
Nginx

- URL
- Nginx
- Nginx Nginx

```
location / {
    content_by_lua_block {
        local ngx = require "ngx";
        --     lua-resty-memcached          memcahed
        local memcached = require "resty.memcached"
        --     lua-resty-http              HTTP
        local http = require "resty.http"
        local httpc = http.new()

        --     memcahed
        local function op_memcached(key,timeout)
            local memc = memcached:new()
             memc:set_timeout(timeout)
            memc:connect("127.0.0.1", 11211)
            local res, err = memc:get(key)
            local ok, err = memc:set_keepalive(10000, 10)
            return res
        end
        local function func_say(a)
            ngx.sleep(a)
            return a
        end

        -- Ngx_Lua
        local function capture_req_http()
            local res = ngx.location.capture("/test_ngx_lua")
```

```
            return res.body
        end
        --        HTTP
        local function cosocket_req_http()
            local httpc = http.new()
            httpc:set_timeout(timeout)
            local  res,  err  =  httpc:request_uri("http://127.0.0.1:8011/
share_set", {
                method = "GET",
                body = "a=1&b=2",
                keepalive_timeout = 60,
                keepalive_pool = 10
            })
            return res.body
        end

        --     URL
        local memc_get = ngx.var.arg_mc_get
        local la_say = ngx.var.arg_la_say
        local lb_get = ngx.var.arg_lb_get
        local lc_get = ngx.var.arg_lc_get
        --        table
        local threads = {}

        --
        if memc_get then
            --     op_memcached   memcached   timeout       “1”
            --
            --
            table.insert(threads,ngx.thread.spawn(op_memcached,memc_get,1))
        end
        if la_say then
            table.insert(threads,ngx.thread.spawn(func_say,1))
        end
        if lb_get then
            table.insert(threads,ngx.thread.spawn(cosocket_req_http,1))
        end
        if lc_get then
            table.insert(threads,ngx.thread.spawn(capture_req_http))
        end
        --
        for i = 1, #threads do
            local ok, res = ngx.thread.wait(threads[i])
```

```
                if not ok then
                    ngx.say(i, ": failed to run: ", res)
                else
                    ngx.say(i, " res: ",res)
                end
            end

        }
    }
```

```
    #   curl 'http://testnginx.com/?mc_get=testnginxlua&la_say=1&lb_get=1&lc_
get=1'
    1 res: bar
    2 res: 1
    3 res: share_set!!
    4 res: test_ngx_lua!!!
```

URL　　Ngx_Lua

14.3.4　使用 cosocket 实现外部访问

lua-resty-http　　cosocket　API　　HTTP

Ngx_Lua　　lua-resty-http　　HTTP

lua-resty-http

Nginx

Nginx

0Y 15 0

'Y;˜

' ÔK¼ 9 .*ó ã ÈF (<? L’ !,X Ä W À Ý,X 97¾ Ø û-¹ á,X ð2ö é ¢ È Ý,X 97¾ á á ?•4£ ô,X5%0- È 3 Ý Ã6Ñ 97¾ ¤ þ Ž Ä V pF ¹ Þ/¡/¡ È Ã6Ñ î Î)„ V ß ™ ‰ Ä

x Î)„ á u û6ÑKÂNl È ª 6*ü A“KÂEó z È*î7Ç Î)„ á u ý$W Ã5%0-+ÿ+¾,X ™ ‰ Ä

x 5%0- D B>•0W ª È @ Ì,X î u D B>• ˆM 4-0²] Í Ä

0´ Ú î Í(<?,X á à ™ ‰E¯> Ú d È J4§ Ü Nginx 9 Ÿ4¡ V) { (<?,X> Ä

g 3 I2O B û'Y;˜ ¹ £ <'Y;˜

(<? û7È Ã Ú ø2O Ö ð2ö é ¢,X(<? ` J ã(<? ÄFw V) Ú W À 6 Û Ã ¹ S*ü fL8 "© È ð2ö é ¢,X(<?FÑ ú Ý â ,X User-Agent(M U È _ V ð(«,X Sogou web spider/4.0 Sogou inst spider/4.0Ã,R z,X BaiduSpider/2.0Ä " ÛAš 3 Ã ¹ þEô È ¹ \ î J ã(<? î þEô User-AgentE¯> A“KÂ Ä V) Ú þEô(<? 6 Û# /ß V ß Ä

x .BAx ð2ö é ¢(<?,X User-AgentÈE- ü Ø û ð2ö é ¢,X5%0- ÞFÑ Ý Û"¼ È V ð(«,X User-AgentÃ ¹ ¹,ß l • Wiki È5% http://help.sogou.com/spider.htmÄl

x ¢ ¹ « 9‹ ª0ú ÜE- o(<?,X User-Agent,X IP Ä

x S*ü Linux Q ¸ nslookup9‹ ª ¡ å?· d È k !8 IP Í h,X ³ á Ä

x k ,X ³ á V p á ð2ö é ¢ @ Ì,X ³ á ÈFw E- o(<? 2 b þEô(<? Ä

ßM6 Nginx ¹ «,X ´ 5(<?A„) Ö

123.125.71.29 www.zhe800.com - [23/May/2018:18:59:07 +0800] "GET /ju_deal/doudouxiao_31986454 HTTP/1.1" 200 671 "-" "**Mozilla/5.0 (compatible;**

```
Baiduspider/2.0; +http://www.baidu.com/search/spider.html)"
   220.181.125.106  www.zhe800.com  -  [23/May/2018:18:56:20  +0800]  "GET
/concern/11923041     HTTP/1.1"     200     162     "-"     "Sogou     web
spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"
   24.125.7.34   www.zhe800.com   -   [23/May/2018:18:56:20   +0800]   "GET
/concern/11923041     HTTP/1.1"     200     162     "-"     "Sogou     web
spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"
```

IP　　nslookup　　IP　　123.125.71.29

```
   # nslookup 123.125.71.29
   Server:      192.168.1.4
   Address: 192.168.1.4#53

   Non-authoritative answer:
   29.71.125.123.in-addr.arpaname = baiduspider-123-125-71-29.crawl.baidu.com.

   Authoritative answers can be found from:
   125.123.in-addr.arpa nameserver = ns2.bta.net.cn.
   125.123.in-addr.arpa nameserver = ns.bta.net.cn.
   ns.bta.net.cn    internet address = 202.96.0.133
   ns2.bta.net.cn   internet address = 202.106.196.28
```

IP　　220.181.125.106　　nslookup

```
   # nslookup 220.181.125.106
   Server:      192.168.1.4
   Address: 192.168.1.4#53

   Non-authoritative answer:
   106.125.181.220.in-addr.arpa name       =       sogouspider-220-181-125-
106.crawl.sogou.com.

   Authoritative answers can be found from:
   181.220.in-addr.arpa nameserver = idc-ns3.bjtelecom.net.
   181.220.in-addr.arpa nameserver = idc-ns1.bjtelecom.net.
   181.220.in-addr.arpa nameserver = idc-ns2.bjtelecom.net.
   idc-ns1.bjtelecom.netinternet address = 218.30.26.68
   idc-ns2.bjtelecom.netinternet address = 218.30.26.70
   idc-ns3.bjtelecom.netinternet address = 211.100.2.125
```

IP　　24.125.7.34　　nslookup

```
   # nslookup 24.125.7.34
   Server:      192.168.1.4
   Address: 192.168.1.4#53
```

```
Non-authoritative answer:
34.7.125.24.in-addr.arpa name = c-24-125-7-34.hsd1.ga.comcast.net.

Authoritative answers can be found from:
125.24.in-addr.arpa  nameserver = dns105.comcast.net.
125.24.in-addr.arpa  nameserver = dns102.comcast.net.
```

IP IP 24.125.7.34

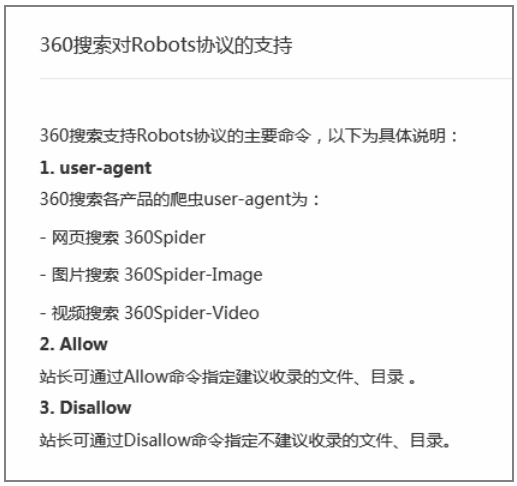
IP

15.2.1 搜索引擎的 User-Agent

User-Agent

360 User-Agent https://www.so.com/help/help_3_2.html 15-1

360 User-Agent



15-1 360 User-Agent

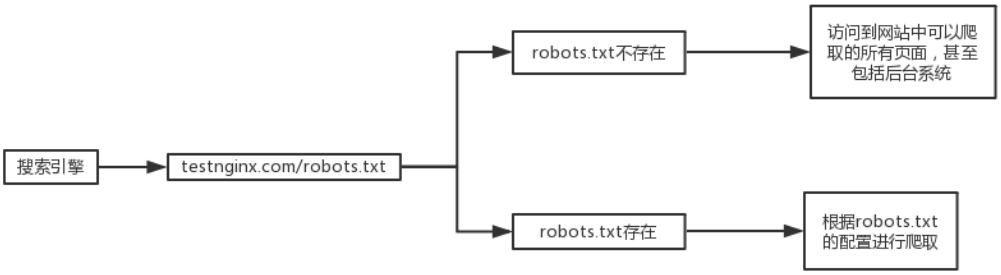
15.2.2　Robots 协议

Robots　　Robots Exclusion Protocol　　Robots

robots.txt

robots.txt　　15-2

robots.txt



15-2　robots.txt

robots.txt　　15-1

表 15-1　robots 文件的配置说明

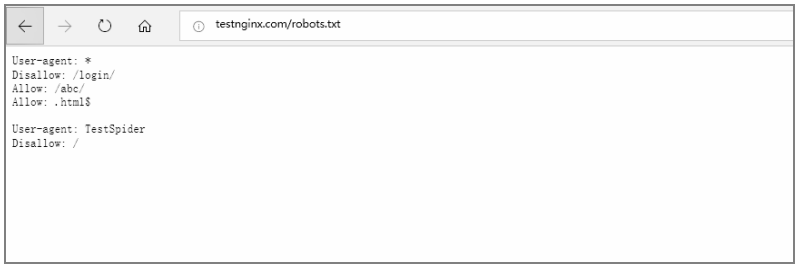
语　　法	说　　明
User-agent: *	
Disallow: /	URL
Disallow: /login/	/login
Allow: /abc/	/abc
Allow: .html$	“ .html ”　　URL

robots.txt　　15-3

15-3　robots.txt

TestSpider

15-4



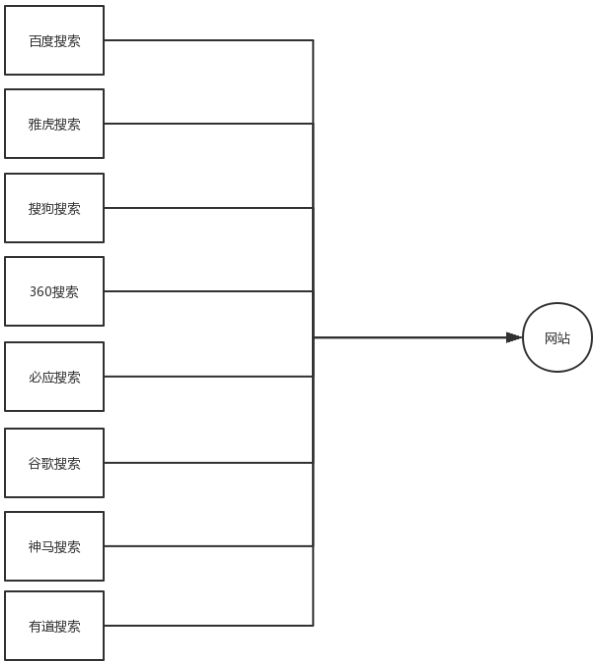
15-4 TestSpider

注意：robots.txt ǯ E ˎ ˃ ` E__ ᵠ ɓ SEO Search Engine Optimizers ˚ ˆ ᵛ ˊ ∀ . ˃̈ T` ð Ў__ ᵠ ä† Ҳ т Ί ¦ л ᴱ ≠ ˛ Ї robots.txt î

15.2.3 控制搜索引擎爬虫实战

15-5

“ ”



15-5

1　　Nginx
SEO

2　　SEO
　　robots.txt

3

4　　3

Nginx

360

```
# cat nginx.conf

user  webuser webuser;
worker_processes  1;

worker_rlimit_nofile 102400;

events {
    use epoll;
    worker_connections  102400;
}

http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;

    upstream  test_12 {
        server 127.0.0.1:81  weight=20  max_fails=300000 fail_timeout=5s;
        server 127.0.0.1:82  weight=20  max_fails=300000 fail_timeout=5s;
    }

    upstream spider_cache {
         server 127.0.0.1:8000;
    }
```

```
map $http_user_agent  $server {
    #     user_agent
    default             test_12;
    #     user_agent        360
   ~360Spider          spider_cache;
    #     user_agent
   ~BaiduSpider        spider_cache;
}
server {
    listen       80;
    server_name  testnginx.com;
    location  /  {
    # $server    map
    proxy_pass http://$server;
    }
}
```

15.3.1 发现恶意爬虫

1 User-Agent 15.1

2 User-Agent LWP.*Simple BBBike Python-urllib ^trasin ^curl *libcurl WWW.*Mechanize

3 User-Agent

IP

IP

- URL
- URL
- Cookie Cookie

- Cookie　　Cookie
- IP　　IP

4

3

IP

3

5　　3　　4　　IP

Cookie　　Cookie

Cookie

“　　” Cookie　　Cookie　　“

”

注意：Ą м App ⁵ Ь　　н ∕₀　　V Ч ̶ ̷ ̃ Ф н к`　　‴　4G　　ض ∖　　∖ Ũ

Ь　IP ą̆ ẽ　　خ ״ ص　　ʒ غ ˍ　　IP ą̆ ẽ　　ش ڭ ŧ Ч ̶ ̷　　״ †　　ᴬ URL

∖ Î Cookie ϐ　Ь　　ŧ Ч ̶ ̷ î Ш Ą　　ſ　　Ь Ф н　ʬ ‘ __ ’　ヽ　ヽ Î ң

Ь ɔ ɔ　　ق ω　　خ ∖　　ھ ذ　　ˍ　　ϖ ヽ- т ヽ　ᴬ　　¨　　ƒ　∖　　Ч ̶ ̷ î

15.3.2　抵御恶意爬虫之禁止访问

```
    server {
        listen       80;
        server_name  testnginx.com;

        deny xxx.xxx.xxx.xxx;   #                IP          deny
        deny 124.124.124.123;   #                IP          deny

        if  ($http_user_agent  ~*  LWP.*Simple|BBBike|Python-urllib|^trasin|^
curl. *libcurl|WWW. *Mechanize) {
           return 403;          #           User-Agent
        }
        location  /  {
          proxy_pass http://test_12;
        }

    }
```

15.3.3 抵御恶意爬虫之验证码拦截

IP

“ ”

IP Nginx Lua Nginx Nginx

14

IP

IP Nginx

deny 2 IP

IP

IP

“ ”

- 503

200

- IP User-Agent
- IP
- JSON

Nginx

```
server {
    listen       80;
    server_name  testnginx.com;

    # /a.json   b.json
    location ~ ^(/a.json|/b.json) $ {
    # $remote_addr       IP          IP          IP                Lua
    #             Redis        geo                              if
    #                       IP          CDN                          realip
    #           IP
```

```
        if ($remote_addr ~ (xxx\.xxx\.xxx\.xx|xxx\.xxx\.xxx\.xxx)) {
        #            /make_false/  location
            rewrite ^/(.*) /make_false/$1 last;
        }
        proxy_pass http://test_12;
    }

    location ~ ^/make_false/ {
        #        $request_uri                URL          /make_false/
       proxy_pass http://err_data_servers$request_uri;
    }
}
```

upstream　　err_data_servers

“　　”

“　　”

16

Nginx ʔ ʍ ╗ worker
à ╗ à Â Nginx ʃ
ʔ ╗ a Nginx Â
é ┘Ո Nginx Â

注意： Ngx_Lua Nginx
OpenResty LuaJIT 2.0 LuaJIT 2.1

У Ю ˆ Э Ŀ Ю
˜ ┘Ո Ŀ Э Â Debug
Ո Debug У Â

16.1.1 安装 SystemTap

SystemTap Linux é ┘Ո
ʔ SystemTap Â

Linux 3.5 æ
è utrace Φ CentOS 6.4 Э Ю'

```
# uname -r
```

```
   2.6.32
```

ǐ　　kernel-devel'

```
   #  rpm -qa |grep   kernel-devel
   kernel-devel-2.6.32-696.30.1.el6.x86_64
```

'

```
   # yum install kernel-devel
```

ʌ　　Debug　　'

```
   # wget -S http://debuginfo.centos.org/6/x86_64/kernel-debuginfo-common-
x86_64-2.6.32-696.30.1.el6.x86_64.rpm
   # wget -S http://debuginfo.centos.org/6/x86_64/kernel-debuginfo-2.6.32-
696.30.1.el6.x86_64.rpm
   # rpm -ivh kernel-debuginfo-2.6.32-696.30.1.el6.x86_64.rpm    kernel-
debuginfo-common-x86_64-2.6.32-696.30.1.el6.x86_64.rpm
```

SystemTap'

```
   # yum install systemtap
```

ǐ　　ㄇ'

```
   # stap -ve 'probe begin { log("Test Nginx Systemtap!")exit() }'
   Pass 1: parsed user script and 117 library script(s) using
213788virt/41172res/3232shr/38628data kb, in 330usr/20sys/348real ms.
   Pass 2: analyzed script: 1 probe(s), 2 function(s), 0 embed(s), 0
global(s) using 214580virt/42280res/3536shr/39420data kb, in 10usr/0sys/
8real ms.
   Pass 3: translated to C into "/tmp/stapldNGqo/stap_193cfbe6fbce06a86a95
81c649f20084_960_src.c" using 214580virt/42668res/3892shr/39420data kb, in
0usr/0sys/0real ms.
   Pass 4: compiled C into "stap_193cfbe6fbce06a86a9581c649f20084_960.ko"
in 1040usr/210sys/1281real ms.
   Pass 5: starting run.
   Test Nginx Systemtap!
   Pass 5: run completed in 0usr/10sys/345real ms.
```

16.1.2　LuaJIT 的 Debug 模式

LuaJIT　　Debug　　make CCDEBUG=-g　　HO'

```
   # wget http://luajit.org/download/LuaJIT-2.1.0-beta3.tar.gz
   # tar -zxvf LuaJIT-2.1.0-beta3.tar.gz
   # cd  LuaJIT-2.1.0-beta3
   # make CCDEBUG=-g
```

```
# make install
# export LUAJIT_LIB=/usr/local/lib
# export LUAJIT_INC=/usr/local/include/luajit-2.1
```

16.1.3 开启 PCRE 的 Debug 模式

Nginx Â

Nginx --with-pcre= Debug '

```
# cd nginx-1.12.2
# ./configure --with-pcre=/path/to/my/pcre-8.39 \
    --with-pcre-jit  --with-pcre-opt=-g \
```

PCRE rpm ¸'

```
#   wget   http://debuginfo.centos.org/6/x86_64/pcre-debuginfo-7.8-7.el6.
x86_64.rpm
# rpm -ivh pcre-debuginfo-7.8-7.el6.x86_64.rpm
```

Ю 3 '

```
# rpm -qa |grep pcre
pcre-debuginfo-7.8-7.el6.x86_64
pcre-7.8-7.el6.x86_64
pcre-devel-7.8-7.el6.x86_64
```

16.1.4 分析工具下载

Л ¸Л openresty-systemtap-toolkit stapxx é

é è stapxx openresty-systemtap-toolkit

б Ngx_Lua ˜Ω ÂЮ é Ю

Ю'

```
# git clone https://github.com/openresty/openresty-systemtap-toolkit
# git clone https://github.com/openresty/stapxx.git
```

Л Ю FlameGraphˆ FlameGraph Э é

é ~'

```
# git clone https://github.com/brendangregg/FlameGraph.git
```

注意：Nginx Debug --with-debug Linux

Perl 5.6.1

16.1.5　找出不支持 Debug 模式的 lib 库

lib a Debug ╗ lib ˜Ω Â
ㄴ Л∪ ϟ check-debug-info -p pid a Debug lib Â è pid ㄴ Л∪ Nginx
worker PID ㄴ Л∪ è Linux Ю ˆ é a ㄴ Л∪ Nginx
˜ PID。 ϟ 16.1.4 openresty-systemtap-toolkit ϟ Ю'

```
# cd openresty-systemtap-toolkit
# ./check-debug-info -p  30396
File /lib64/ld-2.12.so has no debug info embedded.
File /lib64/libc-2.12.so has no debug info embedded.
File /lib64/libcrypt-2.12.so has no debug info embedded.
File /lib64/libdl-2.12.so has no debug info embedded.
File /lib64/libfreebl3.so has no debug info embedded.
File /lib64/libm-2.12.so has no debug info embedded.
File /lib64/libnss_files-2.12.so has no debug info embedded.
File /lib64/libpthread-2.12.so has no debug info embedded.
File /lib64/libresolv-2.12.so has no debug info embedded.
```

ㄴ Л∪ ϟ Â ㄴ
Л∪ goreplay Nginx Â 13.5 a
Â

Ь URL ㄴ Л∪ Э ㄴ ѫ ㄴ Л∪
é AB ˆ Apache Benchmark Apache é ˜ à Webbench Â

Æ ㄴ Л∪ Â

注意：　Nginx worker PID

16.3.1　连接池使用状态分析

lua-resty-redis lua-resty-mysql é Ю'

```
local ok, err = db:set_keepalive(10000, 100)
if not ok then
```

```
        ngx.say("failed to set keepalive: ", err)
        return
    end
```

ㄐ ᒪᑌ Linux timewait Nginx
ĭ ㄐ ᕮO '

```
# cd openresty-systemtap-toolkit
# ./ngx-lua-conn-pools -p  30396 --luajit20
Tracing 30396 (/usr/local/nginx/sbin/nginx) for LuaJIT 2.0···

pool "172.16.1.51:6301"
    out-of-pool reused connections: 1
    in-pool connections: 2
        reused times (max/avg/min): 29/18/7
    pool capacity: 100

pool "172.16.13.171:6398"
    out-of-pool reused connections: 0
    in-pool connections: 1
        reused times (max/avg/min): 0/0/0
    pool capacity: 100

pool "172.16.1.55:8186"
    out-of-pool reused connections: 0
    in-pool connections: 1
        reused times (max/avg/min): 377/377/377
    pool capacity: 30

pool "172.16.1.7:5689"
    out-of-pool reused connections: 0
    in-pool connections: 1
        reused times (max/avg/min): 1/1/1
    pool capacity: 100

For total 4 connection pool(s) found.
122 microseconds elapsed in the probe handler.
```

指令：ngx-lua-conn-pools

' ngx-lua-conn-pools -p $pid (--luajit20 --lua51)

ᕮ' ㄐ worker Ngx_Lua Â Nginx
Lua 5.1 --lua51' LuaJIT 2.0 --luajit20 LuaJIT2.1 ㄐ ᒪᑌ --luajit20 Â

16-1 Â

表 16-1　连接池信息说明

结　　果	说　　明
pool "172.16.1.51:6301"	Redis à MySQL
out-of-pool reused connections: 1	
in-pool connections: 2	
reused times (max/avg/min): 29/18/7	à　　à　　à
pool capacity: 100	

16.3.2　找出读/写频繁的文件

10

```
# cd openresty-systemtap-toolkit
# ./accessed-files -p 48070 -r
Tracing 48070 (/usr/local/nginx/sbin/nginx)···
Hit Ctrl-C to end.
^C
=== Top 10 file reads ===
#1: 1 times, 2033 bytes reads in file middle_page.lua.
#2: 1 times, 2374 bytes reads in file rand_str.lua.
```

10

```
# ./accessed-files -p 48070 -w
Tracing 48070 (/usr/local/nginx/sbin/nginx)···
Hit Ctrl-C to end.
^C
=== Top 10 file writes ===
#1: 1976 times, 3353103 bytes writes in file access.log-2018-06-28-20-
14-01.log.
#2: 1975 times, 841837 bytes writes in file wireless.access.log.
#3: 1360 times, 2206474 bytes writes in file  access.log.
#5: 17 times, 5598 bytes writes in file error.log.
```

16.3.3　执行阶段耗时分析

Nginx

```
# cd stapxx
```

```
# export PATH=$PWD:$PATH
# ./samples/ngx-single-req-latency.sxx -x  18993
Start tracing process 18993 (/usr/local/nginx/sbin/nginx)···

[1530153801023919]  pid:18993  GET  /zhe800_n_api/search/hot_words?new_
user=1&user_id=0&user_type=0&callback=hot_words
      total: 1235us, accept() ~ header-read: 65us, rewrite: 23us, pre-
access: 23us, access: 26us, content: 985us
      upstream: connect=270us, time-to-first-byte=496us, read=0us
```

./samples/ngx-single-req-latency.sxx -x $pid worker

μs

goreplay

URL

16.3.4 HTTP 连接数和文件打开数分析

worker

```
# cd stapxx
# export PATH=$PWD:$PATH
# ./samples/ngx-count-conns.sxx -x 18993
Start tracing 18993 (/usr/local/nginx/sbin/nginx)···

====== CONNECTIONS ======
Max connections: 102400      #
Free connections: 101854     #
Used connections: 546        #

====== FILES ======
Max files: 102400            #
Open normal files: 20        #
```

```
^C
#1 consul: 38% (5 samples)
#2 events/0: 30% (4 samples)
#3 kblockd/0: 15% (2 samples)
#4 watchdog/0: 7% (1 samples)
#5 zabbix_agentd: 7% (1 samples)
```

16.3.6　正则表达式耗时分析

Nginx　　Ngx_Lua

CPU　　Nginx

```
# cd stapxx
# export PATH=$PWD:$PATH
# ./samples/ngx-pcre-top.sxx --skip-badvars -x  18999
Found exact match for libluajit: /usr/local/lib/libluajit-5.1.so.2.1.0
Found exact match for libpcre: /lib64/libpcre.so.0.0.1
Start tracing 18999 (/usr/local/nginx/sbin/nginx)
Hit Ctrl-C to end.
^C
Top N regexes with longest total running time:
1. pattern //*/: 7921us (total data size: 17947)
2.  pattern  //address/[a-zA-Z]+[0-9]+/view/edit$/:  6801us  (total  data
size: 14684)
3.  pattern  //address/[a-zA-Z]+[0-9]+/view/default$/:  6555us  (total  data
size: 14088)
4.  pattern  //address/[a-zA-Z]+[0-9]+/queryAddressById$/:  6253us  (total
data size: 14684)
5. pattern //address/[a-zA-Z]+[0-9]+/view/add$/: 6161us (total data
size: 14684)
6.  pattern  //address/[a-zA-Z]+[0-9]+/view/query$/:  5834us  (total  data
size: 14684)
7.  pattern  //address/[a-zA-Z]+[0-9]+/view/delete$/:  5634us  (total  data
size: 14088)
8.  pattern  //address/[a-zA-Z]+[0-9]+/get_default$/:  5475us  (total  data
size: 14088)
9. pattern /0/: 3521us (total data size: 7258)
10. pattern //orders/*/: 3088us (total data size: 7530)
```

--arg utime=1　　Ngx_Lua　lua-resty-core　API　　--arg utime

```
# cd stapxx
# export PATH=$PWD:$PATH
```

```
   # ./samples/ngx-pcre-top.sxx --skip-badvars -x  18999 --arg utime=1
   ^[[AFound exact match for libluajit: /usr/local/lib/libluajit-
5.1.so.2.1.0
   Found exact match for libpcre: /lib64/libpcre.so.0.0.1
   Start tracing 18999 (/usr/local/nginx/sbin/nginx)
   Hit Ctrl-C to end.
   ^C
   Top N regexes with longest total running time:
   1. pattern //*/: 3000us (total data size: 20418)
   2. pattern //address/[a-zA-Z]+[0-9]+/view/add$/: 2000us (total data
size: 11302)
   3. pattern //address/[a-zA-Z]+[0-9]+/view/query$/: 2000us (total data
size: 11302)
   4. pattern //address/[a-zA-Z]+[0-9]+/view/manage$/: 2000us (total data
size: 5428)
   5. pattern //mz/catelist/[a-zA-Z]+[0-9]+$/: 1000us (total data size:
1188)
   6. pattern //mz/baoyou/[a-zA-Z]+[0-9]+$/: 1000us (total data size: 594)
   7. pattern //m/list/baoyou/[a-zA-Z]+[0-9]+$/: 1000us (total data size:
594)
   8. pattern //m/detail/*/: 1000us (total data size: 718)
   9. pattern //cn/z_key/*/: 1000us (total data size: 997)
   10. pattern //nnc/orders/[a-zA-Z]{14}.[a-zA-Z]{4}$/: 1000us (total data
size: 1445)
```

```
   # cd stapxx
   # export PATH=$PWD:$PATH
   # ./samples/ngx-pcre-dist.sxx -x  18999
   Found exact match for libpcre: /lib64/libpcre.so.0.0.1
   Start tracing 18999 (/usr/local/nginx/sbin/nginx)
   Hit Ctrl-C to end.
   ^C
   Logarithmic histogram for data length distribution (byte) for 196882
samples:
   (min/avg/max: 0/22/4749)
   value |-------------------------------------------------- count
       0 |                                                    1255
       1 |                                                    1272
       2 |@                                                   3336
       4 |                                                    1481
       8 |@@@@@@@@@@@@@                                      50690
      16 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@                  113128
      32 |@@@@@@                                              23912
      64 |                                                    1341
```

```
     128 |                                                    419
     256 |                                                     30
     512 |                                                     12
    1024 |                                                      4
    2048 |                                                      0
    4096 |                                                      2
    8192 |                                                      0
   16384 |                                                      0
```

Э　　ﻝ　　16μs　　16μs　　113128　Â

./samples/ngx-pcre-dist.sxx -x $pid　　--arg utime=1　　Ю

╗　--arg utime　　ﾉ　　'

```
   # cd stapxx
   # export PATH=$PWD:$PATH
   # ./samples/ngx-pcre-dist.sxx -x  18999 --arg utime=1
   Found exact match for libpcre: /lib64/libpcre.so.0.0.1
   Start tracing 18999 (/usr/local/nginx/sbin/nginx)
   Hit Ctrl-C to end.
   ^C
   Logarithmic histogram for data length distribution (byte) for 46247
samples:
   (min/avg/max: 0/23/805)
   value |-------------------------------------------------- count
       0 |                                                    215
       1 |                                                    214
       2 |@                                                   715
       4 |                                                    349
       8|@@@@@@@@@@@@@@@@@@                                 10117
      16 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@                 28346
      32 |@@@@@@@@@@                                          5973
      64 |                                                    247
     128 |                                                     66
     256 |                                                      3
     512 |                                                      2
    1024 |                                                      0
    2048 |                                                      0
```

16.3.7　找出消耗 CPU 资源较多的指令

Nginx　CPU　　Э　　ﻝ Ⲗ　lua-stacks.sxx　　ﻝ Ⲗ

ﻝ　Nginx　worker　　ϟ　CPU　　Â

LuaJIT 2.1　　ϟ　Ю'

```
# cd stapxx
# export PATH=$PWD:$PATH
# ./samples/lj-lua-stacks.sxx --arg time=10 --skip-badvars -x 36984
  ./samples/lj-lua-stacks.sxx --arg time=10 --skip-badvars -x  48070
```

LuaJIT2.0 Lua5.1 Ю ҙ'

```
# cd openresty-systemtap-toolkit
# ./ngx-sample-lua-bt -p 9768 --luajit20 -t 10 #        Lua5.1,    --
luajit20    --lua51  ↲
```

У Ю '

```
Found exact match for libluajit: /usr/local/lib/libluajit-5.1.so.2.1.0
WARNING: Start tracing 48070 (/usr/local/nginx/sbin/nginx)
WARNING: Please wait for 10 seconds···
WARNING: Time's up. Quitting now···
match
builtin#88
@/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:4
@/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:9
@/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:29
@/usr/local/nginx/conf/lua/log_jk/log_to_influxdb.lua:1
    131
match
builtin#84
@/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:9
@/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:29
@/usr/local/nginx/conf/lua/log_jk/log_to_influxdb.lua:1
    114
compile_regex
C:ngx_http_lua_ngx_re_find
@/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:29
@/usr/local/nginx/conf/lua/log_jk/log_to_influxdb.lua:1
    75
match
C:ngx_http_lua_ngx_exit
    26
lj_str_new
C:ngx_http_lua_var_get
@/usr/local/nginx/conf/lua/log_jk/log_to_influxdb.lua:1
    21
max_expand
builtin#88
@/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:4
```

```
    @/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:9
    @/usr/local/nginx/conf/lua/log_jk/utils/utils.lua:29
    @/usr/local/nginx/conf/lua/log_jk/log_to_influxdb.lua:1
```

a　　CPU　　FlameGraph　Â

16.3.8　利用火焰图展示和分析数据

FlameGraph　　é　ㄴ厸　　è

ЮÂ

16.3.7　　tmp　Ю　a.bt　　'

```
    # cd stapxx
    # export PATH=$PWD:$PATH
    #  ./samples/lj-lua-stacks.sxx  --arg  time=10  --skip-badvars  -x  36984
>/tmp/a.bt
```

ㄲ　　openresty-systemtap-toolkit　　fix-lua-bt　ㄴ a.bt　　Lua ㄱ　　è a_new.bt　　'

```
    # cd openresty-systemtap-toolkit
    # ./fix-lua-bt /tmp/a.bt > /tmp/a_new.bt
```

FlameGraph　　a_new.cbt　　a.svg'

```
    # cd FlameGraph
    # ./stackcollapse-stap.pl /tmp/a_new.bt > /tmp/a_new.cbt
    # ./flamegraph.pl /tmp/a_new.cbt > /tmp/a.svg
```

a.svg　　16-1　　worker　　Â

16-1　worker

a.svg　　ЮÂ

1　　16-2　　*y*　　Â

Ю　　*y*　Э　　Â

x　　CPU　　*x*　Э

Â

Э Ї Ŀ ┘

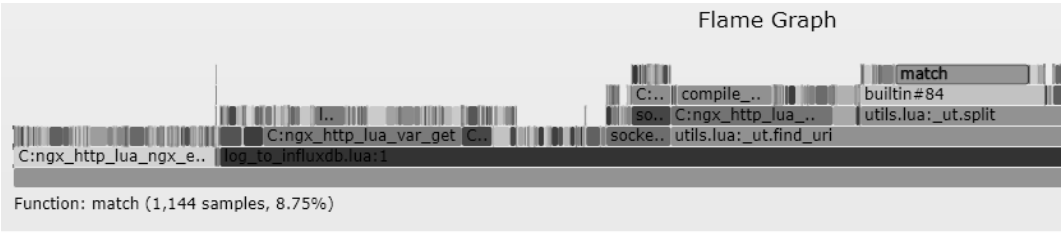
a Ї ĿÂ

2 Л CPU Æ ɛ

Â

16-2 CPU ┘ЛЦ

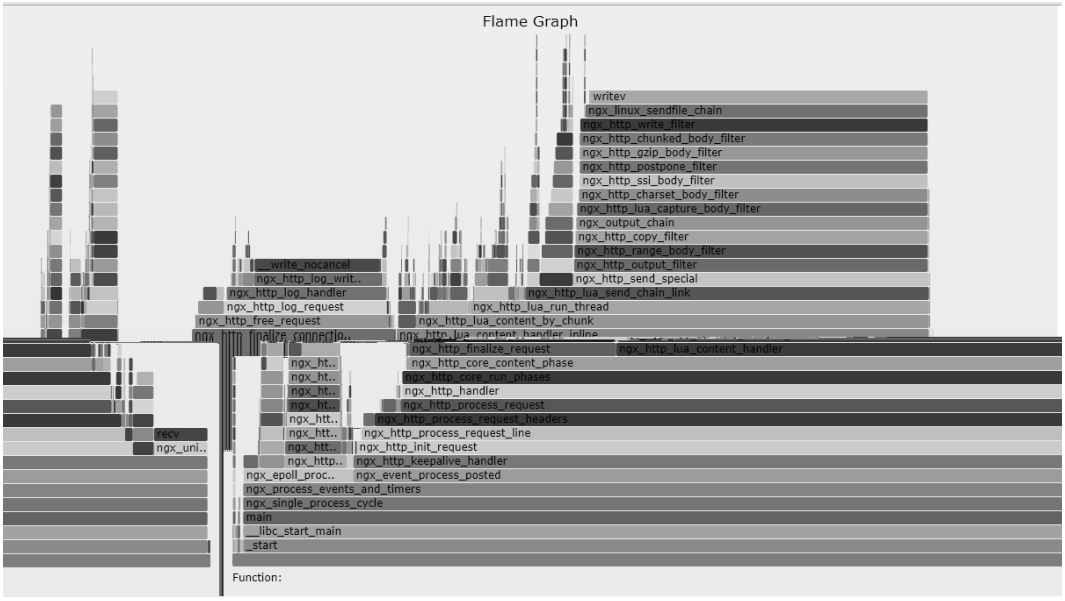
Л ȝ match match 1144 CPU 8.75% Â



16-2 CPU

3 Ї Ŀ Æ ┘Â

16-3 OpenResty Â



16-3 OpenResty

Ngx_Lua

a　　é luacheck

é luacheck

```
# yum install luarocks
# luarocks install luacheck --deps-mode=none
```

--deps-mode=none

У　　Lua　　LuaFileSystem　luacheck

```
# luarocks install luafilesystem
```

У　　Ngx_Lua　　LuaJIT 2.0/2.1　　--std Ngx_Lua　　Lua5.1　　--std lua51 --std

```
# luacheck  --std ngx_lua  /tmp/ab_version.lua

Checking /tmp/ab_version.lua                      2 warnings

    /tmp/ab_version.lua:1:7:  value  assigned  to  variable  version  is
unused
    /tmp/ab_version.lua:2:1: setting non-standard global variable xx

Total: 2 warnings / 0 errors in 1 file, couldn't check 1 file
```

version　xx

openresty-systemtap-toolkit　stapxx

off-cpu

GitHub　Wiki

17

Nginx

OpenResty Nginx OpenResty

注意： OpenResty Nginx
Wiki Nginx OpenResty

OpenResty

OpenResty

```
# wget https://openresty.org/download/openresty-1.13.6.2.tar.gz
# tar -zxvf openresty-1.13.6.2.tar.gz
# cd openresty-1.13.6.2
# ./configure
# make
# sudo make install
```

prefix /usr/local/openresty/ -V OpenResty

```
# /usr/local/openresty/nginx/sbin/nginx  -V
nginx version: openresty/1.13.6.2
built by gcc 4.8.5 20150623 (Red Hat 4.8.5-28) (GCC)
built with OpenSSL 1.0.2k-fips  26 Jan 2017
TLS SNI support enabled
configure arguments:
```

```
   --prefix=/usr/local/openresty/nginx
   --with-cc-opt=-O2
   --add-module=../ngx_devel_kit-0.3.0
   --add-module=../echo-nginx-module-0.61
   --add-module=../xss-nginx-module-0.06 --add-module=../ngx_coolkit-0.2rc3
   --add-module=../set-misc-nginx-module-0.32
   --add-module=../form-input-nginx-module-0.12
   --add-module=../encrypted-session-nginx-module-0.08
   --add-module=../srcache-nginx-module-0.31
   --add-module=../ngx_lua-0.10.13 --add-module=../ngx_lua_upstream-0.07
   --add-module=../headers-more-nginx-module-0.33
   --add-module=../array-var-nginx-module-0.05
   --add-module=../memc-nginx-module-0.19
   --add-module=../redis2-nginx-module-0.15
   --add-module=../redis-nginx-module-0.3.7
   --add-module=../rds-json-nginx-module-0.15
   --add-module=../rds-csv-nginx-module-0.09
   --add-module=../ngx_stream_lua-0.0.5
   --with-ld-opt=-Wl,-rpath,/usr/local/openresty/luajit/lib
   --with-stream --with-stream_ssl_module --with-http_ssl_module
```

OpenResty

Nginx Ngx_Lua

“ ” OpenResty OPM

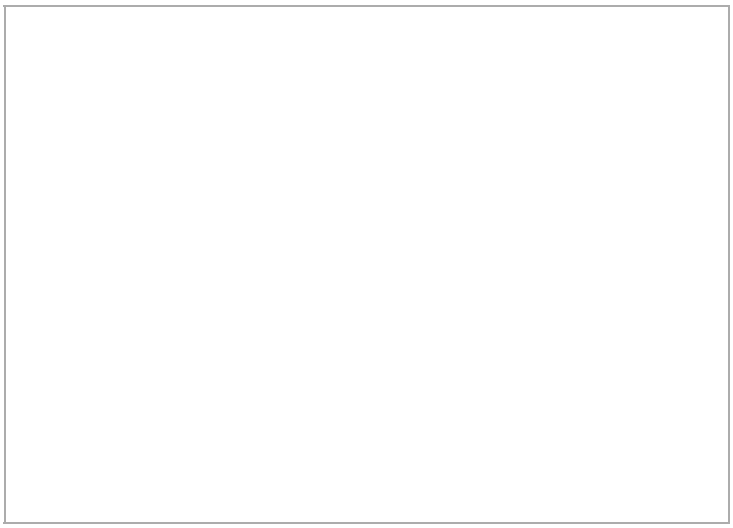
OpenResty Package Manager

OPM 17-1 OPM

http://opm.openresty.org/ OPM

1 2

```
   # /usr/local/openresty/bin/opm  search lua-resty-http
   pintsized/lua-resty-http       Lua   HTTP   client   cosocket   driver   for
OpenResty/ngx_lua
   agentzh/lua-resty-http         Lua   HTTP   client   cosocket   driver   for
OpenResty/ngx_lua
   # /usr/local/openresty/bin/opm  search lua-resty-url
   3scale/lua-resty-url
```



17-1　OPM

2

```
   # /usr/local/openresty/bin/opm get pintsized/lua-resty-http
   * Fetching pintsized/lua-resty-http
     Downloading    https://opm.openresty.org/api/pkg/tarball/pintsized/lua-
resty-http-0.12.opm.tar.gz
     % Total     % Received % Xferd  Average Speed    Time     Time      Time
Current
     Dload  Upload   Total   Spent    Left  Speed
   100 19953  100 19953    0     0   134k      0 --:--:-- --:--:-- --:--:--
134k
   Package  pintsized/lua-resty-http  0.12  installed  successfully  under
/usr/local/openresty/site/
```

3

```
   # /usr/local/openresty/bin/opm update
   * Fetching pintsized/lua-resty-http > 0.12
   Package pintsized/lua-resty-http 0.12 is already the latest version.
   * Fetching 3scale/lua-resty-url > 0.3.3
   Package 3scale/lua-resty-url 0.3.3 is already the latest version.
```

4

```
   # /usr/local/openresty/bin/opm upgrade  pintsized/lua-resty-http
   * Fetching pintsized/lua-resty-http > 0.12
   Package pintsized/lua-resty-http 0.12 is already the latest version.
```

5

```
#  /usr/local/openresty/bin/opm remove pintsized/lua-resty-http
Package pintsized/lua-resty-http 0.12 removed successfully.
```

OPM　Wiki　https://github.com/openresty/opm

API　HTTP
API　Nginx　resolver
IP　resolver　Nginx　“　”
lua-resty-dns

注意：　DNS　IP

lua-resty-dns

```
content_by_lua_block {
            -- OpenResty
            local resolver = require "resty.dns.resolver"
            local r, err = resolver:new{
                nameservers = {"192.168.1.2"},
                retrans = 3,
                timeout = 2000,
            }
            if not r then
                ngx.say("failed to instantiate the resolver: ", err)
                return
            end
            --     test1.zhe800.com   A
            local  answers,  err,  tries  =  r:query("test1.zhe800.com",
{qtype = r.TYPE_A})
            if not answers then
                ngx.say("failed to query the DNS server: ", err)
                ngx.say("retry  historie:\n   ",  table.concat(tries,  "\n
"))
                return
            end
            if answers.errcode then
```

```
                ngx.say("server returned error code: ", answers.errcode,
                        ": ", answers.errstr)
            end
            --            table           DNS
            for i, ans in ipairs(answers) do
                ngx.say(ans.name, " ", ans.address or ans.cname,
                        " type:", ans.type, " class:", ans.class,
                        " ttl:", ans.ttl)
            end
        }
```

A

```
# curl  http://testnginx.com/test
test1.zhe800.com 192.168.1.12 type:1 class:1 ttl:60
test1.zhe800.com 192.168.1.11 type:1 class:1 ttl:60
test1.zhe800.com 192.168.1.10 type:1 class:1 ttl:60
```

DNS DNS

DNS

- lua-resty-dns DNS DNS
- ngx.timer.every DNS Ngx_Lua
- DNS
- Ngx_Lua DNS

注意： https://github.com/bungle/lua-resty-random

Nginx 1.9 TCP/UDP ngx_stream_core_module Ngx_Lua ngx_stream_lua_module TCP/UDP ngx_stream_core_module

```
# stream        TCP/UDP
stream {
    server {
        listen 1111;

        #     ngx.socke.*          memcached
        content_by_lua_block {
            local sock = ngx.socket.tcp()
            sock:settimeout(1000)
            local ok, err = sock:connect("127.0.0.1", 11211)
            if not ok then
               ngx.say("failed to connect: ", err)
               return
            end
            --     memcached   testnginxlua
            local bytes, err = sock:send("get testnginxlua\r\n")
            if not bytes then
                ngx.say("failed to send query: ", err)
                return
            end
            local line, err = sock:receive()
            if not line then
                ngx.say("failed to receive a line: ", err)
                return
            end
            ngx.say("result: ", line)

        }
    }
}
```

telnet　　TCP　　Nginx

```
# telnet  127.0.0.1 1111
Trying 127.0.0.1···
Connected to 127.0.0.1.
Escape character is '^]'.
result: VALUE testnginxlua 0 3
Connection closed by foreign host.
```

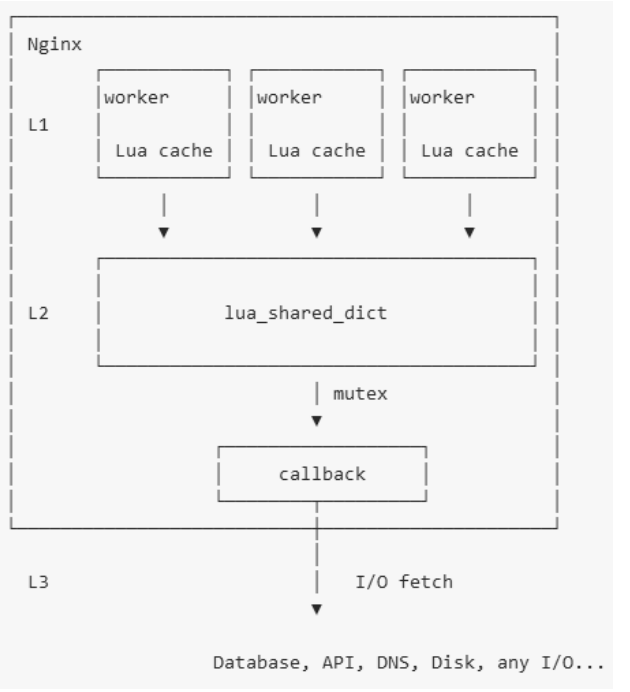
UDP　　“ listen 1111 ”　　“ listen 1111 udp ”

ngx_stream_core_module　　TCP/UDP　　Ngx_Lua

Wiki　　https://github.com/openresty/stream-lua-nginx-module

10 Ngx_Lua lua_shared_dict lua-resty-lrucache lua-resty-mlcache

17-2 lua-resty-mlcache Wiki



17-2 lua-resty-mlcache Wiki

lua-resty-mlcache lua-resty-lrucache 1 lua_shared_dict 2 3

- 1 lua-resty-lrucache lua_shared_dict
- 2 lua_shared_dict 1 2
- 2 lua-resty-lock “ ”
- https://github.com/thibaultcha/lua-resty-mlcache

lua-resty-core　OpenResty　Ngx_Lua
lua-resty-core
https://github.com/openresty/lua-resty-core

17.5.1　字符串分割

lua-resty-core　ngx.re.split

```
location / {
    content_by_lua_block {
       local ngx_re = require "ngx.re"
       --                        res   table
       local res, err = ngx_re.split("test nginx 123", " ")
       --
 for i, v in ipairs(res) do
           ngx.say(v)
       end
    }
}
```

```
# curl  http://127.0.0.1:80/t?sa
test
nginx
123
```

17.5.2　Nginx 进程管理

lua-resty-core　ngx.process　nginx agent
master　Nginx　root　master
agent　root　agent　Nginx
worker

```
init_by_lua_block {
local process = require "ngx.process"
--     agent
    local ok, err = process.enable_privileged_agent()
    if not ok then
        ngx.log(ngx.ERR, "enables privileged agent failed error:", err)
    end
```

```
        ngx.log(ngx.INFO, "process type: ", process.type())
    }
    server {
        listen       80;
        server_name  localhost;
        location = /t {
            content_by_lua_block {
                local process = require "ngx.process"
                --          worker
                process.signal_graceful_exit()
                ngx.say("process type: ", process.type())
            }
        }
    }
```

Nginx Nginx agent

```
    # ps ax |grep nginx
    10204   ?                   Ss            0:00   nginx:   master   process
/usr/local/openresty/nginx/sbin/nginx
    -c /usr/local/openresty/nginx/conf/nginx.conf
    10620 pts/0    R+     0:00 grep --color=auto nginx
    27634 ?        S      0:00 nginx: privileged agent process
    27642 ?        S      0:00 nginx: worker process
```

/t URL

```
    # curl  http://127.0.0.1:80/t
    process type: worker
```

Nginx worker PID

```
    # ps ax |grep nginx
    10204   ?                   Ss            0:00   nginx:   master   process
/usr/local/openresty/nginx/sbin/nginx
    -c /usr/local/openresty/nginx/conf/nginx.conf
    11907 ?        S      0:00 nginx: worker process
    11909 pts/0    S+     0:00 grep --color=auto nginx
    27634 ?        S      0:00 nginx: privileged agent process
```

worker master worker

“ ” Nginx Passenger Ruby Node.js

Ruby Node.js touch

Nginx Ruby Node.js Nginx OpenResty

HTTP touch

touch

注意：　Nginx 1.13.8　　lua-resty-core 0.1.14　　get_master_pid　　master　　PID

Nginx　　UUID　Universally Unique Identifier　　UUID　　lua-resty-jit-uuid　　UUID

OPM　　lua-resty-jit-uuid

```
# export PATH=$PATH:/usr/local/openresty/bin/
# /usr/local/openresty/bin/opm  get thibaultcha/lua-resty-jit-uuid
```

OpenResty　　UUID

```
init_worker_by_lua_block {
        local uuid = require 'resty.jit-uuid'
        uuid.seed()
    }
    server {
        listen 80;
        location / {
            content_by_lua_block {
                local uuid = require 'resty.jit-uuid'
                ngx.say(uuid())
            }
        }
    }
```

```
[root@VM_3_41_centos /]# curl  http://127.0.0.1:80/t?sa
b3ab29cb-dde8-4af9-9d86-2b18c3eae198
[root@VM_3_41_centos /]# curl  http://127.0.0.1:80/t?sa
7acd4fdc-e11b-4d5e-adea-620a5a234463
[root@VM_3_41_centos /]# curl  http://127.0.0.1:80/t?sa
90cceee4-d53d-43ea-a41c-598bfa6ba2c9
[root@VM_3_41_centos /]# curl  http://127.0.0.1:80/t?sa
18532eec-8af6-4301-8565-47abc9771666
```

OpenResty Nginx awesome-resty
OpenResty Nginx Web API

https://github.com/bungle/awesome-resty
“ ”

OpenResty
OpenResty Java
Python PHP Ngx_Lua
Ngx_Lua

OpenResty OpenResty
OpenResty OpenResty
OpenResty

OpenResty OpenResty OpenResty Inc.
OpenResty

18

Nginx

Nginx

Nginx

Nginx

Nginx proxy_buffering

Nginx nobody

error.log error.log

Nginx /

if location “ ” Nginx

if try_files try_files

```
location / {
   root html;
   try_files /ab.html /a.html @test;
 #  if ($arg_a = "1") {
 #      set $args "a=3&b=4";
 #  }
}
location @test {
    return 200 $args;
}
```

/abxx.html /dasda.html

try_files @test loaction test

a=1 if

a=3&b=4

404

if location if proxy

if 404

- if Nginx
- if location server
- Ngx_Lua

location Nginx

testnginx.com/abc/x testnginx.com/abc/c testnginx.com/abc/[0-9]+

```
location ~ /abc {
}
```

/x/abc /abcde /abc/s

/abc.html

location　　“　　”

```
location ~ ^/abc(/|$)  #          /abc/            /abc   URL
location ~  /abc$      #            /abc   URI
location /abc/         #
location ~ ^/abc/(x|c|[0-9]+)$ #
```

401　403　404　405

HTTP

API　　URL

```
testnginx.com/abc/[0-9]+/[a-z]-[0-9]+/api/f.json,
testnginx.com/[a-z]+/api/x.json
```

URL

- 　　URL
  URL
- 　　Nginx

Nginx

```
server {
    listen 80;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_pass http://f_servers;

    location /abc {
        proxy_set_header Test_a 'a';
        proxy_pass http://f_servers;
    }
}
```

404

Host

proxy_set_header proxy_set_header
proxy_set_header

- server location
- Ngx_Lua more_set_input_headers

API HTTPS
HTTPS App
“ ”

Nginx HTTPS HTTP
HTTPS HTTP 2.0

1.10 Nginx TCP --
with-stream HTTPS

```
user webuser webuser;
worker_processes 1;

error_log /var/log/nginx/error.log info;

events {
    worker_connections  1024;
}
#     TCP                                Nginx
stream {
upstream backend {
    # HTTPS
        server 192.168.100.22:443;
    }

    server {
        listen 443 so_keepalive=on;
        # proxy_protocol on;
        # proxy_bind $remote_addr transparent;
        proxy_connect_timeout 300s;
        proxy_timeout 300s;
        proxy_pass backend;
    }
}

http {
    include       mime.types;
    default_type  application/octet-stream;
    server {
   #                                        80
   # HTTPS
        listen       80;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_hide_header X-Runtime;

        location / {
            proxy_pass  http://127.0.0.1:8081;
        }
    }

}
```

Nginx　　443　　HTTPS

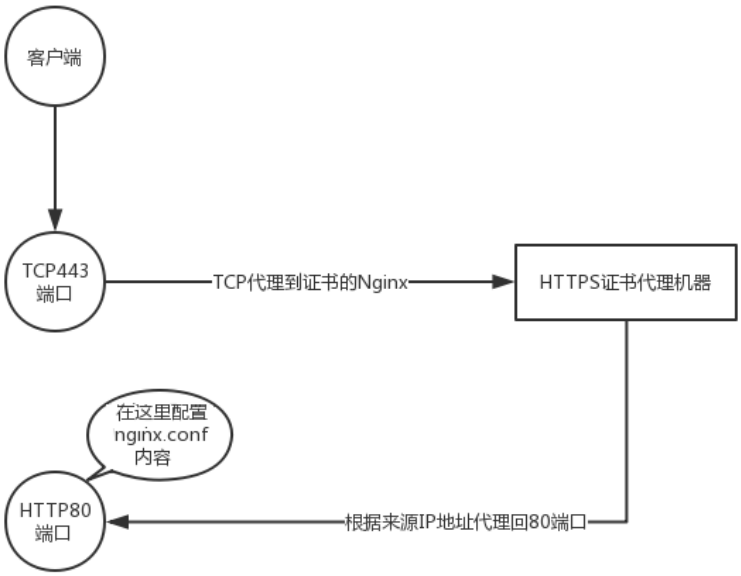
Nginx HTTPS Nginx upstream backend 192.168.100.22

```
server {
    listen              443 ssl ;

    ssl_certificate /path/xxx.crt;
    ssl_certificate_key /path/xxx.key;
    location / {
        # $remote_addr                                    443
        #              $remote_addr                          HTTP
        proxy_pass http://$remote_addr:80;
    }

}
```

192.168.100.22
HTTPS 18-1



18-1 HTTPS

TCP 443 HTTPS
80 80 Nginx

error_page　　proxy_intercept_errors
　　error_page　　Nginx　　http

```
    proxy_intercept_errors on;
```

```
    recursive_error_pages on;  #     error_page
    proxy_intercept_errors on;
```

　　Nginx
　　503
　　Nginx
Nginx

- 　　3.7
-

　　QPS　　0

　　Linux