阿里云开发者社区
ALIBABA CLOUD DEVELOPER COMMUNITY

Nginx 代理系统常用手册

Nginx 常用配置查询手册

阿里云开发者学堂推荐配套教材

藏经阁

阿里云开发者电子书系列

阿里云开发者“藏经阁”

海量电子书免费下载

钉钉扫一扫

进入官方答疑群

开 发 者 学 堂 【 Alibaba Cloud Linux 技术图谱】

更多好课免费学

目录

前言

Nginx 在运维工程师的日常工作中，是一定会用到的，无论是做业务服务器，还是做反向代理服务器，都是不可避免会用到的工具。而在这本电子书中，希望可以给你一个快速查询的手册，帮助你更好的完成 Nginx 的相关工作。

这个速查手册中内容分为以下四个部分：

Nginx 安装：如何在不同的发行版中安装 Nginx。

Nginx 配置文件解读：Nginx 配置文件的完全解读，帮助你在编写 Nginx 配置文件时，快速查询相关资料。

Nginx 命令行操作：为你提供了在配置 Nginx 时，最常用的 Nginx 命令行操作，帮助你快速完成运维工作。

常用 Nginx Snippet：为你准备了 Nginx 的 Snippet，你在工作中遇到的绝大多数场景，都可以在这里找到。

Nginx 安装

CentOS

在 CentOS 中，你可以使用如下命令来安装 Nginx：

```
sudo yum install epel-release
sudo yum install nginx
sudo systemctl enable nginx
sudo systemctl start nginx
```

Ubuntu

在 Ubuntu 中，你可以使用如下命令来安装 Nginx：

```
sudo apt update -y
sudo apt install nginx
sudo systemctl enable nginx
sudo systemctl start nginx
```

Nginx 配置文件

以下为一个标准的 Nginx 配置文件，相关配置项目及解读如下：

```
user  www www; # Nginx 使用的用户和用户组

worker_processes  2; # worker 进程数量

pid /var/run/nginx.pid; # nginx 使用的 PID 文件路径

# 日志等级 [ debug | info | notice | warn | error | crit ]

error_log  /var/log/nginx.error_log  info; # 日志路径、日志等级

events {
    worker_connections   2000; # 每个 Worker 处理的

    # use [ kqueue | epoll | /dev/poll | select | poll ];
    use kqueue; # 事件处理模型
}

http {

    include       conf/mime.types; # 引入 mime-types
    default_type  application/octet-stream; # 设置默认情况下的 mime-type
```

```
log_format main      '$remote_addr - $remote_user [$time_local] '
                     '"$request" $status $bytes_sent '
                     '"$http_referer" "$http_user_agent" '
                     '"$gzip_ratio"'; # 日志格式

log_format download  '$remote_addr - $remote_user [$time_local] '
                     '"$request" $status $bytes_sent '
                     '"$http_referer" "$http_user_agent" '
                     '"$http_range" "$sent_http_content_range"'; # 自定义日志格式

client_header_timeout  3m; #客户端头超时时间
client_body_timeout    3m; #客户端体超时时间
send_timeout           3m; #返回结果超时时间

client_header_buffer_size    1k; # 客户端头缓存大小
large_client_header_buffers  4 4k; # 较大客户端头缓存大小

gzip on; #是否开启 Gzip
gzip_min_length  1100; # Gzip 处理的底线阈值
gzip_buffers     4 8k; # gzip 压缩所用 Buffer
gzip_types       text/plain; # gzip 使用的 mime-type

output_buffers   1 32k; # 输出 buffer
postpone_output  1460; #指定 Nginx 发送给客户端最小的数值，如果可能的话，没有数据会发送，直到达到此值
```

```
    sendfile        on; # 使用 sendfile 发送文件
    tcp_nopush      on; # 仅依赖于 sendfile 的使用。它能够使 Nginx 在一个数据包中尝试发送响应头，以及在数据包中发送一个完整的文件
    tcp_nodelay     on; #启用或禁用 TCP_NODELAY 选项，用于 keep-alive 连接
    send_lowat      12000; # 如果非零，Nginx 将会在客户端套接字尝试减少发送操作

    keepalive_timeout  75 20; #指定 keep-alive 连接持续多久。第二个参数用于在响应头中这只”Keep-Alive”头

    #lingering_time    30; #在使用 lingering_close 指令的连接中，使用该指令指定客户端连接为了处理更多的数据需要保持打开连接的时间
    #lingering_timeout  10; #结合 lingering_close，该指令显示 Nginx 在关闭客户端连接之前，为获得更多数据会等待多久
    #reset_timedout_connection  on; #使用这个指令之后，超时的连接会被立即关闭，释放相关的内存。默认的状态是处于 FIN_WAIT1，这种状态将会一直保持连接


    server {
        listen        one.example.com; # 监听域名
        server_name   one.example.com  www.one.example.com; # 主机域名

        access_log   /var/log/nginx.access_log  main; #记录日志

        location / {
            proxy_pass        http://127.0.0.1/; # 设置代理地址
```

```
        proxy_redirect     off; # 设置跟随转发

        proxy_set_header   Host             $host; # 设置 proxy 头
        proxy_set_header   X-Real-IP        $remote_addr; # 设置真实 IP
        #proxy_set_header  X-Forwarded-For  $proxy_add_x_forwarded_for; # 设置
转发头

        client_max_body_size       10m; # 客户端请求最大值
        client_body_buffer_size    128k; # 客户端请求 Buffer 最大值

        client_body_temp_path      /var/nginx/client_body_temp; # 客户端请求临时
路径

        proxy_connect_timeout      70; # 客户端链接超时
        proxy_send_timeout         90;  # 客户端发送超时
        proxy_read_timeout         90; # 客户端接收超时
        proxy_send_lowat           12000; # 客户端链接超时

        proxy_buffer_size          4k; # 代理 Buffer 大小
        proxy_buffers              4 32k; # 代理大量 buffer 配置
        proxy_busy_buffers_size    64k; # 代理较忙碌情况下 Buffer 配置
        proxy_temp_file_write_size 64k; # 代理写文件缓存配置

        proxy_temp_path            /var/nginx/proxy_temp; # 代理缓存路径

        charset utf-8; # charset 设置
```

```
}

error_page  404  /404.html; # 配置 404 页面

location = /404.html { # 设置 404 页面的规则
    root  /spool/www;  # 设置 404 页面使用 /spool/www 目录
}

location /old_stuff/ { # 设置 /old_staff/规则
    rewrite    ^/old_stuff/(.*)$  /new_stuff/$1  permanent; # 301 跳转到新的规则
}

location /download/ { # 设置 /download 规则

    valid_referers  none  blocked  server_names  *.example.com;

    if ($invalid_referer) {  #屏蔽不是来自 *.example.com 的请求
        #rewrite    ^/    http://www.example.com/;
        return   403;
    }

    #rewrite_log  on;

    # rewrite /download/*/mp3/*.any_ext to /download/*/mp3/*.mp3
    rewrite ^/(download/.*)/mp3/(.*)\..*$
```

```
            /$1/mp3/$2.mp3                  break;


        root         /spool/www;
        #autoindex    on; # 是否开启自动 Index
        access_log   /var/log/nginx-download.access_log  download;
    }


    location ~* \.(jpg|jpeg|gif)$ {
        root         /spool/www;
        access_log   off;
        expires      30d;
    }
  }
}
```

Nginx 命令操作

Nginx 的常用命令行操作如下：

- `nginx -t` 测试 Nginx 配置文件是否符合要求；
- `nginx -s reload` 重新加载 Nginx 配置文件；
- `nginx -V` 查看 Nginx 版本信息、编译参数等。

常用 Nginx 的 snippet

从 @ 域跳转到 www 域，加上 www

```
server {
    listen 80;
    server_name example.org;
    return 301 $scheme://www.example.org$request_uri;
}

server {
    listen 80;
    server_name www.example.org;
    ...
}
```

从 www 域跳转至 @域，去除 www

```
server {
      listen 80;
      server_name www.domain.com;


      return 301 $scheme://domain.com$request_uri;
}

server {
```

```
    listen 80;
    server_name domain.com;



    # nginx config for virtualhost goes here
}
```

整站 301 跳转

```
server {
    server_name old-site.com
    return 301 $scheme://new-site.com$request_uri;
}
```

设置 Edge 使用最新的 IE 内核渲染

```
add_header "X-UA-Compatible" "ie=edge";
```

设置内容过期时间

```
add_header Cache-Control "max-age=31536000, immutable";
```

设置强制 HTTPS

```
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains"
always;
```

设置代理头

```
proxy_set_header Host $http_host;
```

```
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme;
```

开启 Gzip 压缩

```
gzip  on;
gzip_buffers 16 8k;
gzip_comp_level 6;
gzip_http_version 1.1;
gzip_min_length 256;
gzip_proxied any;
gzip_vary on;
gzip_types
    text/xml application/xml application/atom+xml application/rss+xml
application/xhtml+xml image/svg+xml
    text/javascript application/javascript application/x-javascript
    text/x-json application/json application/x-web-app-manifest+json
    text/css text/plain text/x-component
    font/opentype application/x-font-ttf application/vnd.ms-fontobject
    image/x-icon;
gzip_disable  "msie6";
```

开启 Nginx 文件缓存

```
open_file_cache max=1000 inactive=20s;
open_file_cache_valid 30s;
```

```
open_file_cache_min_uses 2;

open_file_cache_errors on;
```

开启 Nginx SSL 缓存

```
ssl_session_cache shared:SSL:10m;

ssl_session_timeout 10m;
```

移除结尾多余的 /

```
if ($request_uri ~ "^[^?]*?//") {

    return 301 $scheme://$host$uri$is_args$args;

}
```

为不含 / 结尾的 URL 加上 /

```
if ($uri !~ "\.[a-z0-9]{2,4}$") {

    rewrite "[^/]$" $scheme://$host$uri/ permanent;

}
```

开启 Nginx 监控

```
location /status {

  stub_status on;

  access_log off;

}
```

屏蔽特定 IP

```
location / {
```

```
  # block one workstation
  deny    192.168.1.1;
  # allow anyone in 192.168.1.0/24
  allow   192.168.1.0/24;
  # drop rest of the world
  deny    all;
}
```

隐藏除了 .well-known 以外的所有隐藏文件

```
location ~ /\.(?!well-known).* {
      access_log off;
      log_not_found off;
      return 404;
}
```

隐藏 .git

```
location ~ /\.git {
      access_log off;
      log_not_found off;
      return 404;
}
```

隐藏 .htaccess

```
location ~ /\.ht {
      access_log off;
```

```
    log_not_found off;
    return 404;
}
```