Create By zhang.pn

目　录

Nginx的安装与使用

安装 :zap:

> 下载 安装包 nginx-1.16.1.tar.gz http://nginx.org/en/download.html

编译与安装 :zap:

```
./configure --prefix=/usr/local/nginx
make
make install
```

- 报错

  ```
  ./configure: error: the HTTP rewrite module requires the PCRE library.
  You can either disable the module by using --without-http_rewrite_module
  option, or install the PCRE library into the system, or build the PCRE libr
  ary
  statically from the source with nginx by using --with-pcre=<path> option.
  ```

  解决

  ```
  yum -y install pcre-devel
  ```

- 报错

  ```
  ./configure: error: the HTTP gzip module requires the zlib library.
  You can either disable the module by using --without-http_gzip_module
  option, or install the zlib library into the system, or build the zlib libr
  ary
  statically from the source with nginx by using --with-zlib=<path> option.
  ```

  解决

  ```
  yum install -y zlib-devel
  ```

使用 :zap:

检查配置 :zap:

```
nginx -t
```

启动程序 :zap:

```
nginx
```

重载配置 :zap:

```
nginx -s reload
```

立即停止 :zap:

```
nginx -s stop
```

平滑停止 :zap:

```
nginx -s quit
```

查看进程 :zap:

```
ps -ef | grep nginx
```

第一章：HTTP代理

```
location / {
    root /home/html;
    index index.html;
    expires 30d;
    gzip on;
    gzip_comp_level 3;
    gzip_types text/css text/xml application/javascript text/plain application/
msword application/pdf application/x-javascript;
}
```

第二章：反向代理

- 代理配置

```
location ^~ /api/ {
    proxy_pass http://127.0.0.1:8080;
    proxy_redirect 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;
}
```

- WebSocket代理配置

```
location ^~ /api/socket/ {
    proxy_pass http://127.0.0.1:8080;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "Upgrade";
    proxy_set_header X-Real-IP $remote_addr;
}
```

第三章：expires 缓存

Nginx官网文档

> 在回应请求时，在响应头中添加[Expires]和[Cache-Control],起到控制页面缓存的作用。

```
location ~ \.(jpg|png|gif)$ {
    root /home/images;
    expires 30d;  # 表示 30天后过期。
}
```

- expires -1; 表示永久过期。

校验Expires、Cache-Control(max_age)头，若未过期，则直接使用缓存。

若已过期，则：

校验Etag，若存在则校验Etag，ID一致则返回304使用缓存，若一致则服务器返回最新文件。

若不存在Etag，则：

校验Last-Modified，若一致则返回304使用缓存，若一致则服务器返回最新文件。

第四章：查看编译参数

```
nginx -V
```

- 测试结果

```
zhangpn@pnivy:~$ sudo nginx -V
nginx version: nginx/1.14.0 (Ubuntu)
built with OpenSSL 1.1.1  11 Sep 2018
TLS SNI support enabled
configure arguments: --with-cc-opt='-g -O2 -fdebug-prefix-map=/build/nginx-DUgh
aW/nginx-1.14.0=. -fstack-protector-strong -Wformat -Werror=format-security -fP
IC -Wdate-time -D_FORTIFY_SOURCE=2' --with-ld-opt='-Wl,-Bsymbolic-functions -Wl
,-z,relro -Wl,-z,now -fPIC' --prefix=/usr/share/nginx --conf-path=/etc/nginx/ng
inx.conf --http-log-path=/var/log/nginx/access.log --error-log-path=/var/log/ng
inx/error.log --lock-path=/var/lock/nginx.lock --pid-path=/run/nginx.pid --modu
les-path=/usr/lib/nginx/modules --http-client-body-temp-path=/var/lib/nginx/bod
y --http-fastcgi-temp-path=/var/lib/nginx/fastcgi --http-proxy-temp-path=/var/l
ib/nginx/proxy --http-scgi-temp-path=/var/lib/nginx/scgi --http-uwsgi-temp-path=
/var/lib/nginx/uwsgi --with-debug --with-pcre-jit --with-http_ssl_module --with-
http_stub_status_module --with-http_realip_module --with-http_auth_request_modu
le --with-http_v2_module --with-http_dav_module --with-http_slice_module --with-
threads --with-http_addition_module --with-http_geoip_module=dynamic --with-htt
p_gunzip_module --with-http_gzip_static_module --with-http_image_filter_module=
dynamic --with-http_sub_module --with-http_xslt_module=dynamic --with-stream=dy
namic --with-stream_ssl_module --with-mail=dynamic --with-mail_ssl_module
```

第五章：日志

Nginx官网文档

设置参考

```
HTTP标准头(HEADER)：User-Agent
```

```
Nginx中变量：http_user_agent
```

第六章：tcp_nopush

Nginx官网文档

多个包，一起发送，高效。

The options are enabled only when sendfile is used.

该选项仅仅在开启 sendfile 时，生效。

第七章：use用法

Nginx官网文档 - use

Nginx官网文档 - epoll

可以指定使用的方法，通常不需要指定，因为nginx通常会默认使用最有效的方法。

```
events {
    use epoll;
}
```

第八章：tcp_nodelay

Nginx官网文档

与tcp_nodelay相反，立即传送。不要延迟。

条件是，在keepalive长连接的状态下使用。

支持websocket代理。

第九章：跨站访问

HTTP标准头：Access-Control-Allow-Origin

```
add_header Access-Control-Allow-Origin *;
```

第十章：random_index

Nginx官网文档

随机指定一个首页。

第十一章：with_http_sub_module

Nginx官网文档

过滤串

第十二章：Nginx如何处理请求

Nginx官网文档

> 假设有3台虚拟主机都监听80端口

```
server {     # 主机 1
    listen      80;
    server_name example.org www.example.org;
    ...
}

server {     # 主机 2
    listen      80;
    server_name example.net www.example.net;
    ...
}

server {     # 主机 3
    listen      80;
    server_name example.com www.example.com;
    ...
}
```

> 当有一个请求发生时，通过请求头 `Host` 决定哪台主机相应。当 `Host` 是 `example.net` 时，则第二台主机响应。如果请求头 `Host` 是未知的(`未匹配`)则默认为第一台主机响应，但是，若某台主机的 `listen` 属性有 `default_server` 修饰，则默认由该主机响应。如：

```
server {
    listen      80 default_server;
    server_name example.net www.example.net;
    ...
}
```

第十三章：Nginx的location指令

Nginx官网文档

```
location [ = | ~ | ~* | ^~ ] uri { ... }
```

> A location can either be defined by a prefix string, or by a regular expression. Regular expressions are specified with the preceding “ `~*` ” modifier (for case-insensitive matching), or the “ `~` ” modifier (for case-sensitive matching). To find location matching a given request, nginx first checks locations defined using the prefix strings (prefix locations). Among them, the location with the longest matching prefix is selected and remembered. Then regular expressions are checked, in the order of their appearance in the configuration file. The search of regular expressions terminates on the first match, and the corresponding configuration is used. If no match with a regular expression is found then the configuration of the prefix location remembered earlier is used.

`~*` 表示正则忽略大小写匹配。 `~` 表示正则不忽略大小写匹配。
先检查前缀匹配，路径最长匹配的location将被记录下来；再检查正则匹配，按照在配置文件出现位置的顺序；一旦正则匹配到，该被匹配的location配置将被使用。否则使用被记录下的路径最长匹配的location。

> If the longest matching prefix location has the “ `^~` ” modifier then regular expressions are not checked.

如果最长匹配前缀位置具有 `^~` 修饰符，则不检查正则表达式。

> Also, using the “ `=` ” modifier it is possible to define an exact match of URI and location. If an exact match is found, the search terminates. For example, if a “ `/` ” request happens frequently, defining “ `location = /` ” will speed up the processing of these requests, as search terminates right after the first comparison. Such a location cannot obviously contain nested locations.

`=` 表示精确匹配。
一旦精确匹配成功，则匹配立刻终止。例如，如果请求 `/` 十分频繁，那么设置 `location = /` 将会提高速度。

总结：匹配类型大体分为 `精确` 、 `前缀` 、 `正则` 。前缀分为 `普通` 前缀和带 `^~` 前缀。正则分是否区分大小写两种。精确匹配优先级最高(匹配即终止)。
然后在 `前缀` 、 `正则` 中选择：首先前缀匹配选出最长路径匹配的location，判断其是否有 `^~` 修饰；若有则立即终止；若无则记录下此最长匹配路径的location，进行正则匹配；正则匹配按照在配置文件中的顺序依次匹配，若匹配到则终止；若未匹配到则使用记录下的最长匹配路径的location。流程图展示

Nginx 指令 官方文档

Nginx 变量 官方文档

第十四章：proxy_redirect指令

Nginx官网文档

设置重定向或刷新时的内容。

> Sets the text that should be changed in the “Location” and “Refresh” header fields of a proxied server response. Suppose a proxied server returned the header field “Location: `http://localhost:8000/two/some/uri/`”.

```
proxy_redirect http://localhost:8000/two/ http://frontend/one/;
```

原本在头中有Location属性，值为 `http://localhost:8000/two/some/uri/` ，通过以上指令，客户端接受到的指令是： `http://frontend/one/some/uri/` 。

> will rewrite this string to “`Location: http://frontend/one/some/uri/`”.

第十五章：proxy_set_header指令

Nginx官网文档

请求头重新定义。

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

第十六章：proxy_pass指令

Nginx官方文档

```
proxy_pass http://localhost:8000/uri/;
```

可以是具体的IP，也可是域名。

URI传递规则：(关于proxy_pass的参数路径问题)

> If the proxy_pass directive is specified with a URI, then when a request is passed to the server, the part of a normalized request URI matching the location is replaced by a URI specified in the directive:

```
location /name/ {
    proxy_pass http://127.0.0.1/remote/;
}
```

如果 proxy_pass 包含有URI，那么location的path将会被替换。说白了，举个例子：

如果请求是 `http://www.demo.com/name/gosuncn/index.html` 那么，被location的 `/name/` 匹配到了，然后转发给代理时， `/name/` 将会被proxy_pass替换掉：

`http://127.0.0.1/remote/gosuncn/index.html` 。

> If proxy_pass is specified without a URI, the request URI is passed to the server in the same form as sent by a client when the original request is processed, or the full normalized request URI is passed when processing the changed URI:

```
location /some/path/ {
    proxy_pass http://127.0.0.1;
}
```

这种情况，比如传来了 `http://www.demo.com/some/path/gosuncn/index.html` ,转发给代理后：

`http://127.0.0.1/some/path/gosuncn/index.html` 。

如果location使用正则表达式，那么proxy_pass不应该带有URI。

如果使用了rewrite命令，那么proxy_pass也不应该带有URI。

在proxy_pass中使用变量时，proxy_pass也不应该带有URI。

WebSocket 代理时，需要其他特殊配置。

第十七章：root和alias指令

Nginx官方文档alias

Nginx官方文档root

语法:	`root path;`
默认值:	`root html;`
使用位置:	`http` , `server` , `location` , `if in location`

```
location /i/ {
    root /data/w3;
}
```

请求： `/i/top.gif` ，服务器将 `/data/w3/i/top.gif` 文件响应给请求。

请求： `/i/top.gif`

响应： `/data/w3/i/top.gif`

语法:	`alias path;`
默认值:	—
使用位置:	`location`

```
location /i/ {
    alias /data/w3/images/;
}
```

请求： `/i/top.gif` ，服务器将 `/data/w3/images/top.gif` 文件响应给请求。

请求： `/i/top.gif`

响应： `/data/w3/images/top.gif`

第十八章：sendfile指令

Nginx官方文档

设置为on表示启动高效传输文件的模式。

sendfile可以让Nginx在传输文件时直接在磁盘和tcp socket之间传输数据。

如果这个参数不开启，会先在用户空间（Nginx进程空间）申请一个buffer，用read函数把数据从磁盘读到cache，再从cache读取到用户空间的buffer，再用write函数把数据从用户空间的buffer写入到内核的buffer，最后到tcp socket。

开启这个参数后可以让数据不用经过用户buffer。

```
location /video/ {
    sendfile       on;
    tcp_nopush     on;
    aio            on;
}
```

零拷贝是什么?博客

nginx AIO机制与sendfile机制

aio官方异步文件读取

```
location /video/ {
    sendfile on;
    sendfile_max_chunk 256k;
    aio threads;
    directio 512k;
    output_buffers 1 128k;
}
```