Nginx

- □□□□

□□□□

nginx□□□□

```
server {    listen 80 default_server;
        listen [::]:80 default_server;
# □□□ HTTPS
        rewrite ^(.*) https://$host$1 permanent;
 }
server {
# SSL □□
        listen 443 ssl default_server;
        listen [::]:443 ssl default_server;
# □□□□□□□□
        ssl_certificate /etc/nginx/ssl/nginx.crt;
        ssl_certificate_key /etc/nginx/ssl/nginx.key;
# ... }
```

proxy □□□□

```
  location / {
        proxy_pass_header Server;
        proxy_redirect off;
#         proxy_set_header Host $Host;
        proxy_set_header Host $http_Host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-scheme $scheme;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Host $server_name;
        proxy_pass https://survey.dreamteenet.com;
#               try_files $uri $uri/ =404;
        }
```

SSL□□□□

```
ssl_protocols TLSv1.2 TLSv1.3;
ssl_prefer_server_ciphers on;
ssl_ciphers EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:RSA+AES128:RSA+AES256;
ssl_ecdh_curve secp384r1;
ssl_session_cache shared:SSL:10m;
ssl_session_tickets off;
ssl_stapling on;
ssl_stapling_verify on;
resolver 168.95.1.1 168.95.192.1 valid=300s;
resolver_timeout 5s;
add_header X-Frame-Options SAMEORIGIN;
add_header X-Content-Type-Options nosniff;
ssl_dhparam /etc/ssl/certs/dhparam2.pem;
```

proxy_redirect□□□

```
location /one/ {
proxy_pass       http://upstream:port/kevin/;
proxy_redirect   default;
}

location /one/ {
proxy_pass       http://upstream:port/kevin/;
proxy_redirect   http://upstream:port/kevin/   /one/;
}
```

□□□□□□

http{} #□□□□□□□□
server{} #□□□□□□□□
location{} #□□□□□□□□□□□

nginx□□□□□□□□

```
server {
...
client_max_body_size 10M;
...
}
```

備註：預設**2MB**，記得**php**也需要修改**fpm**

```
client_max_body_size 0; #無限制
client_body_timeout 300s #設定timeout時間
```

變量名

$host 请求信息中的 Host，如果请求中没有 Host 行，则等于设置的服务器名
$request_method 客户端请求类型，如 GET、POST
$remote_addr 客户端的 IP 地址
$args 请求中的参数
$content_length 请求头中的 Content-length 字段
$http_user_agent 客户端 agent 信息
$http_cookie 客户端 cookie 信息
$remote_addr 客户端的 IP 地址
$remote_port 客户端的端口
$server_protocol 请求使用的协议，如 HTTP/1.0、HTTP/1.1\
$server_addr 服务器地址
$server_name 服务器名称
$server_port 服务器的端口号

location匹配规则

语法规则: location [=|~|~*|^~] /uri/ { … }
模式 含义
location = /uri = 表示精确匹配，只有完全匹配上才能生效
location ^~ /uri ^~ 开头对URL路径进行前缀匹配，并且在正则之前。
location ~ pattern 开头表示区分大小写的正则匹配
location ~* pattern 开头表示不区分大小写的正则匹配
location /uri 不带任何修饰符，也表示前缀匹配，但是在正则匹配之后
location / 通用匹配，任何未匹配到其它location的请求都会匹配到，相当于switch中的default

前缀匹配时，Nginx 不对 url 做编码，因此请求为 /static/20%/aa，可以被规则 ^~ /static/ /aa 匹配到（注意是空格）

多个 location 配置的情况下匹配顺序为:

首先精确匹配 =
其次前缀匹配 ^~
其次是按文件中顺序的正则匹配
然后匹配不带任何修饰的前缀匹配。
最后是交给 / 通用匹配
当有匹配成功时候，停止匹配，按当前匹配规则处理请求

意：前缀匹配，如果有包含关系时，按最大匹配原则进行匹配。比如在前缀匹配：location /dir01 与 location /dir01/dir02，如有请求 http://localhost/dir01/

例子，有如下匹配规则：

```
location = / {
   echo "规则A";
}
location = /login {
   echo "规则B";
}
location ^~ /static/ {
   echo "规则C";
}
location ^~ /static/files {
    echo "规则X";
}
location ~ \.(gif|jpg|png|js|css)$ {
   echo "规则D";
}
location ~* \.png$ {
   echo "规则E";
}
location /img {
    echo "规则Y";
}
location / {
   echo "规则F";
}
```

那么产生的效果如下：

访问根目录 /，比如 http://localhost/ 将匹配 规则A

访问 http://localhost/login 将匹配 规则B，http://localhost/register 则匹配 规则F

访问 http://localhost/static/a.html 将匹配 规则C

访问 http://localhost/static/files/a.exe 将匹配 规则X，虽然 规则C 也能匹配到，但因为最大匹配原则，最终选中了 规则X。你可以测试下，去掉规则 X ，则当前 URL

访问 http://localhost/a.gif, http://localhost/b.jpg 将匹配 规则D 和 规则 E ，但是 规则 D 顺序优先，规则 E 不起作用，而 http://localhos

访问 http://localhost/a.PNG 则匹配 规则 E ，而不会匹配 规则 D ，因为 规则 E 不区分大小写。

访问 http://localhost/img/a.gif 会匹配上 规则D,虽然 规则Y 也可以匹配上，但是因为正则匹配优先，而忽略了 规则Y。

访问 http://localhost/img/a.tiff 会匹配上 规则Y。

访问 http://localhost/category/id/1111 则最终匹配到规则 F ，因为以上规则都不匹配，这个时候应该是 Nginx 转发请求给后端应用服务器，比如 FastCGI (php

理解Nginx中Server和Location的匹配逻辑

nginx日誌

字段	作用
$remote_addr与$http_x_forwarded_for	记录客户端IP地址
$remote_user	记录客户端用户名称

$request 记录请求的URI和HTTP协议
$status 记录请求状态
$body_bytes_sent 发送给客户端的字节数，不包括响应头的大小
$bytes_sent 发送给客户端的总字节数
$connection 连接的序列号
$connection_requests 当前通过一个连接获得的请求数量
$msec 日志写入时间。单位为秒，精度是毫秒
$pipe 如果请求是通过HTTP流水线(pipelined)发送，pipe值为“p”，否则为“.”
$http_referer 记录从哪个页面链接访问过来的
$http_user_agent 记录客户端浏览器相关信息
$request_length 请求的长度（包括请求行，请求头和请求正文）
$request_time 请求处理时间，单位为秒，精度毫秒
$time_iso8601 ISO8601标准格式下的本地时间
$time_local 记录访问时间与时区

nginx负载均衡

Upstream 指定后端服务器地址列表，在 server 中拦截响应请求，并将请求转发到 Upstream 中配置的服务器列表。

```
upstream balanceServer {
    server 10.1.22.33:12345;
    server 10.1.22.34:12345;
    server 10.1.22.35:12345;
}

server {
    server_name  fe.server.com;
    listen 80;
    location /api {
        proxy_pass http://balanceServer;
    }
}
```

add header的設定

```
# HTTP response headers borrowed from Nextcloud `.htaccess`
    add_header Referrer-Policy                "no-referrer"   always;
    add_header X-Content-Type-Options         "nosniff"       always;
    add_header X-Download-Options             "noopen"        always;
    add_header X-Frame-Options                "SAMEORIGIN"    always;
    add_header X-Permitted-Cross-Domain-Policies "none"       always;
    add_header X-Robots-Tag                   "noindex, nofollow"
```

```
always;
    add_header X-XSS-Protection               "1; mode=block" always;
```

x-robot-tag介紹

Noindex: 告訴搜索引擎不要索引該網頁。
Index: 告訴搜索引擎索引該網頁，不必特意加入此標籤，因為這是預設的。
Follow:即使網頁沒有被索引，網頁爬蟲也應該追蹤網頁上的所有鏈接，並將權限傳遞給連結的網頁。
Nofollow: 告訴網頁爬蟲也不要追蹤網頁上的所有鏈接，並將不可以將權限傳遞給連結的網頁。
Noimageindex: 告訴網頁爬蟲不要索引該網頁上的任何圖像。
None:相當於同時使用noindex和nofollow標籤。
Noarchive:搜索引擎不應該在SERP(搜尋結果頁面)上顯示暫存的連結到這個網頁。
Nocache:與noarchive相同，但只用於Internet Explorer和Firefox瀏覽器。
Nosnippet:告訴搜索引擎不要在SERP(搜尋結果頁面)上顯示該網頁的一個片段（即Meta描述）。
Noodyp/noydir [OBSOLETE]:阻止搜索引擎使用DMOZ的網頁描述作為此頁面的SERP(搜尋結果頁面)片段說明。但是，DMOZ在2017年初關站，這個標籤已經過時了
Unavailable_after: 搜索引擎不應該在特定的日期之後索引該網頁。

noindex: 指示搜索引擎不應該索引這個頁面的內容。
X-Robots-Tag: noindex

nofollow: 指示搜索引擎不應該跟隨頁面中的連結。
X-Robots-Tag: nofollow

noarchive: 指示搜索引擎不應該將這個頁面存檔。
X-Robots-Tag: noarchive

nosnippet: 指示搜索引擎不應該顯示這個頁面的摘要或片段。
X-Robots-Tag: nosnippet

max-snippet:[number]: 指定搜索引擎在搜索結果中顯示的最大摘要長度（以字符為單位）。
X-Robots-Tag: max-snippet:150

max-video-preview:[number]: 指定搜索引擎在搜索結果中顯示的最大影片預覽長度（以秒為單位）。
X-Robots-Tag: max-video-preview:30

noindex, nofollow: 組合使用，指示搜索引擎不應該索引該頁面，並且不應該跟隨頁面中的連結。
X-Robots-Tag: noindex, nofollow

這些標頭應該在服務器端生成並添加到 HTTP 響應中，以便告訴搜索引擎如何處理這些頁面。具體的設置方式取決於您使用的服務器和網站架構。例如，在 Nginx 中，您可以使用 add_header 指

nginx

```
location /path/to/page {
    add_header X-Robots-Tag "noindex, nofollow";
}
```

這將使得位於 /path/to/page 的頁面在響應時添加 X-Robots-Tag: noindex, nofollow 標頭，告訴搜索引擎不要索引這個頁面，並且不要跟隨頁面中的連結。