

9.7_Nginx

1、 Nginx 安装

1、 下载 nginx 源码包

```
# wget http://nginx.org/download/nginx-1.20.1.tar.gz
```

2、 安装 nginx

```
# tar xf nginx-1.20.1.tar.gz
```

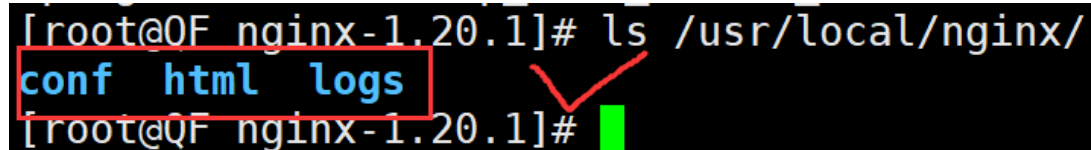
```
# cd nginx-1.20.1/
```

```
# yum install pcre-devel openssl-devel -y
```

```
//安装编译环境所需要的依赖
```

```
# ./configure --prefix=/usr/local/nginx --sbin-path=/sbin/nginx --user=nginx --group=nginx --with-http_stub_status_module --with-http_ssl_module
```

```
# make && make install
```



```
[root@0F nginx-1.20.1]# ls /usr/local/nginx/  
conf  html  logs  
[root@QF nginx-1.20.1]#
```

```
//安装结束后如果有这三个目录代表安装成功
```

3、 启动 Nginx

```
# useradd -s /sbin/nologin -M nginx
```

```
//创建 nginx 用户管理 nginx 的进程
```

```
# systemctl stop httpd
```

```
//防止 80 端口冲突，先暂时将 apache 服务关闭
```

```
# nginx //启动
```

```
# nginx -v //查看版本
```

```
# nginx -V //查看编译选项
```

```
# nginx -t //检查配置文件语法
```

```
# nginx -s stop //停止服务
```

```
# nginx -s reload //平滑重启
```

nginx 的配置文件详解:

nginx 主配置文件主要有以下几大块

1、全局块: 配置影响 nginx 全局的指令。一般有运行 nginx 服务器的用户组, nginx 进程 pid 存放路径, 日志存放路径, 配置文件引入, 允许生成 worker process 数等。

2、events 块: 配置影响 nginx 服务器或与用户的网络连接。有每个进程的最大连接数, 选取哪种事件驱动模型处理连接请求, 是否允许同时接受多个网路连接, 开启多个网络连接序列化等。epoll

3、http 块: 可以嵌套多个 server, 配置代理, 缓存, 日志定义等绝大多数功能和第三方模块的配置。如文件引入, mime-type 定义, 日志自定义, 是否使用 sendfile 传输文件, 连接超时时间, 单连接请求数等。

4、server 块: 配置虚拟主机的相关参数, 一个 http 中可以有多个 server。

5、location 块: 配置请求的路由, 以及各种页面的处理情况。

2、Nginx 状态访问

状态访问统计:

```
# vim /usr/local/nginx/conf/nginx.conf
```

在 server 中添加如下行

```
location = /status {
    stub_status on;
    access_log off;
}
```

查看访问状态统计:

浏览器:http://公网 IP 地址/status 刷新可得到如下变化结果

Active connections: 5

server accepts handled requests

32 32 7

Reading: 0 Writing: 1 Waiting: 4

```

server {
    listen      80;
    server_name localhost;

    #charset koi8-r;

    #access_log  logs/host.access.log  main;

    location / {
        root    html;
        index   index.html index.htm;
    }
    location = /status {
        stub_status on;
        access_log off;
    }
}

```

3、Nginx 自定义日志

自己定义日志:在日志部分写入

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

(http 块写入, 配置日志格式)

log_format cust '\$remote_addr : \$time_local : \$http_user_agent';

(server 块写入)

access_log logs/test.log cust; #cust:自定义的日志名称

测试自定义日志: # curl 自己的 IP

tail -Of /usr/local/nginx/logs/test.log

访问的 Ip:访问的时间:访问的客户端的类型

常见的日志变量:

\$remote_addr	记录客户端的 ip
\$remote_user	记录远程客户端的名字
\$time_local	记录访问时间
\$request	记录请求的 URL
\$status	记录请求状态
\$body_bytes_sent	记录发送给客户端的文件的內容的大小
\$http_referer	记录从哪个页面链接访问过来的
\$http_user_agent	记录客户端浏览器的信息
\$http_x_forwarded_for	记录代理 ip

更多内嵌变量详见:

http://tengine.taobao.org/nginx_docs/cn/docs/http/nginx_http_core_module.html#variables

4、访问控制

访问控制:

有时我们会有这么一种需求,就是你的网站并不想提供一个公共的访问或者某些页面不希望公开,我们希望的是某些特定的客户端可以访问.

那么我们可以在访问时要求进行身份认证,就如给你自己的家门加一把锁,以拒绝那些不速之客.

我们在服务课程中学习过 apache 的访问控制,对于 Nginx 来说同样可以实现,并且整个过程和 Apache 非常的相似.

用户认证:

```
location / {  
    root    html;  
    index  index.html index.htm;  
    auth_basic "haha";           #服务器描述信息  
    auth_basic_user_file /usr/local/nginx/passwd.db;  #存放用户名和密码的文件  
}
```

```
[root@web html]# htpasswd -c /usr/local/nginx/passwd.db user1
```

New password:

Re-type new password:

Adding password for user user1

访问控制: deny/allow 顺序:从上到下

```
location / {  
    root    html;  
    index  index.html index.htm;  
    allow   192.168.10.0/24;  
    deny    all;  
}
```

apache:

```
order allow,deny  
allow from 192.168.10.0/24  
deny from all
```

限速:

使用 limit_rate 指令

Syntax: limit_rate rate;

Default: limit_rate 0;

Context: http, server, location, if in location

案例配置:

```

#access_log logs/host.access.log main;
access_log logs/test.log test;
location / {
    root html;
    index index.html index.htm;
    #deny 171.34.164.15;
    #deny 124.71.170.184;
    #deny 182.102.144.208; } 拒绝这三个IP访问
    auth_basic "请充值VIP";
    auth_basic_user_file /usr/local/nginx/passwd.db; } 认证访问
    limit_rate 20k; 限速, 固定加载下载速度为20K
}
location = /status {
    stub_status on;
    access_log off;
}

#error_page 404 /404.html;

# redirect server error pages to the static page /50x.html
#
"/usr/local/nginx/conf/nginx.conf" 129L, 3046C

```

5、Nginx 虚拟主机配置

虚拟主机:

基于域名的虚拟主机:

(虚拟主机最好用子配置文件进行配置)

- 1、在 http {} 块添加一行 include /usr/local/nginx/conf.d/*.conf;
- 2、mkdir /usr/local/nginx/conf.d/

1.解析

vim /etc/hosts //添加以下内容

自己的 IP www.google.com

自己的 IP www.baidu.com

自己的 IP www.163.com

2.修改配置文件

vim /usr/local/nginx/conf.d/google.conf

```

server {
    listen      80;
    server_name www.google.com;
    location / {
        root    html/google;
        index   index.html index.htm;
    }
}

```

```
# vim /usr/local/nginx/conf.d/baidu.conf
server {
    listen      80 default;           #default 在浏览器中直接键入 IP 地址会进
入这个
    server_name www.baidu.com;
    location / {
        root    html/baidu;          #发布目录在 html 下的 163 目录
        index   index.html index.htm;
    }
}
```

测试:

```
# mkdir /usr/local/nginx/html/{google.baidu} //同时创建 2 个目录
//分别在两个目录下写上 index.html 测试页面
```

```
# nginx -t
# nginx -s reload
```

最后的访问结果：

```
[root@QF nginx]# curl www.google.com
this is google!
[root@QF nginx]# curl www.baidu.com
this is baidu
[root@QF nginx]#
```

6、Nginx 的反向代理

反向代理服务器(反向加速服务器)

目的：外网客户端通过代理服务器，能够访问内网服务器的资源

原理：外网客户端访问正常的域名或者 IP，其实访问的是代理服务器，代理服务器帮助客户端请求页面，在代理服务器上缓存，然后再发送给客户端。

反向代理:

该功能由 ngx_http_proxy_module 模块提供

```
location / {
    proxy_pass http://192.168.10.12;
}
```

案例：代理访问 apache 部署的 qq 农场

为了防止端口冲突，将 apache 服务端口修改为 88

```
# vim /etc/httpd/conf/httpd.conf
```

Listen 80 改为 88

systemctl restart httpd

```
server {  
    listen 80 default;  
    server_name      www.163.com;  
    location / {  
        proxy_pass http://124.71.170.184:88/upload/home/;  
        index index.html;  
    }  
}
```

↓
LAMP修改完端口后，完整的URL访问地址

"/usr/local/nginx/conf.d/163.conf" 8L, 144C

结果：



7、Nginx 负载均衡



该功能由 ngx_http_upstream_module+ngx_http_proxy_module 模块提供

调度算法:负载均衡:

轮循 静态

加权轮循 静态

least_conn: 根据其权重值, 将请求发送到活跃连接数最少的那台服务器, 动态

ip_hash: 把同一客户端的请求调度到同一台真实服务器上.

1. 轮循 - 后端每台服务器的权重相同

```
upstream webs {
    server www.google.com:81;
    server www.baidu.com:82;
}
server {
    listen 80 default;
    server_name www.163.com;
    location / {
        proxy_pass http://webs;
    }
}
```

后端服务器=虚拟主机google.conf
后端服务器=虚拟主机baidu.conf

2.加权轮循: 在每台服务器上加入权值,权值越高的服务器分配到的请求越多

```
upstream webs {
    server 192.168.10.12 weight=1;
    server 192.168.10.13 weight=2;
```



```
}
```

```
-----  
upstream qfedu {  
    server 192.168.10.11 weight=1 max_fails=2 fail_timeout=30s;  
    server 192.168.10.12 weight=2;  
    server 192.168.10.13 backup;    #backup:备份,其他服务器全部宕机后启用  
}
```

3. ip_hash: 让同一客户端在一定时间内访问到同一台服务器

IPv4 地址的前三个字节或者 IPv6 的整个地址，会被用来作为一个散列 key。这种方法可以确保从同一个客户端过来的请求，会被传给同一台服务器。

```
upstream qfedu {  
    ip_hash;  
    server 172.16.0.10;  
    server 172.16.5.100;  
    server 172.16.16.100;  
}
```

客户端测试:

同一客户端访问一直得到同一个页面.

8、Nginx location URI 匹配

location 用于匹配客户请求 uri

语法:

```
location [修饰符] /uri|pattern {...}
```

修饰符:

=	精确匹配,优先级最高
^~	前缀匹配,优先级高于正则匹配
~	正则匹配,区分大小写
~*	正则匹配,不区分大小写
	没有修饰符,优先级最低

无修饰符的情况也表示 uri 前缀匹配,但优先级低于正则匹配

若 uri 为/,表示匹配所有.其他 location 未匹配到的均能匹配到

优先级顺序:

1. location = /uri
2. location ^~ /uri
3. location ~ pattern location ~* pattern

4. location /uri

5. location /

案例：

```
server {
    listen          80;
    server_name     www.qf.com;
    access_log      logs/qf.access.log;
    location = / {
        return 403;
    }

    location / {
        root        /www/qf;
        index       index.html;
    }

    location ^~ /imgs/ {
        root        /www;
    }

    location ~ /\.jpg$ {
        root        html;
    }
}
```

server 配置如上图:

1.客户端访问 www.qf.com/ => 匹配到 1 2 → 403

2.客户端访问 www.qf.com/index.html => 匹配 2 -> /www/qf/index.html

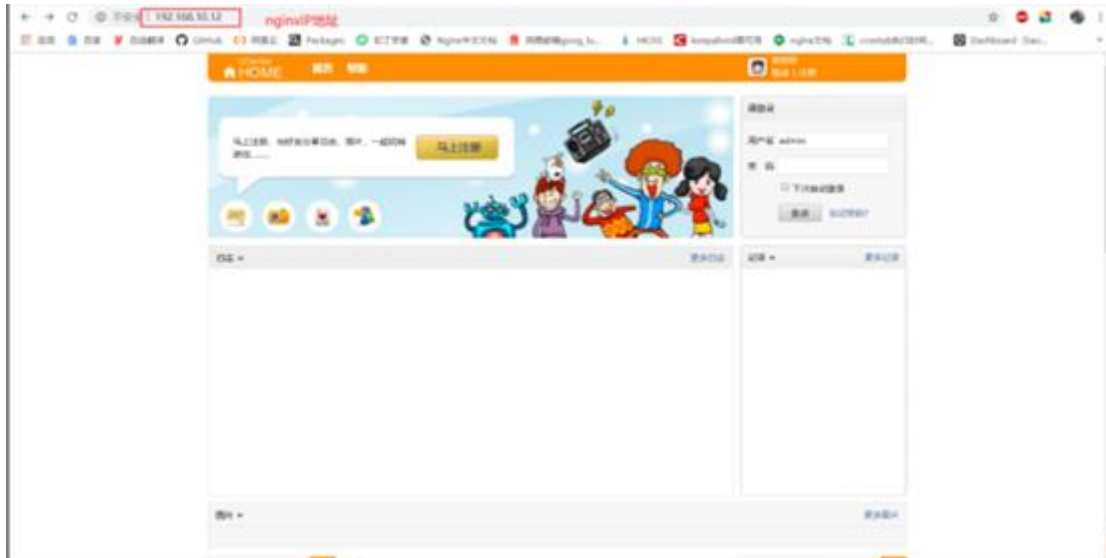
3.客户端访问 www.qf.com/imgs/a.jpg => 匹配 2 3 4 -> /www/imgs/a.jpg

4. 客户端访问 www.qf.com/a.jpg => 匹配 2 4 -> /usr/local/nginx/html/a.jpg

nginx location 测试题目

公司初期，需要部署新的项目工程，只有服务器一台，由你进行项目规划和部署，要求如下：

- 1、要求访问服务器 IP 可以直接访问到 qq 农场登入界面



由于只有一台服务器，推荐将 apache 监听端口换成 88 （下面两条命令更换端口）

```
# sed -i "s/Listen 80/Listen 88/g" /etc/httpd/conf/httpd.conf
```

```
# systemctl restart httpd
```

- 2、要求 NGINX 服务同时要承担公司官网，OA，考勤

在后端机器上部署 nginx 虚拟主机，

nginx 配置要求

访问 <http://qf.com/web1> 跳转到后端 nginx 网站 OA

访问 <http://qf.com/web2> 跳转到后端 nginx 网站 官网

访问 <http://qf.com/web3> 跳转到后端 nginx 网站 考勤

步骤+截图图文结合

```
[root@QF ~]# cat /etc/hosts
::1      localhost        localhost.localdomain  localhost6      localhost6.l
127.0.0.1 localhost        localhost.localdomain  localhost4      loca
127.0.0.1      hecs-x-medium-2-linux-20210301103657  hecs-x-medium-2-linu
124.71.170.184 qf.com

[root@QF ~]# curl http://qf.com/web1
这是官网
[root@QF ~]# curl http://qf.com/web2
这是OA
[root@QF ~]# curl http://qf.com/web3
这是考勤
[root@QF ~]#
```

最后验证结果

答案:

```
server {
    listen 80 default;
    server_name www.qf.com;
    charset utf-8;
    location / {
        proxy_pass http://124.71.170.184:88/upload/home/;
        index index.php;
    }
    location ^~ /web1 {
        root html;
        index index.html;
    }
    location ^~ /web2 {
        root html;
        index index.html;
    }
    location ^~ /web3 {
        root html;
        index index.html;
    }
}

~
~
"/usr/local/nginx/conf.d/163.conf" 21L, 328C
```

4,1-8

9、基于 LNMP 上线 wordpress 项目

LNMP linux+nginx+mysql+php

实验:搭建个人博客

一. 安装 nginx(略)

二. 安装 mysql(略)

三. 安装 php

```
# yum -y install php php-mysql php-devel php-gd php-fpm php-mcrypt php-mbstring php-xml
```

```
# systemctl start php-fpm
```

四. 配置 nginx 支持 php

```
# vim /usr/local/nginx/conf.d/wordpress.conf
```

```
server {
    listen 80 default;
    server_name www.wordpress.com;
```

```

        charset utf-8;
        location / {
            root /usr/local/nginx/html/wordpress;
            index index.php index.html;
        }
        location ~ \.php$ {
            root          /usr/local/nginx/html/wordpress;
            fastcgi_pass   127.0.0.1:9000;
            fastcgi_index  index.php;
            fastcgi_param  SCRIPT_FILENAME  $document_root$fastcgi_script_name;
            include         fastcgi_params;
        }
    }
}

```

五. 测试 nginx 跟 php 的联动

```

# mkdir /usr/local/nginx/html/wordpress
# vim /usr/local/nginx/html/wordpress/index.php
<?php
    phpinfo();
?>

```

（观察页面是否有 php 版本测试页面）

六. 部署 WEB 应用

```

# rz    //上传 wordpress-4.5.3-zh_CN.tar.gz 项目包
# rm -rf /usr/local/nginx/html/wordpress/
# tar xf wordpress-4.5.3-zh_CN.tar.gz -C /usr/local/nginx/html/

```

七. 启动服务

```

# nginx -s reload
# systemctl restart php-fpm
# netstat -tnlp    //查看 9000 端口和 80 端口是否正常运行

```

八. 进 WEB 页面安装

打开浏览器进入 <http://192.168.10.21/wordpress>

注意填写的用户名和密码和数据库

```

# systemctl start mysqld
# mysql -uroot -p'Qf..2021'
mysql> create database wordpress;

```



请在下方填写您的数据库连接信息。如果您不确定，请联系您的服务提供商。

数据库名	<input type="text" value="wp"/>	将WordPress安装到哪个数据库？
用户名	<input type="text" value="wpadmin"/>	您的数据库用户名。
密码	<input type="password" value="密码"/>	您的数据库密码。
数据库主机	<input type="text" value="192.168.10.21"/>	如果localhost不能用，您通常可以从网站服务提供商处得到正确的信息。
表前缀	<input type="text" value="wp_"/>	如果您希望在同一个数据库安装多个WordPress，请修改前缀。

提交



抱歉，我不能写入wp-config.php文件。

您可以手工创建wp-config.php文件并将以下信息贴入其中。

```
<?php
/**
 * WordPress基础配置文件。
 *
 * 这个文件被安装程序用于自动生成wp-config.php配置文件，
 * 您可以不使用网站，您需要手动复制这个文件，
 * 并重命名为“wp-config.php”，然后填入相关信息。
 *
 * 本文件包含以下配置选项：
 *
 * * MySQL设置
 * * 密钥
 * * 数据库表名前缀
 * * ABSPATH
 */
```

完成之后，请点击“进行安装”。

进行安装

vim /usr/local/nginx/html/wordpress/wp-config.php

最后会要求你填写站点名称，管理员用户名和密码，之后就正常登入