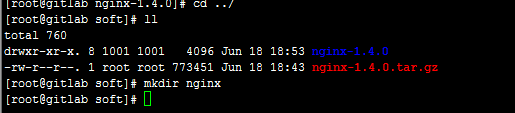
1. 源码下载地址  
   [http://nginx.org/download/选择要下载的版本](http://nginx.org/download/%E9%80%89%E6%8B%A9%E8%A6%81%E4%B8%8B%E8%BD%BD%E7%9A%84%E7%89%88%E6%9C%AC)
2. 解压源码  
   tar -zxvf nginx-1.4.0.tar.gz
3. 创建文件夹nginx；mkdir nginx  
   
4. 进入到源码包目录下配置安装包路径 （usr/soft/nginx为安装路径）

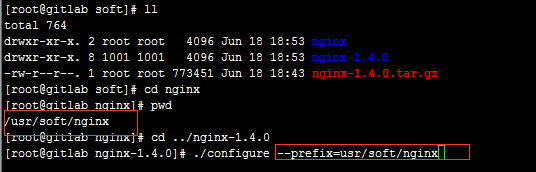
[root@gitlab soft]# cd nginx

[root@gitlab nginx]# pwd

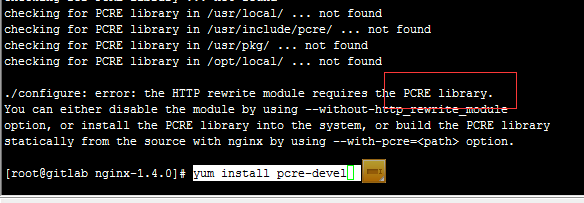
/usr/soft/nginx

[root@gitlab nginx]# cd ../nginx-1.4.0

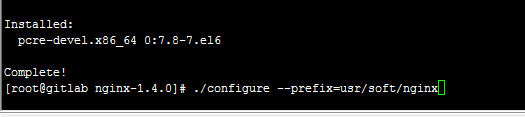
[root@gitlab nginx-1.4.0]# ./configure --prefix=usr/soft/nginx

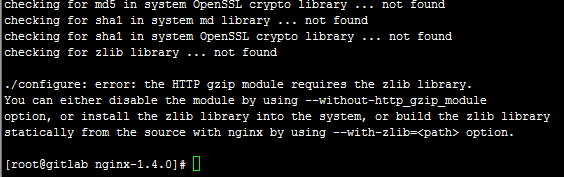


1. yum install pcre-devel（因为缺少包报错）



1. 再次执行命令./configure --prefix=usr/soft/nginx

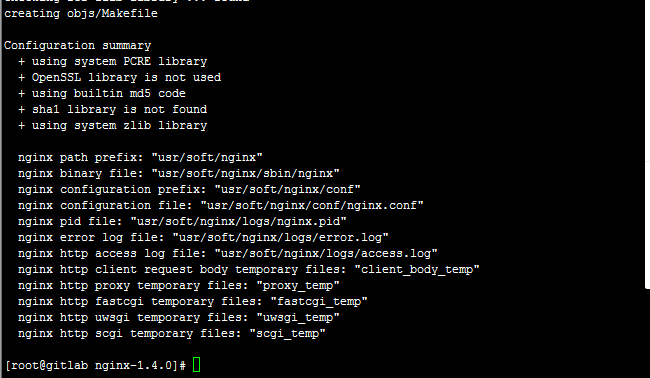




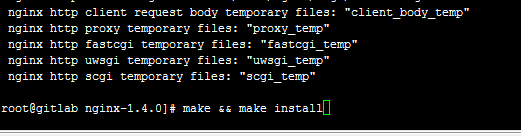
又报错了

1. yum install zlib-devel

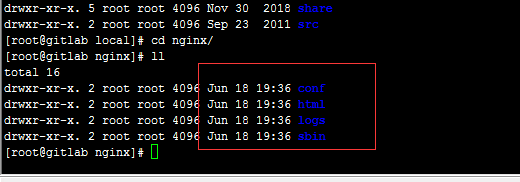
再次执行命令./configure --prefix=usr/soft/nginx



1. 执行make && make install

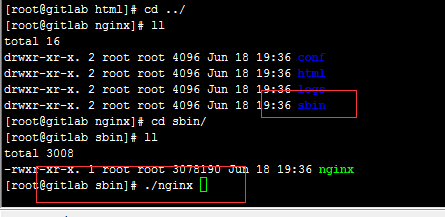


1. 执行完了之后，定义的安装路径下应该有

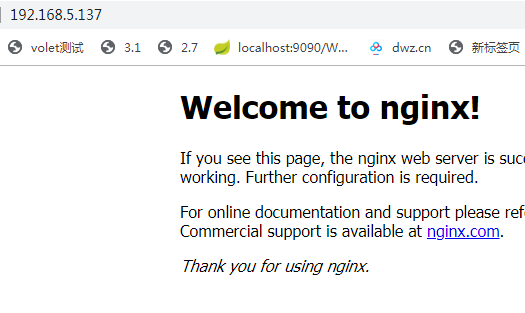


如果没有生成这些文件，可以考虑更换安装目录

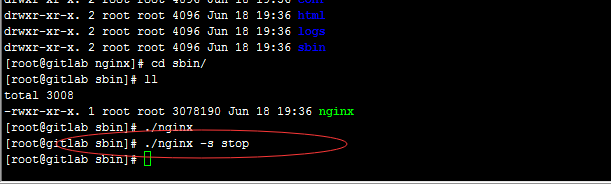
1. 启动nginx ./nginx



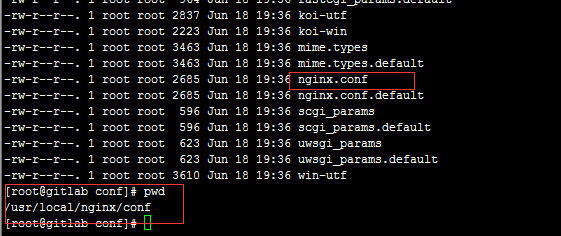
1. 访问所在服务器的IP，出现以下画面，证明启动成功



1. Stop nginx



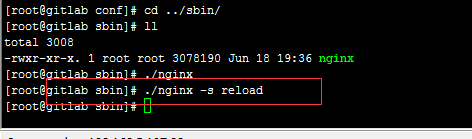
1. Nginx核心配置文件



Nginx.conf分为3段，

Main段 events（连接数量）http

1. 虚拟主机配置



server {

listen 80;

server\_name localhost;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

root html;

index index.html index.htm;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

# proxy the PHP scripts to Apache listening on 127.0.0.1:80

#

#location ~ \.php$ {

# proxy\_pass http://127.0.0.1;

#}

# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000

#

#location ~ \.php$ {

# root html;

# fastcgi\_pass 127.0.0.1:9000;

# fastcgi\_index index.php;

# fastcgi\_param SCRIPT\_FILENAME /scripts$fastcgi\_script\_name;

# include fastcgi\_params;

#}

# deny access to .htaccess files, if Apache's document root

# concurs with nginx's one

#

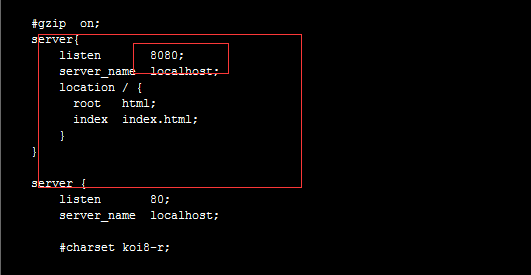
#location ~ /\.ht {

# deny all;

#}

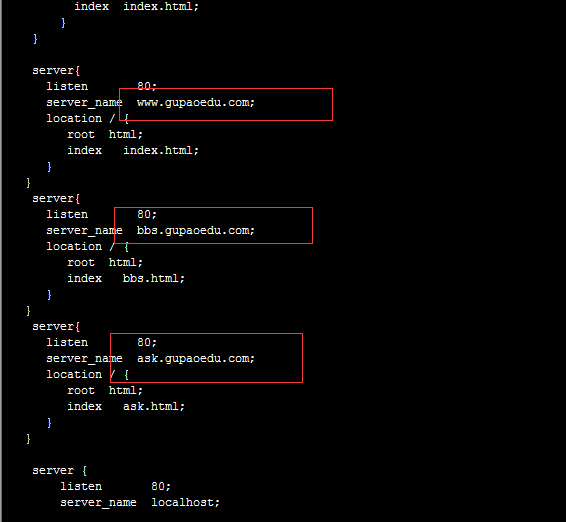
}

1. 基于ip的虚拟主机
2. 基于端口号的虚拟主机



1. 基于域名的虚拟主机

可以修改C:\Windows\System32\drivers\etc下的hosts文件进行假的dns解析



1. Location
2. 配置语法
3. 配置规则

精准匹配

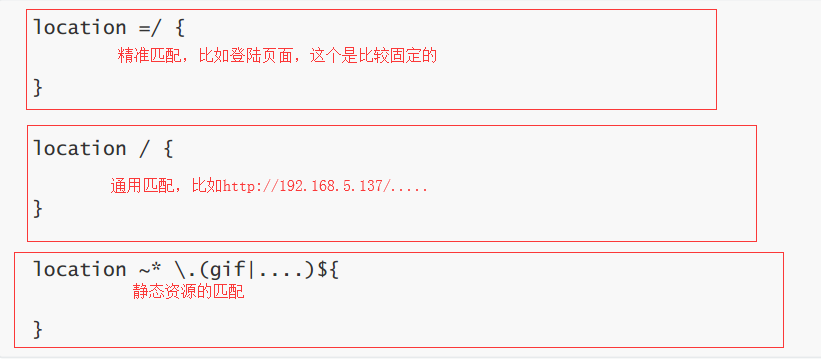
前缀匹配

通用匹配

1. 规则匹配的优先级



实际使用的建议如下：



server {

listen 80;

server\_name localhost;

location / {

root html;

index index.html index.htm;

}

location =/index.html{ 精准匹配

root html/jz;

index index.html

}

location ^~/.../../

}

1. 模块

Nginx模块

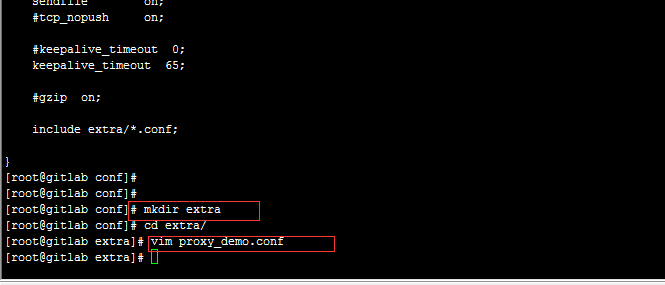
1. 核心模块 ngx\_http\_core\_module
2. 标准模块 http模块
3. 第三方模块

查看以前的配置路径

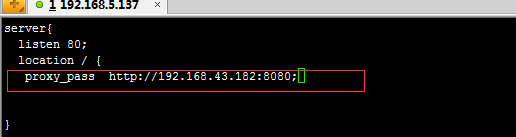
./nginx –V

17.代理配置实现 反向代理



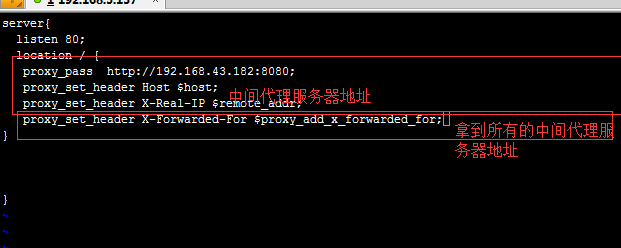


Proxy\_dmeo.conf文件内容如下：



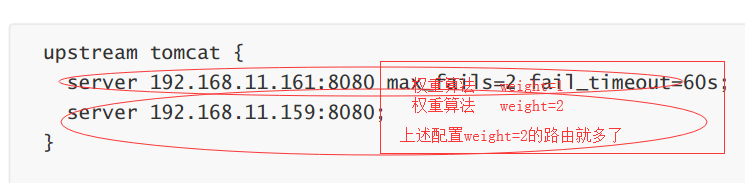
这样就实现了服务器的代理

下面的图是拿到中间原始服务器地址和中间所有代理服务器地址的配置



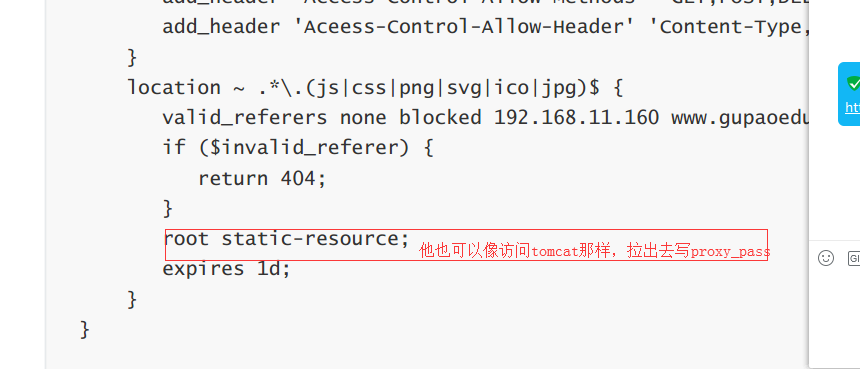
1. 负载均衡配置

古炮第二节笔记有



1. 动静分离

静态资源也可以放到专门的服务器上，如下



Nginx可以作为缓存服务器 比如，静态资源图片

Nginx高可用解决方案

安装keepalived

查看keepalived状态 service keepalived status

Keepalived配置文件 路径 etc/keepalived/keepalive.conf