# 1.Nginx的一些理解

## 1.1 Nginx的一些概念

概念：Nginx是一个高性能的HTTP和反向代理服务器，也是一个IMAP/POP3/SMTP服务器

特点：占用内存少，并发能力强，使用nginx常用的网站：百度、京东、新浪、网易、腾讯、淘宝

兼容性：Linux系统、Window NT系统

优势：支持高达50000个并发数连接的响应，Nginx为我们选择了epoll and kqueue作为开发模型

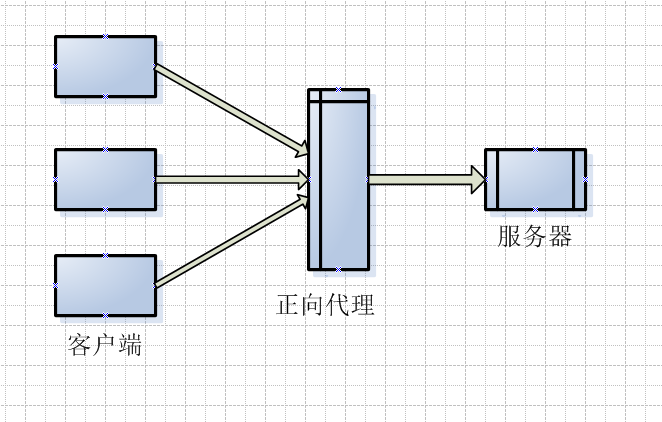
## 1.2 一些关键词

**关于代理**

一个代表，一个渠道，此时涉及到两个角色，一个叫被代理角色，一个叫目标角色，被代理角色通过这个代理访问目标角色完成一些任务的过程称为代理操作过程。举例说明：客户去专卖店买商品，客户称为目标角色，专卖店就是个代理，商品的生产厂家就是这个被代理角色

**正向代理**

我们通过浏览器要去访问国外的网站，是没有办法访问，需要操作FQ进行访问，FQ实际上就是个代理服务器，我们将请求发送到代理服务器，代理服务器去访问国外的网站，然后将访问到的数据传递给我们

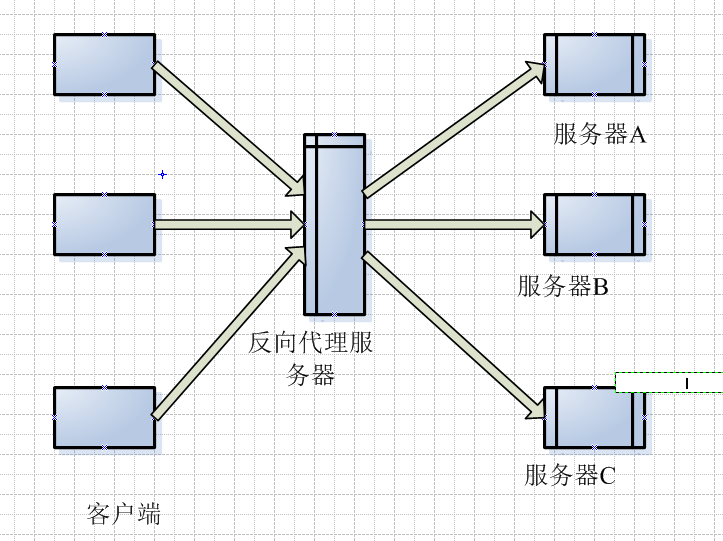


特点：客户端非常明确要访问的服务器地址，服务器只清楚来自哪个代理服务器，而不清楚是来自哪个用户。正向代理屏蔽或者隐藏了真实客户端信息

用途：

1. 访问原来无法访问的资源
2. 可以做缓存，加速访问资源
3. 对客户端进行授权，上网进行认证
4. 代理可以记录用户访问记录，对外隐藏用户信息

**反向代理**



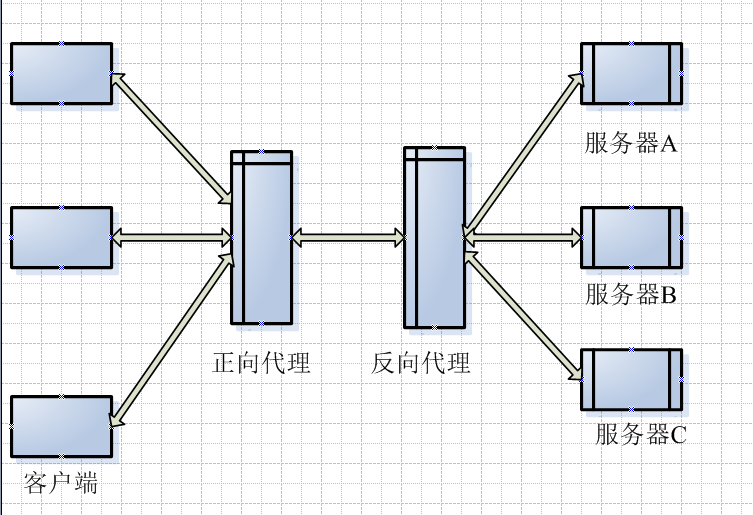
多个客户端给服务器发送请求，Nginx服务器接受到后，按照一定的规则分发给后端的业务处理服务器进行处理，此时请求的来源也就是客户端是明确的，但是具体由那台服务器处理就不明确了，Nginx就是扮演一个反向代理角色

用途：

1. 保证内网的安全，通常将反向代理作为公网访问地址，Web服务器就是内网
2. 负载均衡，通过反向代理服务器来优化网站的负载

## 1.3项目的场景

通常情况下，正向代理和反向代理很可能存在同一个应用场景中



# Nginx安装与部署配置

## 2.1 Window下安装



## 2.2 Nginx启动

启动方式：

1. 直接双击nginx.exe
2. 进入命令行，执行start nginx

基本操作指令：

启动服务：start nginx

退出服务：nginx -s quit

强制关闭服务：nginx -s stop

重载服务：nginx -s reload

使用帮助：nginx -h

## 2.3 Linux下安装

Nginx依赖包：

ssl功能需要openssl库

gzip模块需要zlib库

rewrite模块需要pcre库

依赖包的安装顺序：openssl、zlib、pcre，最后安装nginx

安装命令：yum -y install gcc pcre-devel zlib-devel openssl openssl-devel

(1)下载nginx:[https://nginx.org/download/](https://nginx.org/download/" \t "https://www.cnblogs.com/xxoome/p/_blank) nginx-1.9.9.tar.gz

(2)解压nginx:tar -zxvf nginx-1.9.9.tar.gz

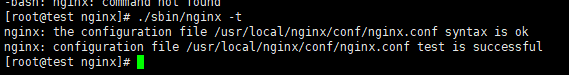
(3)进入nginx目录:cd nginx-1.9.9

(4)配置./configure --prefix=/usr/local/nginx  
(5)编译

make

make install

（6）检查nginx是否安装成功：切换到/usr/local/nginx下，执行./sbin/nginx -t



1. 启动nginx:在/usr/local/nginx/sbin目录下，执行./nginx启动nginx服务
2. 查看nginx启动是否成功，可以通过ps -ef |grep nginx查看nginx进程是否存在
3. 若nginx启动成功，页面访问不到，那需要vi /etc/sysconfig/iptables加入-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT，再重启防火墙service iptables restart

## 2.4 Nginx启动、停止

启动命令：/.nginx

停止：./nginx -s quit

强制停止：/.nginx -s stop

重启：./nginx -s reload

## 2.5 Nginx配置文件说明

Nginx服务器的配置信息主要集中在nginx.conf配置文件中

### 2.5.1 Main模块

|  |
| --- |
| #指定nginx worker进程运行用户以及用户组，默认nobody账号运行  #user nobody;  #指定nginx要开启的子进程数量  worker\_processes 1;  #定义错误日志文件的位置及级别  #error\_log logs/error.log;  #error\_log logs/error.log notice;  #error\_log logs/error.log info;  #用来指定进程id的存储文件位置  #pid logs/nginx.pid; |

### 2.5.2 Event模块

|  |
| --- |
| events {  #指定最大可以同时接收的连接数量  worker\_connections 1024;  } |

**http模块**

|  |
| --- |
| http {  include mime.types;  default\_type application/octet-stream;  #log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '  # '$status $body\_bytes\_sent "$http\_referer" '  # '"$http\_user\_agent" "$http\_x\_forwarded\_for"';  #access\_log logs/access.log main;  #配置on让sendfile发挥作用，将文件的回写过程交给缓冲去完成，而不是放在应用中  sendfile on;  #让nginx在一个数据包中发送所有的头文件，而不是一个一个的单独发  #tcp\_nopush on;  #给客户端分配连接超时时间，服务器会在这个时间过后关闭连接  #keepalive\_timeout 0;  keepalive\_timeout 65;    #gzip on; |

### 2.5.3 Server模块

|  |
| --- |
| #server:一个虚拟主机的配置，一个http可以配置多个server  server {  #监听的默认端口号：80  listen 80;  #指定ip地址或域名，多个配置之间用空格分隔  server\_name localhost;  #用于设置www/路径中配置的网页的默认编码格式  #charset koi8-r;  #指定虚拟机主服务器中的访问记录日志存放路径  #access\_log logs/host.access.log main; |

### 2.5.4 Location模块

Location模块是Nginx配置中出现最多的一个配置，主要用于配置路由访问信息

1.基本配置

|  |
| --- |
| #location /表示匹配根目录  location / {  #指定访问根目录时，访问虚拟主机的web目录  root html;  #不指定访问具体资源时，默认展示的资源文件列表  index index.html index.htm;  } |

2.反向代理配置

|  |
| --- |
| location / {  proxy\_pass http://localhost:8888;  proxy\_set\_header X-real-ip $remote\_addr;  proxy\_set\_header Host $http\_host;  } |

1. uwsgi配置

|  |
| --- |
| location / {  include uwsgi\_params;  uwsgi\_pass localhost:8888;  } |

### 2.5.5 负载均衡模块upstream

|  |
| --- |
| Upstream name {  ip\_hash;  #server host:port分发服务器的列表配置  #down-表示主机暂停服务  #max\_fails表示失败最大次数，超过该次数暂停服务  #fail\_timeout表示请求受理失败，暂停指定时间之后重新发起请求  server 192.168.1.100:8000 weight=9;  server 192.168.1.100:8001 down;  server 192.168.1.100:8002 max\_fails=3;  server 192.168.1.100:8003 fail\_timeout=20s;  server 192.168.1.100:8004 max\_fails=3 fail\_timeout=20s;  } |

## 2.6 Nginx的主要配置

### 2.6.1 静态http服务器配置

|  |
| --- |
| server {  listen 80;  server\_name localhost 192.168.1.100;  location / {  #虚拟目录  root /usr/share/nginx/www;  }  } |

### 2.6.2 反向代理服务器配置

|  |
| --- |
| server {  listen 80;  location / {  #应用服务的http地址  proxy\_pass http://192.168.1.112:8080;  }  } |

### 2.6.3 负载均衡配置

|  |
| --- |
| Upstream myapp{  ip\_hash;  server 192.168.1.111:8080 weight=9;#应用服务器1  server 192.168.1.112:8080 weight=1;#应用服务器2  } |

### 2.6.4怎么实现反向代理并负载均衡

|  |
| --- |
| upstream myapp {  #三台tomcat服务器  server 192.168.197.136:8091;  server 192.168.197.136:8092;  server 192.168.197.136:8093;  }    server {  listen 80;  server\_name 192.168.197.136;  location / {  proxy\_pass http://myapp;  }  } |

注意：

如果将server\_name的域名改为www.aa.com

如下：server\_name [www.aa.com;](http://www.aa.com;)

1.修改nginx.conf的配置文件，重启nginx

2.需要通过vim /etc/hosts 增加192.168.197.136 [www.aa.com](http://www.aa.com)

3.在window/system32/driver/etc/hosts中同样也增加192.168.197.136 [www.aa.com](http://www.aa.com)

### 2.6.5权重weight

|  |
| --- |
| upstream myapp {  server 192.168.197.136:8091 weight=6;  server 192.168.197.136:8092 weight=3;  server 192.168.197.136:8093 weight=1;  } |

weight代表权重，访问10次tomcat服务器，其中有6次访问8091，3次访问8092，1次访问8093

### 2.6.6 IP\_HASH机制

|  |
| --- |
| upstream myapp {  ip\_hash;  server 192.168.197.136:8091;  server 192.168.197.136:8092;  server 192.168.197.136:8093;  } |

使用Nginx中的IP\_HASH技术，能够根据用户的ip动态绑定到一台服务器中，变向的实现Session共享，IP\_HASH中优先级最高，配置后轮询和权重不生效

缺点：

　　　1.如果服务器宕机，用户访问受限

　　　2.使用IP\_HASH导致负载不均衡

　　　3.安全性不好

## 2.7 Nginx故障迁移

1.手动下线(down)

server 192.168.197.136:8091 down;

1. 备用机机制(backup)

所有的机子down了才生效

server 192.168.197.136:8091 backup;

1. 设定超时时间

说明：max\_fails表示失败次数，fall\_timeout表示在规定时间内不再将请求发往服务器，时间过后再次检测

proxy\_connect\_timeout 3;

　　　proxy\_read\_timeout 3;

　　　proxy\_send\_timeout 3;

1. 健康检测

server 192.168.197.136:8091 max\_fails=1 fail\_timeout=60s;

server 192.168.197.136:8092 max\_fails=1 fail\_timeout=60s;

server 192.168.197.136:8093 max\_fails=1 fail\_timeout=60s;