Nginx

一、 Nginx 介绍

1 Nginx 简介

Nginx (engine x) 是一个高性能的 HTTP 和反向代理服务。Nginx 是由伊戈尔·赛索耶夫为俄罗斯访问量第二的 Rambler.ru 站点（俄文：Рамблер）开发的，第一个公开版本 0.1.0 发布于 2004 年 10 月 4 日。

Nginx 是一个很强大的高性能 Web 和反向代理服务，它具有很多非常优越的特性：在连接高并发的情况下，Nginx 是 Apache 服务不错的替代品：Nginx 在美国是做虚拟主机生意的老板们经常选择的软件平台之一。

2 Nginx 作用

2.1 http 协议代理

2.2 搭建虚拟主机

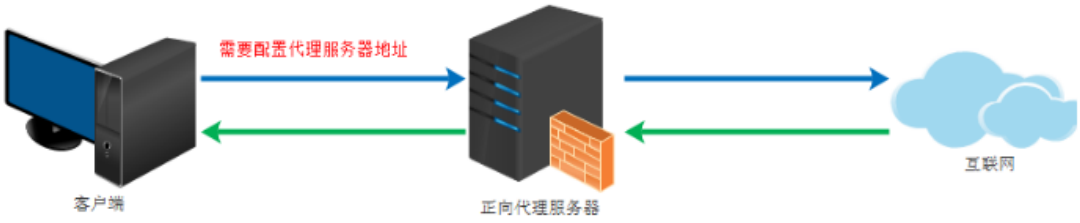
2.3 服务的反向代理

2.4 在反向代理中配置集群的负载均衡

二、 代理方式

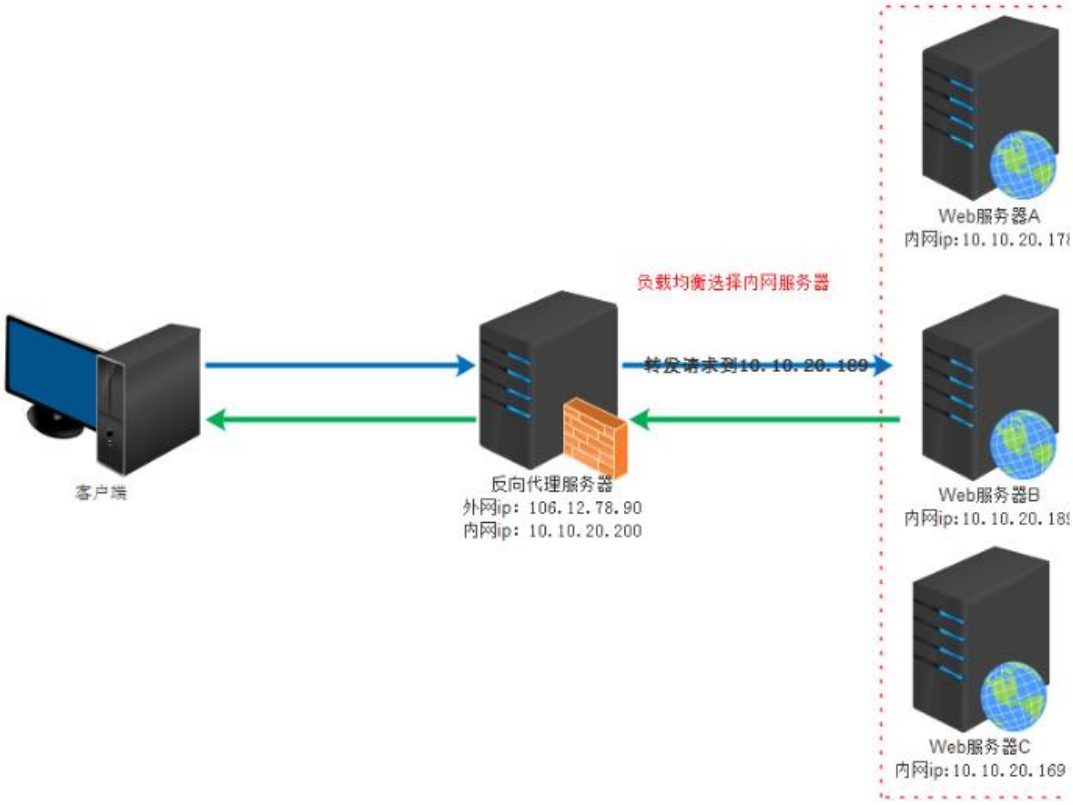
1 正向代理

正向代理，意思是一个位于客户端和原始服务器(origin server)之间的服务器，为了从原始服务器取得内容，客户端向代理发送一个请求并指定目标(原始服务器)，然后代理向原始服务器转交请求并将获得的内容返回给客户端。客户端才能使用正向代理。



2 反向代理

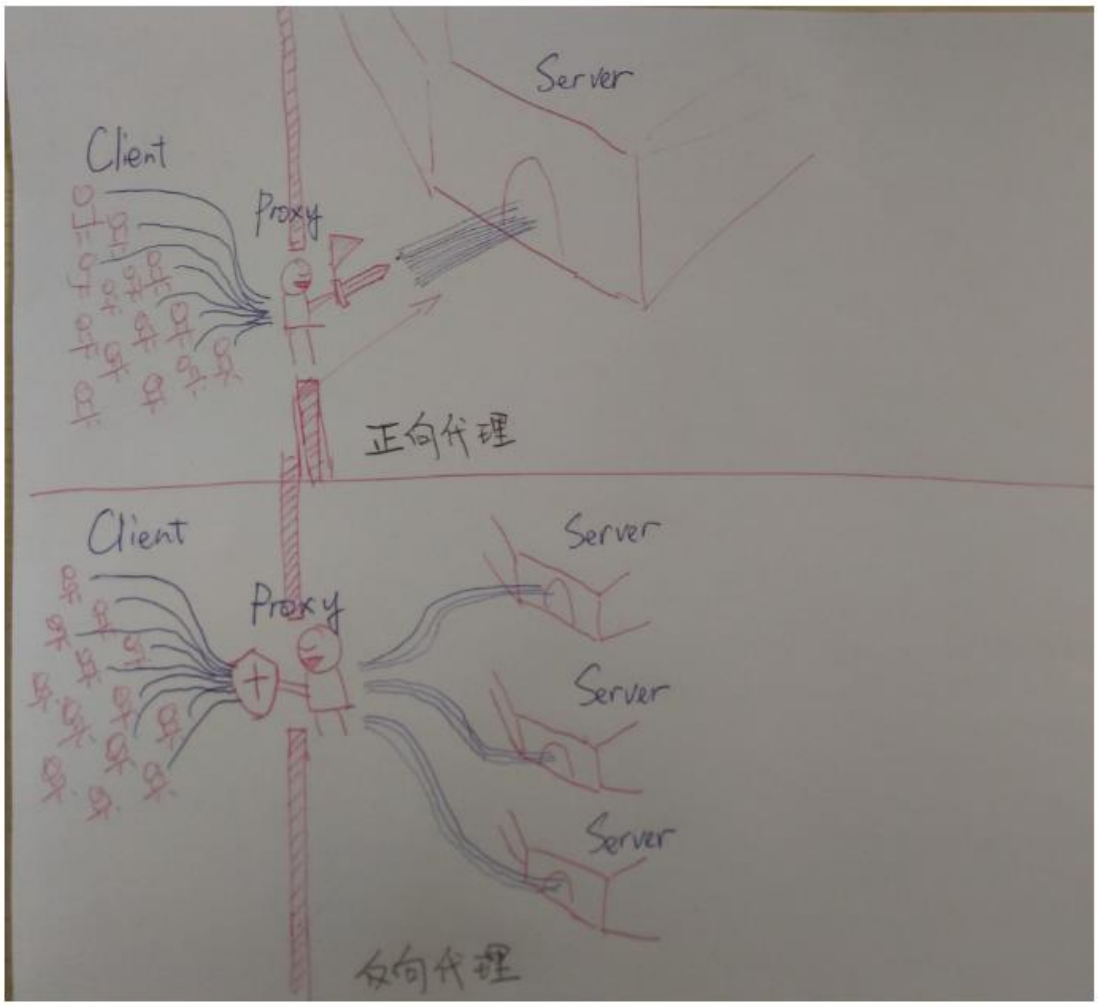
反向代理（Reverse Proxy）方式是指以代理服务器来接受 internet 上的连接请求，然后将请求转发给内部网络上的服务器，并将从服务器上得到的结果返回给 internet 上请求连接的客户端，此时代理服务器对外就表现为一个反向代理服务器。



3 二者之间的区别

位置不同
正向代理，架设在客户机和目标主机之间；
反向代理，架设在服务器端；

代理对象不同
正向代理，代理客户端，服务端不知道实际发起请求的客户端；
反向代理，代理服务端，客户端不知道实际提供服务的服务端；



三、 安装 Nginx

1 将 Nginx 安装包上传到 Linux 中

使用的 Nginx 版本为 nginx-1.8.0.tar.gz

2 nginx 安装环境

nginx 是 C 语言开发，建议在 linux 上运行，本视频使用 Centos6.5 作为安装环境。

- gcc

安装 nginx 需要先将官网下载的源码进行编译，编译依赖 gcc 环境，如果没有 gcc 环境，需要安装 gcc：yum install gcc-c++

- PCRE

PCRE(Perl Compatible Regular Expressions)是一个 Perl 库，包括 perl 兼容的正则表达式库。nginx 的 http 模块使用 pcre 来解析正则表达式，所以需要在 linux 上安装 pcre 库。

yum install -y pcre pcre-devel

注：pcre-devel 是使用 pcre 开发的一个二次开发库。nginx 也需要此库。

- zlib

zlib库提供了很多种压缩和解压缩的方式，nginx使用zlib对http包的内容进行gzip，所以需要在linux上安装zlib库。

yum install -y zlib zlib-devel

- openssl

OpenSSL 是一个强大的安全套接字层密码库，囊括主要的密码算法、常用的密钥和证书封装管理功能及SSL协议，并提供丰富的应用程序供测试或其它目的使用。

nginx不仅支持http协议，还支持https（即在ssl协议上传输http），所以需要在linux安装openssl库。

yum install -y openssl openssl-devel

3 编译安装

解压：tar -zxvf nginx-1.8.0.tar.gz

进入到nginx的根目录
cd nginx-1.8.0

3.1配置安装参数

./configure

参数设置如下：
./configure \
--prefix=/usr/local/nginx \
--pid-path=/var/run/nginx/nginx.pid \
--lock-path=/var/lock/nginx.lock \
--error-log-path=/var/log/nginx/error.log \
--http-log-path=/var/log/nginx/access.log \
--with-http_gzip_static_module \
--http-client-body-temp-path=/var/temp/nginx/client \
--http-proxy-temp-path=/var/temp/nginx/proxy \
--http-fastcgi-temp-path=/var/temp/nginx/fastcgi \
--http-uwsgi-temp-path=/var/temp/nginx/uwsgi \
--http-scgi-temp-path=/var/temp/nginx/scgi

注意：上边将临时文件目录指定为/var/temp/nginx，需要在/var下创建temp及nginx目录

3.2编译安装

编译：make

编译安装 make　install

四、　操作 Nginx

1　启动 nginx

cd /usr/local/nginx/sbin/
./nginx
注意：执行./nginx 启动 nginx，这里可以-c 指定加载的 nginx 配置文件，如下：
./nginx -c /usr/local/nginx/conf/nginx.conf
如果不指定-c，nginx 在启动时默认加载 conf/nginx.conf 文件，此文件的地址也可以在编译安装 nginx 时指定./configure 的参数（--conf-path= 指向配置文件（nginx.conf））

2　停止 nginx

方式 1，快速停止：
cd /usr/local/nginx/sbin
./nginx -s stop
此方式相当于先查出 nginx 进程 id 再使用 kill 命令强制杀掉进程。

方式 2，完整停止(建议使用)：
cd /usr/local/nginx/sbin
./nginx -s quit
此方式停止步骤是待 nginx 进程处理任务完毕进行停止。

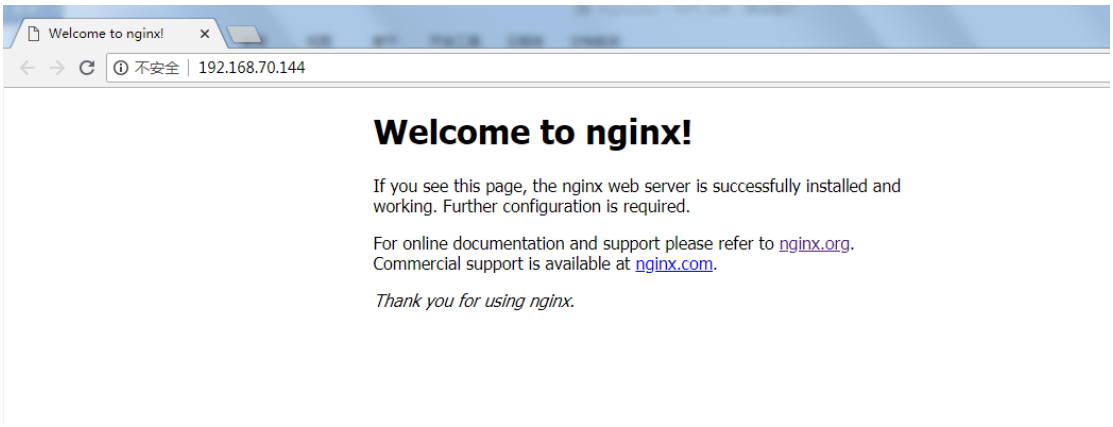
3　重启 nginx

方式 1，先停止再启动（建议使用）：
对 nginx 进行重启相当于先停止 nginx 再启动 nginx，即先执行停止命令再执行启动命令。
如下：
./nginx -s quit
./nginx

方式 2，重新加载配置文件：
当 nginx 的配置文件 nginx.conf 修改后，要想让配置生效需要重启 nginx，使用-s reload 不用先停止 nginx 再启动 nginx 即可将配置信息在 nginx 中生效，如下：
./nginx -s reload

4　测试

nginx 安装成功，启动 nginx，即可访问虚拟机上的 nginx
Nginx 默认的是侦听 80 端口



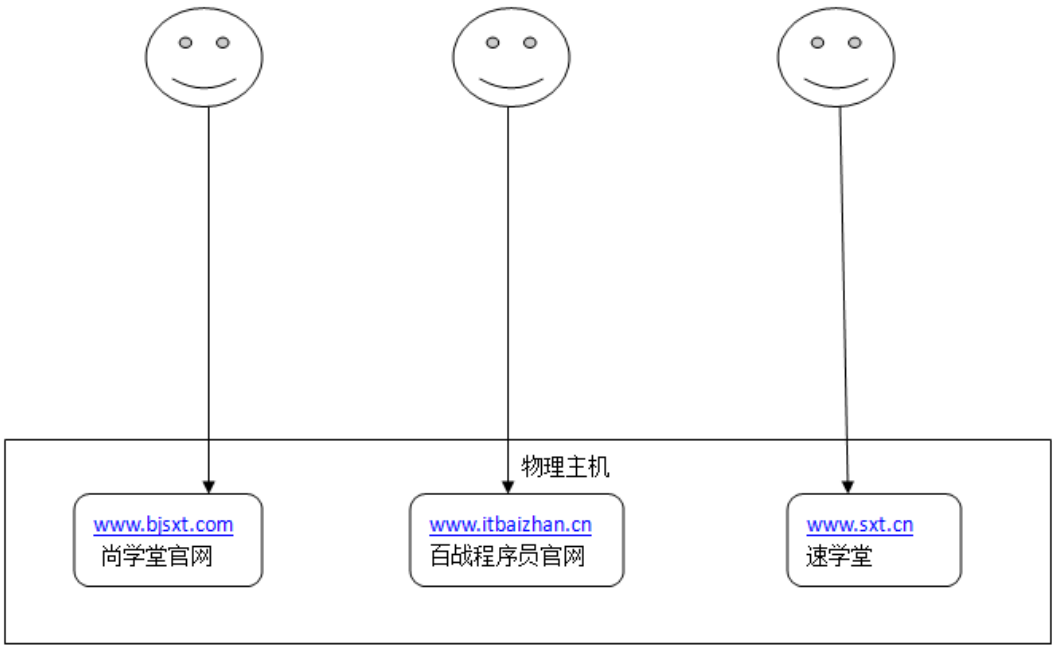
五、 Nginx 的使用

1 配置虚拟主机

1.1 虚拟主机介绍

虚拟主机是一种特殊的软硬件技术，它可以将网络上的每一台计算机分成多个虚拟主机，每个虚拟主机可以独立对外提供 www 服务，这样就可以实现一台主机对外提供多个 web 服务，每个虚拟主机之间是独立的，互不影响的。

虚拟主机技术是互联网服务器采用的节省服务器硬件成本的技术，虚拟主机技术主要应用于 HTTP（Hypertext Transfer Protocol，超文本传输协议）服务，将一台服务器的某项或者全部服务内容逻辑划分为多个服务单位，对外表现为多个服务器，从而充分利用服务器硬件资源。



1.2 Nginx 的虚拟主机配置方式

Nginx 支持三种类型的虚拟主机配置

1. 基于 IP 的虚拟主机
2. 基于端口的虚拟主机
3. 基于域名的虚拟主机

1.2.1基于 IP 的虚拟主机配置方式

1.2.1.1 需求

一台 Linux 服务器绑定两个 ip:192.168.70.144、192.168.70.188
访问不同的 ip 请求不同的 html 目录，即：
访问 http://192.168.70.144 将访问“html144”目录下的 html 网页
访问 http://192.168.70.188 将访问“html188”目录下的 html 网页

1.2.1.2 创建 HTML 目录

```
drwxr-xr-x. 2 root root 4096 Sep 18 01:49 conf
drwxr-xr-x. 2 root root 4096 Sep 18 20:38 html
drwxr-xr-x. 2 root root 4096 Sep 18 01:50 sbin
[root@localhost nginx]# cp html/ html144 -r
[root@localhost nginx]# cp html/ html188 -r
[root@localhost nginx]#
```

1.2.1.3 Linux 绑定多 IP

Linux 操作系统允许绑定多 IP。使用 IP 别名的方式，在一块物理网卡上可以绑定多个 lP 地址。这样就能够在使用单一网卡的同一个服务器上运行多个基于 IP 的虚拟主机。但是在绑定多 IP 时需要将动态的 IP 分配方式修改为静态的指定 IP

1.2.1.3.1将动态 IP 修改为静态 IP

cd /etc/sysconfig/network-scripts

```
IPADDR=192.168.10.144
NETMASK=255.255.255.0
GATEWAY=192.168.10.2
DNS1=114.114.114.114
```

1.2.1.3.2IP 绑定

将/etc/sysconfig/network-scripts/ifcfg-eth0 文件复制一份，命名为 ifcfg-eth0:1
修改其中内容：
DEVICE=eth0:1
IPADDR=192.168.70.188

其他项不用修改
重启系统

1.2.1.4 修改 Nginx 的配置文件完成基于 IP 的虚拟主机配置

Nginx 的配置文件 nginx.conf

如上述配置文件所示，主要由 6 个部分组成：

main：用于进行 nginx 全局信息的配置
events：用于 nginx 工作模式的配置
http：用于进行 http 协议信息的一些配置
server：用于进行服务器访问信息的配置
location：用于进行访问路由的配置
upstream：用于进行负载均衡的配置

Nginx.conf

```
user    root;
worker_processes    1;

#error_log    logs/error.log;
#error_log    logs/error.log    notice;
#error_log    logs/error.log    info;

#pid              logs/nginx.pid;


events {
    worker_connections    1024;
}


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;

    sendfile            on;
    #tcp_nopush         on;
```

```
#keepalive_timeout    0;
keepalive_timeout    65;

#gzip    on;

#一个 Server 就是一个虚拟主机
server {
        listen              80;
#为虚拟机指定 IP 或者是域名
        server_name    192.168.70.144;

#主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root      html144;
        #在不指定访问具体资源时，默认的展示资源的列表
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
        location = /50x.html {
            root      html;
        }

}

#一个 Server 就是一个虚拟主机
server {
        listen              80;
#为虚拟机指定 IP 或者是域名
        server_name    192.168.70.188;

#主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root      html188;
        #在不指定访问具体资源时，默认的展示资源的列表
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
```

```
            location = /50x.html {
                root   html;
            }

        }

}
```

1.2.2基于端口的虚拟主机配置方式

1.2.2.1 需求

Nginx 对提供 8080 与 9090 两个端口的监听服务
请求 8080 端口则访问 html8080 目录下的 index.html
请求 9090 端口则访问 html9090 目录下的 index.html

1.2.2.2 创建 HTML 目录

```
drwxr-xr-x. 2 root root 4096 Sep 18 23:04 conf
drwxr-xr-x. 2 root root 4096 Sep 18 22:52 html
drwxr-xr-x. 2 root root 4096 Sep 18 22:54 html144
drwxr-xr-x. 2 root root 4096 Sep 18 22:54 html188
drwxr-xr-x. 2 root root 4096 Sep 18 22:52 sbin
[root@localhost nginx]# cp html html8080 -r
[root@localhost nginx]# cp html html9090 -r
```

1.2.2.3 修改 Nginx 的配置文件完成基于端口的虚拟主机配置

Nginx.conf

```
user    root;
worker_processes    1;

#error_log    logs/error.log;
#error_log    logs/error.log    notice;
#error_log    logs/error.log    info;

#pid              logs/nginx.pid;


events {
    worker_connections    1024;
}
```

```
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;

    sendfile        on;
    #tcp_nopush     on;

    #keepalive_timeout  0;
    keepalive_timeout  65;

    #gzip  on;

    #一个 Server 就是一个虚拟主机
    server {
        listen       80;
    #为虚拟机指定 IP 或者是域名
        server_name  192.168.70.144;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root   html144;
        #在不指定访问具体资源时，默认的展示资源的列表
            index  index.html index.htm;
        }


        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }

    }

    #一个 Server 就是一个虚拟主机
    server {
```

```
        listen          80;
    #为虚拟机指定 IP 或者是域名
        server_name    192.168.70.188;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root     html188;
        #在不指定访问具体资源时，默认的展示资源的列表
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
        location = /50x.html {
            root     html;
        }

    }

     #一个 Server 就是一个虚拟主机  基于端口
    server {
        listen          8080;
    #为虚拟机指定 IP 或者是域名
        server_name    192.168.70.188;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root     html8080;
        #在不指定访问具体资源时，默认的展示资源的列表
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
        location = /50x.html {
            root     html;
        }

    }

     #一个 Server 就是一个虚拟主机
    server {
```

```
        listen          9090;
    #为虚拟机指定 IP 或者是域名
        server_name    192.168.70.188;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root     html9090;
        #在不指定访问具体资源时，默认的展示资源的列表
            index    index.html index.htm;
        }


        error_page     500 502 503 504    /50x.html;
        location = /50x.html {
            root     html;
        }

    }

}
```

1.2.3基于域名的虚拟主机配置方式

1.2.3.1 需求

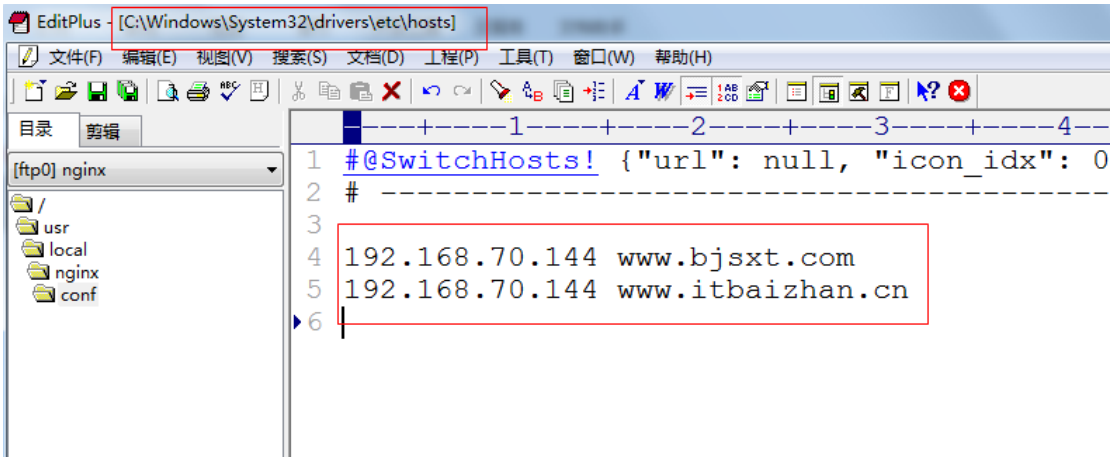
两个域名指向同一个 nginx 服务器，用户访问不同的域名时显示不同的内容。
域名规划：
1, www.bjsxt.com
2, www.itbaizhan.cn

1.2.3.2 创建 HTML 目录

```
drwxr-xr-x. 2 root root 4096 Sep 18 23:04 conf
drwxr-xr-x. 2 root root 4096 Sep 18 22:52 html
drwxr-xr-x. 2 root root 4096 Sep 18 22:54 html144
drwxr-xr-x. 2 root root 4096 Sep 18 22:54 html188
drwxr-xr-x. 2 root root 4096 Sep 18 23:25 html8080
drwxr-xr-x. 2 root root 4096 Sep 18 23:25 html9090
drwxr-xr-x. 2 root root 4096 Sep 18 22:52 sbin
[root@localhost nginx]# cp html html-bjsxt -r
[root@localhost nginx]# cp html html-itbaizhan -r
[root@localhost nginx]# ll
total 36
drwxr-xr-x. 2 root root 4096 Sep 18 23:04 conf
drwxr-xr-x. 2 root root 4096 Sep 18 22:52 html
drwxr-xr-x. 2 root root 4096 Sep 18 22:54 html144
drwxr-xr-x. 2 root root 4096 Sep 18 22:54 html188
drwxr-xr-x. 2 root root 4096 Sep 18 23:25 html8080
drwxr-xr-x. 2 root root 4096 Sep 18 23:25 html9090
drwxr-xr-x. 2 root root 4096 Sep 19 00:06 html-bjsxt
drwxr-xr-x. 2 root root 4096 Sep 19 00:06 html-itbaizhan
drwxr-xr-x. 2 root root 4096 Sep 18 22:52 sbin
```

1.2.3.3 修改 windows 的 hosts 文件配置域名与 ip 的映射

文件路径：C:\Windows\System32\drivers\etc



1.2.3.4 修改 Nginx 的配置文件完成基于域名的虚拟主机配置

nginx.conf

```
server {
        listen          80;
    #为虚拟机指定 IP 或者是域名
        server_name    test.bjsxt.com;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root     html-bjsxt;
        #在不指定访问具体资源时，默认的展示资源的列表
```

```
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
        location = /50x.html {
            root      html;
        }

    }

     #一个 Server 就是一个虚拟主机
    server {
        listen          80;
    #为虚拟机指定 IP 或者是域名
        server_name    test.itbaizhan.cn;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            root      html-itbaizhan;
        #在不指定访问具体资源时，默认的展示资源的列表
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
        location = /50x.html {
            root      html;
        }

    }
```

2 Nginx 中配置服务的反向代理

2.1 需求

安装两个 tomcat 服务，通过 nginx 反向代理。
本案例中使用两台虚拟机演示。
tomcat 安装到 192.168.70.143 环境中。端口为 8080 与 9090
Nginx 安装在 192.168.70.144 环境中

2.2 安装环境

2.3 安装 tomcat

```
[root@localhost temp]# cp apache-tomcat-7.0.47 /usr/local/n-tomcat1 -r
[root@localhost temp]# cp apache-tomcat-7.0.47 /usr/local/n-tomcat2 -r
```

2.4 配置 tomcat

2.4.1 修改端口

```
-rw-------. 1 root root  12321 Sep 19 00:39 catalina.policy
-rw-------. 1 root root   6029 Sep 19 00:39 catalina.properties
-rw-------. 1 root root   1394 Sep 19 00:39 context.xml
-rw-------. 1 root root   3288 Sep 19 00:39 logging.properties
-rw-------. 1 root root   6435 Sep 19 00:39 server.xml
-rw-------. 1 root root   1530 Sep 19 00:39 tomcat-users.xml
-rw-------. 1 root root 162905 Sep 19 00:39 web.xml
[root@localhost conf]# vim server.xml
```

2.4.2 修改首页内容

```
[root@localhost webapps]# cd ROOT/
[root@localhost ROOT]# ll
total 200
-rw-r--r--. 1 root root 17811 Sep 19 00:39 asf-logo.png
-rw-r--r--. 1 root root  5866 Sep 19 00:39 asf-logo-wide.gif
-rw-r--r--. 1 root root   713 Sep 19 00:39 bg-button.png
-rw-r--r--. 1 root root  1918 Sep 19 00:39 bg-middle.png
-rw-r--r--. 1 root root  1392 Sep 19 00:39 bg-nav-item.png
-rw-r--r--. 1 root root  1401 Sep 19 00:39 bg-nav.png
-rw-r--r--. 1 root root  3103 Sep 19 00:39 bg-upper.png
-rw-r--r--. 1 root root  3376 Sep 19 00:39 build.xml
-rw-r--r--. 1 root root 21630 Sep 19 00:39 favicon.ico
-rw-r--r--. 1 root root 12308 Sep 19 00:39 index.jsp
-rw-r--r--. 1 root root  8826 Sep 19 00:39 RELEASE-NOTES.txt
-rw-r--r--. 1 root root  5576 Sep 19 00:39 tomcat.css
-rw-r--r--. 1 root root  2066 Sep 19 00:39 tomcat.gif
-rw-r--r--. 1 root root  5103 Sep 19 00:39 tomcat.png
-rw-r--r--. 1 root root  2376 Sep 19 00:39 tomcat-power.gif
-rw-r--r--. 1 root root 67198 Sep 19 00:39 tomcat.svg
drwxr-xr-x. 2 root root  4096 Sep 19 00:39 WEB-INF
[root@localhost ROOT]# vim index.jsp
```

2.5 配置 Nginx 实现服务的反向代理

nginx.conf

```
user    root;
worker_processes    1;

#error_log    logs/error.log;
#error_log    logs/error.log    notice;
#error_log    logs/error.log    info;

#pid              logs/nginx.pid;


events {
      worker_connections    1024;
}


http {
      include             mime.types;
      default_type    application/octet-stream;


      sendfile              on;
      keepalive_timeout    65;



      upstream tomcat_server1{
      server 192.168.70.143:8080;
      }

      upstream tomcat_server2{
      server 192.168.70.143:9090;
      }

      server {
            listen             80;
      #为虚拟机指定 IP 或者是域名
            server_name    test.bjsxt.com;

      #主要配置路由访问信息
            location / {
            #用于指定访问根目录时，访问虚拟主机的 web 目录
                  proxy_pass    http://tomcat_server1;

            #在不指定访问具体资源时，默认的展示资源的列表
```

```
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
        location = /50x.html {
            root      html;
        }

    }

     #一个 Server 就是一个虚拟主机
    server {
        listen            80;
    #为虚拟机指定 IP 或者是域名
        server_name    test.itbaizhan.cn;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            proxy_pass    http://tomcat_server2;
        #在不指定访问具体资源时，默认的展示资源的列表
            index    index.html index.htm;
        }


        error_page      500 502 503 504    /50x.html;
        location = /50x.html {
            root      html;
        }

    }

}
```

2.6在反向代理中配置负载均衡

2.6.1什么是负载均衡

负载均衡建立在现有网络结构之上，它提供了一种廉价有效透明的方法扩展网络设备和服务器的带宽、增加吞吐量、加强网络数据处理能力、提高网络的灵

活性和可用性。

负载均衡，英文名称为 Load Balance，其意思就是分摊到多个操作单元上进行执行，例如 Web 服务器、FTP 服务器、企业关键应用服务器和其它关键任务服务器等，从而共同完成工作任务。

2.6.2Nginx 负载均衡策略

2.6.2.1 轮询（默认）

每个请求按时间顺序逐一分配到不同的后端服务器，如果后端服务器 down 掉，能自动剔除。

2.6.2.2 指定权重

指定轮询几率，weight 和访问比率成正比，用于后端服务器性能不均的情况。

```
upstream backserver {
server 192.168.0.14 weight=10;
server 192.168.0.15 weight=10;
}
```

2.6.2.3 IP 绑定 ip_hash

每个请求按访问 ip 的 hash 结果分配，这样每个访客固定访问一个后端服务器，可以解决 session 的问题。

```
upstream backserver {
ip_hash;
server 192.168.0.14:88;
server 192.168.0.15:80;
}
```

2.6.3需求

nginx 作为负载均衡服务器，用户请求先到达 nginx，再由 nginx 根据负载配置将请求转发至 tomcat 服务器。

nginx 负载均衡服务器：192.168.70.144

tomcat1 服务器：192.168.70.143:8080

tomcat2 服务器：192.168.70.143:9090

2.6.4Nginx 的集群配置

节点说明：
在 http 节点里添加:

#定义负载均衡设备的 Ip 及设备状态
upstream myServer {

server 127.0.0.1:9090 down;
server 127.0.0.1:8080 weight=2;
server 127.0.0.1:6060;
server 127.0.0.1:7070 backup;
}

在需要使用负载的 Server 节点下添加

proxy_pass http://myServer;

upstream 每个设备的状态:

down 表示单前的 server 暂时不参与负载
weight 默认为 1.weight 越大，负载的权重就越大
fail_timeout:次失败后，暂停的时间 默认 10s
max_fails ：允许请求失败的次数默认为 1.当超过最大次数时，返回
backup： 其它所有的非 backup 机器 down 或者忙的时候，请求 backup 机器。所以这台机器压力会最轻。

nginx.conf

```
user  root;
worker_processes  1;

#error_log  logs/error.log;
#error_log  logs/error.log  notice;
#error_log  logs/error.log  info;

#pid        logs/nginx.pid;


events {
    worker_connections  1024;
```

```
}


http {
    include       mime.types;
    default_type  application/octet-stream;


    sendfile        on;
    keepalive_timeout  65;



    upstream tomcat_server1{
    server 192.168.70.143:8080 weight=10;
    server 192.168.70.143:9090 weight=2;
    }


    server {
        listen       80;
    #为虚拟机指定 IP 或者是域名
        server_name  test.bjsxt.com;

    #主要配置路由访问信息
        location / {
        #用于指定访问根目录时，访问虚拟主机的 web 目录
            proxy_pass   http://tomcat_server1;

        #在不指定访问具体资源时，默认的展示资源的列表
            index  index.html index.htm;
        }


        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }

    }


}
```

3 http 协议代理

由于 ftp 服务器是基于 ftp 协议处理的。那么现在我想在外部访问该图片，是没有办法访问的。他是不能处理 http 协议的。所以我们需要拥有一个能够处理 http 协议代理服务器。其实就是使用了 Nginx 的虚拟主机的方式。

3.1 需求

使用 Nginx 的基于域名的虚拟主机的方式来完成在 KindEditor 中添加图片时的图片回显处理。

nginx：192.168.70.144

VSFTPD：192.168.70.144

注意：nginx 与 VSFTPD 必须安装到同一个环境中。

3.2 解决 KindEditorDemo 项目中图片回显的问题

3.2.1 修改系统的 hosts 文件

```
192.168.70.144 img.bjsxt.com
```

3.2.2 修改项目的 resource.properties 文件

```
FTP_HOST=192.168.70.144
FTP_PORT=21
FTP_USERNAME=ftpuser
FTP_PASSWORD=ftpuser
FTP_BASEPATH=/home/ftpuser/
HTTP_BASE_PATH=http://img.bjsxt.com
```

3.2.3 修改 nginx.conf

```
user    root;
worker_processes    1;
```

```
    #error_log  logs/error.log;
    #error_log  logs/error.log  notice;
    #error_log  logs/error.log  info;

    #pid        logs/nginx.pid;


    events {
        worker_connections  1024;
    }


    http {
        include       mime.types;
        default_type  application/octet-stream;


        sendfile        on;
        keepalive_timeout  65;



        server {
            listen       80;
        #为虚拟机指定 IP 或者是域名
            server_name  img.bjsxt.com;

        #主要配置路由访问信息
            location / {
            #用于指定访问根目录时，访问虚拟主机的 web 目录
                root   /home/ftpuser/;

            #在不指定访问具体资源时，默认的展示资源的列表
                index  index.html index.htm;
            }


            error_page   500 502 503 504  /50x.html;
            location = /50x.html {
                root   html;
            }

        }
```

```
}
```

3.2.4测试