NSD OPERATION DAY01

1 案例1：反向代理

1.1 问题

通过配置代理服务器，实现以下目标：

1. 代理服务器可以将远程的Web服务器页面缓存在本地
2. 代理服务器端口设置为80端口
3. 用户通过访问代理服务器即可获得远程Web服务器上的页面内容
4. 远程Web服务器对客户端用户是透明的
5. 利用缓存机制提高网站的响应速度

1.2 方案

使用3台RHEL7虚拟机，其中一台作为Squid代理服务器，该服务器用来连接两个网段，因此需要配置两块网卡，地址分别为192.168.4.5和192.168.2.5。一台作为客户端测试主机，IP地址为192.168.4.100。一台Web服务器，地址为192.168.2.100，该Web服务器为其他代理提供Web数据源。

实验环境所需要的主机及对应的IP设置列表如表-1所示。

表 - 1 主机列表

主机	IP 地址
Client	eth0(192.168.4.100)
Proxy	eth0(192.168.4.5) eth1(192.168.2.5)
Web1	eth1(192.168.2.100)

Top

实验拓扑如图-1所示。

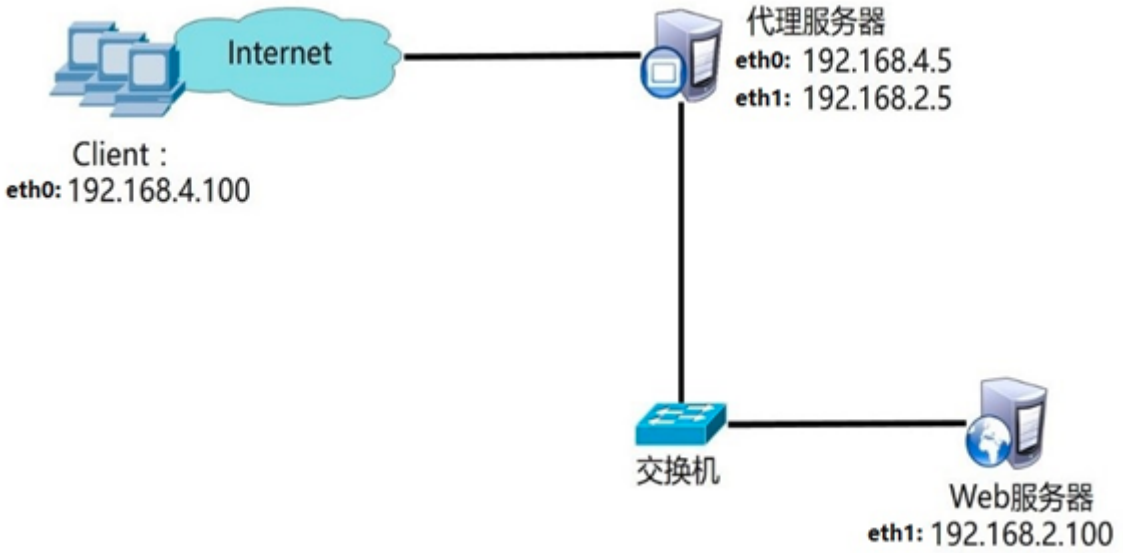

图-1

1.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：构建web服务器

1）使用yum安装web软件包

```
[root@web ~]# yum -y install httpd
.. ..
[root@web ~]# rpm -q httpd
httpd-2.4.6-40.el7.x86_64
```

Top

2）启用httpd服务，并设为开机自动运行

```
[root@web ~]# systemctl start httpd ; systemctl enable httpd
```

httpd服务默认通过TCP 80端口监听客户端请求：

```
[root@web ~]# netstat -anptu | grep httpd
tcp        0      0 :::80                   :::*                    LISTEN      2813/httpd
```

3）为Web访问建立测试文件

在网站根目录/var/www/html下创建一个名为index.html的首页文件：

```
[root@web ~]# cat /var/www/html/index.html
<html>
<title>Welcome</title>
<body>
<h1>192.168.2.100</h1>
</body>
</html>
```

步骤二：部署Squid代理服务器

1）使用yum安装squid软件包：

Top

```
[root@svr5 ~]# yum -y install squid
.. ..
```

2）修改/etc/squid/squid.conf配置文件：

```
[root@svr5 ~]# vim /etc/squid/squid.conf
.. ..
http_port 80 vhost                          //设置反向代理
visible_hostname svr5.tarena.com            //设置主机名，默认没有该语句
cache_peer 192.168.2.100 parent 80 0 originserver    //定义后端真实服务器信息
cache_dir ufs /var/spool/squid 200 16 256       //硬盘缓存，缓存容量为200M，自动创建16个一级子目录和256个二级子目录
http_access allow all                       //允许本机所有主机使用代理服务器
```

3）启动squid服务，并设置为开机启动：

```
[root@svr5 ~]# systemctl start squid ; systemctl enable squid
```

4）squid服务通过TCP 80端口监听客户端请求：

```
[root@svr5 ~]# netstat -anptu | grep 80
tcp        0      0 :::80                   :::*                    LISTEN      3213/(squid)
```

步骤三：客户端测试

2）客户端开启浏览器访问

Top

```
01.  [root@client ~]# curl  http://192.168.4.5          //返回的是2.100服务的页面
```

2 案例2：使用Varnish加速Web

2.1 问题

通过配置Varnish缓存服务器，实现如下目标：

- 使用Varnish加速后端Apache Web服务
- 使用varnishadm命令管理缓存页面
- 使用varnishstat命令查看Varnish状态

2.2 方案

通过源码编译安装Varnish缓存服务器

- 编译安装Varnish软件
- 复制启动脚本与配置文件

修改配置文件，缓存代理源Web服务器，实现Web加速功能

使用3台RHEL7虚拟机，其中一台作为Web服务器（192.168.2.100）、一台作为Varnish代理服务器（192.168.4.5,192.168.2.5)，另外一台作为测试用的Linux客户机（192.168.2.100），如图-2所示。

Top

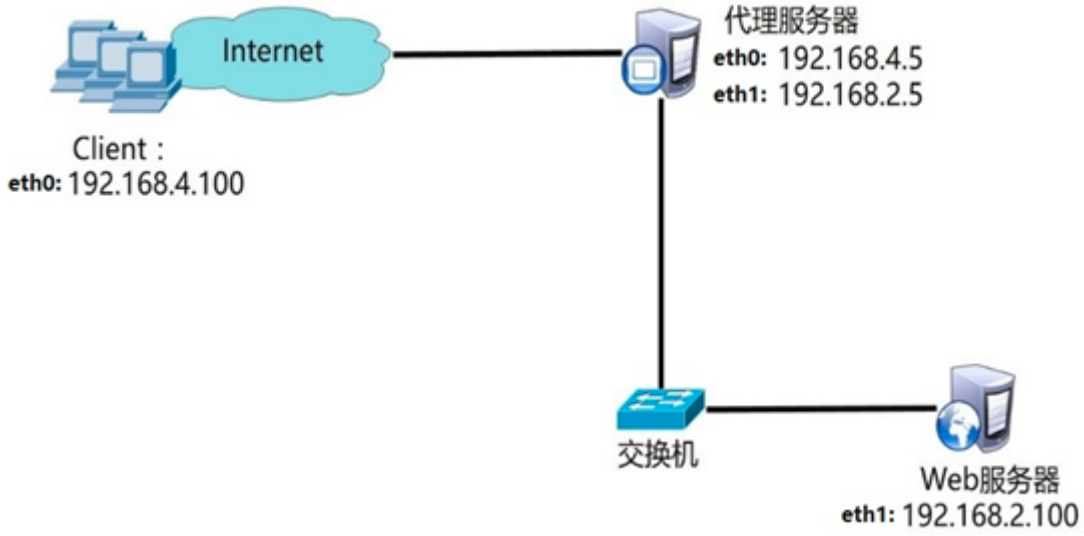

图-2

对于Web服务器的部署，此实验中仅需要安装httpd软件、启动服务，并生成测试首页文件即可，默认httpd网站根路径为/var/www/html，首页文档名称为index.html。

2.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：构建Web服务器

1）使用yum安装web软件包

```
[root@web1 ~]# yum -y install httpd
```

2）启用httpd服务，并设为开机自动运行

Top

```
[root@web1 ~]# systemctl start httpd ; systemctl enable httpd
```

httpd服务默认通过TCP 80端口监听客户端请求：

```
[root@pc205 ~]# netstat -anptu | grep httpd
tcp        0      0 :::80                   :::*                    LISTEN      2813/httpd
```

3）为Web访问建立测试文件

在网站根目录/var/www/html下创建一个名为index.html的首页文件：

```
[root@pc205 ~]# cat /var/www/html/index.html
<html>
<title>Welcome</title>
<body>
<h1>192.168.2.100</h1>
</body>
</html>
```

步骤二：部署Varnish缓存服务器

1）编译安装软件

```
[root@svr5 ~]# yum -y install gcc readline-devel pcre-devel    //安装软件依赖包
[root@svr5 ~]# useradd -s /sbin/nologin varnish                //创建账户
[root@svr5 ~]# tar -xzf varnish-3.0.6.tar.gz
[root@svr5 ~]# cd varnish-3.0.6
```

Top

```
[root@svr5 varnish-3.0.6]# ./configure --prefix=/usr/local/varnish
[root@svr5 varnish-3.0.6]# make && make install
```

2）复制启动脚本及配置文件

```
[root@svr5 varnish-3.0.6]# cp redhat/varnish.initrc /etc/init.d/varnish
[root@svr5 varnish-3.0.6]# cp redhat/varnish.sysconfig /etc/sysconfig/varnish
[root@svr5 varnish-3.0.6]# ln -s /usr/local/varnish/sbin/varnishd /usr/sbin/
[root@svr5 varnish-3.0.6]# ln -s /usr/local/varnish/bin/* /usr/bin/
```

3）修改Varnish文件

```
[root@svr5 ~]# vim /etc/sysconfig/varnish
66行：VARNISH_LISTEN_PORT=80                    #默认端口
89行：VARNISH_STORAGE_SIZE=64M                  #定义缓存大小
92行：VARNISH_STORAGE="malloc,${VARNISH_STORAGE_SIZE}"    #基于内存方式缓存
```

4）修改代理配置文件

```
[root@svr5 ~]# mkdir /etc/varnish
[root@svr5 ~]# cp /usr/local/varnish/etc/default.vcl /etc/varnish/
[root@svr5 ~]# uuidgen > /etc/varnish/secret
[root@svr5 ~]# vim  /etc/varnish/default.vcl
backend default {
```

Top

```
        .host = "192.168.2.100";
        .port = "80";
    }
[root@svr5 ~]# service varnish start
```

步骤三：客户端测试

1）客户端开启浏览器访问

```
[root@client ~]# curl http://192.168.4.5
```

步骤四：其他操作

1）查看varnish日志

```
[root@svr5 ~]# varnishlog                //varnish日志
[root@svr5 ~]# varnishncsa               //访问日志
```

2）更新缓存数据，在后台web服务器更新页面内容后，用户访问代理服务器看到的还是之前的数据，说明缓存中的数据过期了需要更新（默认也会自动更新，但非实时更新）。

```
[root@svr5 ~]# varnishadm -S /etc/varnish/secret -T 127.0.0.1:6082 ban.url 页面文件名
//清空缓存数据，支持正则表达式
```

Top