

《计算机网络》课后习题

目录

《计算机网络》课后习题	1
第一题	2
要求	2
1. 使用 Nginx 返回 qq.com 内容	2
2. 搭建 lvs+nginx	2
2.1 搭建 Nginx	2
2.2 Nginx 实现负载均衡	4
2.3 搭建 lvs	5
第二题	6
要求	6
结论	7
第三题	8
要求	8
抓包	8
结论	8
第四题	9
要求	9
抓包	9
结论	10

第一题

要求

搭建 lvs, nginx 配置, 搭建一个高可用的 web server, 可以返回 qq.com 的内容;

1. 使用 Nginx 返回 qq.com 内容

在 devCloud 上安装配置 nginx, 并且修改配置文件



在 location 里加入 proxy_pass, 接收到请求后, 自动跳转到 qq.com, 以此来达到返回 qq.com 内容的效果



2. 搭建 lvs+nginx

2.1 搭建 Nginx

由于只有一台 devCloud, 只能自己给自己转发, 需要搭建两台 nginx 用于调度

1、下载安装 nginx

2、修改配置文件指定目录:

```
[root@VM-238-73-centos /usr/local/src/nginx-1.6.2]# ./configure
--prefix=/root/nginxs/nginxxs2
checking for OS
+ Linux 3.10.107-1-tlinux2_kvm_guest-0049 x86_64
checking for C compiler ... found
+ using GNU C compiler
```

(上面的命令行的意思就是启动 configure 配置，并设置 nginx 的路径，; --prefix 选项是配置安装的路径，如果不配置该选项，安装后可执行文件默认放在 /usr/local/bin，库文件默认放在 /usr/local/lib，配置文件默认放在 /usr/local/etc，其它的资源文件放在 /usr/local/share，比较凌乱)

3、执行 make && make install 命令安装到指定文件夹，并且进入到对应文件夹

```
[root@VM-238-73-centos ~/nginxs/nginxxs2]# ls
conf  html  logs  sbin
[root@VM-238-73-centos ~/nginxs/nginxxs2]#
```

4、修改 conf/nginx.conf 文件，修改端口号，修改 user root，否则会发生 403 forbidden 访问出错

```
>_ 9.134.238.73 x

user  root;
worker_processes  1;

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

server {
    listen      8002;
    server_name localhost;
```

5、为了验证方便，修改 html 文件

```
<body>
<h1>Welcome to nginx!</h1>

<p>this is nginx 8002!!! [for showrli] </p>
<script>
console.log("this is nginx 8002!!! [for showerli]")
</script>
</body>
```

6、启动 nginx（其实就是打开 sbin 目录下的 nginx 文件），进行访问



Welcome to nginx!

this is nginx 8002!!! [for showrli]

7、同样操作，再次启动一个 nginx 并成功访问。这样就有两个 nginx 监听不同端口，来模拟两台服务器

2.2 Nginx 实现负载均衡

1、修改配置文件，新增 upstream 模块，采用加权轮询，即根据配置的权重的大小而分发给不同服务器不同数量的请求。

```
upstream myserver {
    server 9.134.238.73:8001 weight=2;
    server 9.134.238.73:8002 weight=1;
}
```

2、最后访问 9.135.154.236，我们发现，按照轮询的权重访问两次 8001，再访问一次 8002，即实现了负载均衡。

Welcome to nginx!

this is nginx 8001!!! [for showerli]

Welcome to nginx!

this is nginx 8002!!! [for showerli]

2.3 搭建 lvs

安装 Director Server 和 Real Server: TencentOS 自带

配置 Director Server

在网卡 eth1 上绑定一个虚拟设备 eth1:0, 同时设置虚拟 IP 为 192.168.101.100

```
[root@VM-238-73-centos ~]# ifconfig eth1:0 192.168.101.100 bro  
adcast 192.168.101.100 netmask 255.255.255.255 up
```

添加路由, 指向虚拟 IP

```
[root@VM-238-73-centos ~]# route add -host 192.168.101.100 dev  
eth1:0
```

启用 IP 转发, 设置值为 1

```
[root@VM-238-73-centos ~]# echo 1 > /proc/sys/net/ipv4/ip_forw  
ard
```

添加虚拟 IP 转发规则, -rr 设置为轮询转发

```
[root@VM-238-73-centos ~]# ipvsadm -A -t 192.168.101.100:80 -s  
rr
```

```
[root@VM-238-73-centos ~]# ipvsadm -a -t 192.168.101.100:80 -r  
9.134.238.73:8002 -m  
[root@VM-238-73-centos ~]# ipvsadm -a -t 192.168.101.100:80 -r  
9.134.238.73:8001 -m
```

访问 192.168.101.100, 会转发到 8002 和 8001

```
[root@VM-238-73-centos ~]# curl 192.168.101.100
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>this is nginx 8001!!! [for showerli] </p>
<script>
console.log("this is nginx 8001 [for showerli]")
</script>
</body>
</html>
```

第二题

要求

在命令行下 curl 请求 www.qq.com 并在本机抓包(tcpdump 命令), 找到 TCP 三次握手, TCP 四次挥手, 发送 HTTP 请求, 接收 HTTP 请求的对应包并截图标注出来。(提示 通过 wireshark 查看抓包文件)

抓包

使用 tcpdump 监听 eth1 网卡的 qq.com

```
[root@VM-238-73-centos ~]# tcpdump -i eth1 host qq.com
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
```

再使用 curl 请求 qq.com

```
[root@VM-238-73-centos ~]# curl -vi qq.com
* About to connect() to qq.com port 80 (#0)
* Trying 61.129.7.47...
* Connected to qq.com (61.129.7.47) port 80 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: qq.com
```

可以发现 tcpdump 监听到了我们对 qq.com 的请求，结果如下

```
21:34:52.548284 IP 9.134.238.73.46949 > 61.129.7.47.http: Flag
s [S], seq 627357368, win 14600, options [mss 1460,sackOK,TS v
al 96007189 ecr 0,nop,wscale 7], length 0
21:34:52.548450 IP 61.129.7.47.http > 9.134.238.73.46949: Flag
s [S.], seq 1003400938, ack 627357369, win 14480, options [mss
1424,sackOK,TS val 2206294941 ecr 96007189,nop,wscale 7], len
gth 0
21:34:52.548471 IP 9.134.238.73.46949 > 61.129.7.47.http: Flag
s [.), ack 1, win 115, length 0
21:34:52.548524 IP 9.134.238.73.46949 > 61.129.7.47.http: Flag
s [P.), seq 1:71, ack 1, win 115, length 70
21:34:52.548653 IP 61.129.7.47.http > 9.134.238.73.46949: Flag
s [.), ack 71, win 114, options [nop,nop,TS val 2206294941 ecr
96007189], length 0
21:34:52.621527 IP 61.129.7.47.http > 9.134.238.73.46949: Flag
s [P.), seq 1:348, ack 71, win 114, options [nop,nop,TS val 22
06294959 ecr 96007189], length 347
21:34:52.621541 IP 9.134.238.73.46949 > 61.129.7.47.http: Flag
s [.), ack 348, win 123, length 0
21:34:52.621745 IP 9.134.238.73.46949 > 61.129.7.47.http: Flag
s [F.), seq 71, ack 348, win 123, length 0
21:34:52.621936 IP 61.129.7.47.http > 9.134.238.73.46949: Flag
s [F.), seq 348, ack 72, win 114, options [nop,nop,TS val 2206
294959 ecr 96007189], length 0
21:34:52.621948 IP 9.134.238.73.46949 > 61.129.7.47.http: Flag
s [.), ack 349, win 123, length 0
```

结论

tcpdump 中[S]表示建立连接，[S.]表示确认收到连接；[F]表示结束连接，[F.]表示确认结束连接，前三个包为 TCP 三次握手，后四个包为 TCP 四次挥手，中间表示 HTTP 请求。

```
21:34:52.548284 IP 9.134.238.73.46949 > 61.129.7.47.http: Flags [S], seq 62735736
8, win 14600, options [mss 1460,sackOK,TS val 96007189 ecr 0,nop,wscale 7], lengt
h 0
21:34:52.548450 IP 61.129.7.47.http > 9.134.238.73.46949: Flags [S.), seq 1003400
938, ack 627357369, win 14480, options [mss 1424,sackOK,TS val 2206294941 ecr 960
07189,nop,wscale 7], length 0
21:34:52.548471 IP 9.134.238.73.46949 > 61.129.7.47.http: Flags [.), ack 1, win 1
15, length 0
21:34:52.548524 IP 9.134.238.73.46949 > 61.129.7.47.http: Flags [P.), seq 1:71, a
ck 1, win 115, length 70
21:34:52.548653 IP 61.129.7.47.http > 9.134.238.73.46949: Flags [.), ack 71, win
114, options [nop,nop,TS val 2206294941 ecr 96007189], length 0
21:34:52.621527 IP 61.129.7.47.http > 9.134.238.73.46949: Flags [P.), seq 1:348,
ack 71, win 114, options [nop,nop,TS val 2206294959 ecr 96007189], length 347
21:34:52.621541 IP 9.134.238.73.46949 > 61.129.7.47.http: Flags [.), ack 348, win
123, length 0
21:34:52.621745 IP 9.134.238.73.46949 > 61.129.7.47.http: Flags [F.), seq 71, ack
348, win 123, length 0
21:34:52.621936 IP 61.129.7.47.http > 9.134.238.73.46949: Flags [F.), seq 348, ac
k 72, win 114, options [nop,nop,TS val 2206294959 ecr 96007189], length 0
21:34:52.621948 IP 9.134.238.73.46949 > 61.129.7.47.http: Flags [.), ack 349, win
123, length 0
```

第三题

要求

Traceroute www.qq.com 并抓包，总共探测了多少次，探测包的 TTL 是如何变化的，使用的哪个传输层协议？

抓包

先使用 tcpdump 监听 eth1 网卡的 qq.com，再用 traceroute 探测 qq.com。可以看到 tcpdump 抓到了很多包：

```
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
21:41:48.059200 IP 9.134.238.73.58299 > 61.129.7.47.traceroute : UDP, length 32
21:41:48.059217 IP 9.134.238.73.33022 > 61.129.7.47.33435: UDP, length 32
21:41:48.059223 IP 9.134.238.73.58320 > 61.129.7.47.33436: UDP, length 32
21:41:48.059228 IP 9.134.238.73.60568 > 61.129.7.47.33437: UDP, length 32
21:41:48.059233 IP 9.134.238.73.56781 > 61.129.7.47.33438: UDP, length 32
21:41:48.059238 IP 9.134.238.73.60362 > 61.129.7.47.33439: UDP, length 32
21:41:48.059243 IP 9.134.238.73.44523 > 61.129.7.47.33440: UDP, length 32
21:41:48.059248 IP 9.134.238.73.54753 > 61.129.7.47.33441: UDP, length 32
21:41:48.059252 IP 9.134.238.73.53521 > 61.129.7.47.33442: UDP, length 32
21:41:48.059257 IP 9.134.238.73.52447 > 61.129.7.47.33443: UDP, length 32
21:41:48.059263 IP 9.134.238.73.45711 > 61.129.7.47.33444: UDP, length 32
21:41:48.059268 IP 9.134.238.73.39735 > 61.129.7.47.33445: UDP, length 32
21:41:48.059275 IP 9.134.238.73.41007 > 61.129.7.47.33446: UDP, length 32

traceroute to qq.com (61.129.7.47), 30 hops max, 60 byte packets
 0  9.134.108.78 (9.134.108.78)  0.177 ms 9.134.108.112 (9.134.108.112)  0.206 ms  0.235 ms
 1  * * *
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
```

中间省略若干

结论

- 1、总共探测了 30 次。
- 2、可以看到每次的 TTL 都+1，traceroute 送出一个 TTL 是 1 的 IP 数据包到目的地，当路径上的第一个路由器收到这个数据包时，它将 TTL 减 1。此时，TTL 变为 0，所以该路由器会将此数据包丢掉，并送回一个「ICMP time exceeded」消息（包括发 IP 包的源地址，IP 包的所有内容及路由器的 IP 地址），traceroute 收到这个消息后，便知道这个路由器存在于这个路径上，接着 traceroute 再送出另一个 TTL 是 2 的数据包，发现第 2 个路由器..... traceroute 每次将送出的数据包的 TTL 加 1 来发现另一个路由器，这个重复的动作一直持续到某数据包抵达目的地。当数据包到达目的地后，该主机则不会送回 ICMP time exceeded 消息，一旦到达目的地，由于 traceroute 通过 UDP 数据包向不常见端口(30000 以上)发送数据包，因此会收到「ICMP port unreachable」消息，故可判断到达目的地。
- 3、这里使用的是 UDP 协议。

第四题

要求

ping www.qq.com, 然后抓包, 观察本机发送的包, 目的 IP 地址和目的 MAC 分别是什么的地址, 为什么?

抓包 ping qq.com

```
[root@VM-238-73-centos ~]# ping qq.com
PING qq.com (61.129.7.47) 56(84) bytes of data.
64 bytes from 61.129.7.47: icmp_seq=1 ttl=48 time=33.0 ms
64 bytes from 61.129.7.47: icmp_seq=2 ttl=48 time=32.9 ms
64 bytes from 61.129.7.47: icmp_seq=3 ttl=48 time=33.0 ms
64 bytes from 61.129.7.47: icmp_seq=4 ttl=48 time=32.9 ms
64 bytes from 61.129.7.47: icmp_seq=5 ttl=48 time=32.9 ms
64 bytes from 61.129.7.47: icmp_seq=6 ttl=48 time=32.9 ms
```

使用 tcpdump 抓包并且保存到文件,

```
[root@VM-238-73-centos ~]# tcpdump -i eth1 host qq.com -v -wping
qq.pcap
tcpdump: listening on eth1, link-type EN10MB (Ethernet), capture
size 65535 bytes
^C280 packets captured
280 packets received by filter
0 packets dropped by kernel
[root@VM-238-73-centos ~]#
```

将得到 pingqq.pcap 的下载到本地:

```
[root@VM-238-73-centos ~]# sz ~/pingqq.pcap
Received - pingqq.pcap 725.47 KB/s Spend: 0 seconds
```

放到 wireshark 上解析

The screenshot shows the Wireshark interface with the file 'pingqq.pcap' loaded. The packet list pane displays several ICMP Echo (ping) requests and replies. The first packet is selected, showing details in the packet details pane. The Ethernet II section shows the destination MAC address as 'fe:ee:98:6f:c0:b7' and the source MAC address as 'fe:ee:98:6f:c0:b7'. The packet bytes pane shows the raw data of the ICMP request, including the type, code, sequence number, and TTL.

结论

- 1、目的 IP 地址是 61.129.7.47，而目的 MAC 地址是 fe:ee:98:9f:c0:b7。
- 2、这是因 IP 用于网络寻址，它的作用空间是整个地球的网络，而 MAC 用于链路层寻址，作用空间是局域网。IP 为逻辑地址，它不是一成不变的；MAC 是物理地址，网卡以及路由器的物理地址都是在出厂时固定的，而且全世界的每一个设备的 MAC 地址都是不重复的，唯一的。