Linux Firewall Exploration Lab

实验环境：
Machine A IP: 192.168.220.133
Machine B IP: 192.168.220.129

Task 1: Using Firewall

```
[09/17/20]seed@VM:~$ sudo ufw deny from 192.168.220.133 to any port 23
Rule added
```

图 1.1 添加规则

```
[09/17/20]seed@VM:~$ telnet 192.168.220.129
Trying 192.168.220.129...
telnet: Unable to connect to remote host: Connection timed out
```

图 1.2 telnet deny

对主机 B 设置过滤规则，主机 A telnet 访问失败。

```
[09/17/20]seed@VM:~$ ifconfig
ens33     Link encap:Ethernet  HWaddr 00:0c:29:73:43:db
          inet addr:192.168.220.134  Bcast:192.168.220.255  Mask:255.255.255.0
          inet6 addr: fe80::f517:6227:78a7:2efa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:17 errors:0 dropped:0 overruns:0 frame:0
          TX packets:67 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1894 (1.8 KB)  TX bytes:7914 (7.9 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:64 errors:0 dropped:0 overruns:0 frame:0
          TX packets:64 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:21261 (21.2 KB)  TX bytes:21261 (21.2 KB)

[09/17/20]seed@VM:~$ telnet 192.168.220.129
Trying 192.168.220.129...
Connected to 192.168.220.129.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Thu Sep 17 10:14:08 CST 2020 from 192.168.220.129 on pts/17
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)
```

图 1.3 telnet allow

主机 C（IP 地址为 192.168.220.134）telnet 访问主机 B，连接成功，说明 B 的过滤只针对 A。
主机 A 对主机 B 的过滤同上。

```
[09/17/20]seed@VM:~$ ping www.baidu.com
PING www.a.shifen.com (61.135.185.32) 56(84) bytes of data.
64 bytes from 61.135.185.32: icmp_seq=1 ttl=128 time=43.5 ms
64 bytes from 61.135.185.32: icmp_seq=2 ttl=128 time=39.6 ms
64 bytes from 61.135.185.32: icmp_seq=3 ttl=128 time=37.3 ms
64 bytes from 61.135.185.32: icmp_seq=4 ttl=128 time=42.9 ms
64 bytes from 61.135.185.32: icmp_seq=5 ttl=128 time=38.7 ms
64 bytes from 61.135.185.32: icmp_seq=6 ttl=128 time=36.9 ms
```

图 1.4 连接外部网站

选择 www.baidu.com（IP 61.135.185.32）作为外部网站，设置规则禁止主机 A 访问此网站。通过修改/etc/hosts 文件使得域名唯一定位到此 IP，避免网站的负载均衡对实验结果造成影响。

```
[09/17/20]seed@VM:~$ sudo ufw deny out from any to 61.135.185.32
Rule added
[09/17/20]seed@VM:~$ ping www.baidu.com
PING www.baidu.com (61.135.185.32) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
```

图 1.5 添加规则

添加规则禁止主机访问外部 IP 地址。此时再运行 ping 发现操作被禁止。

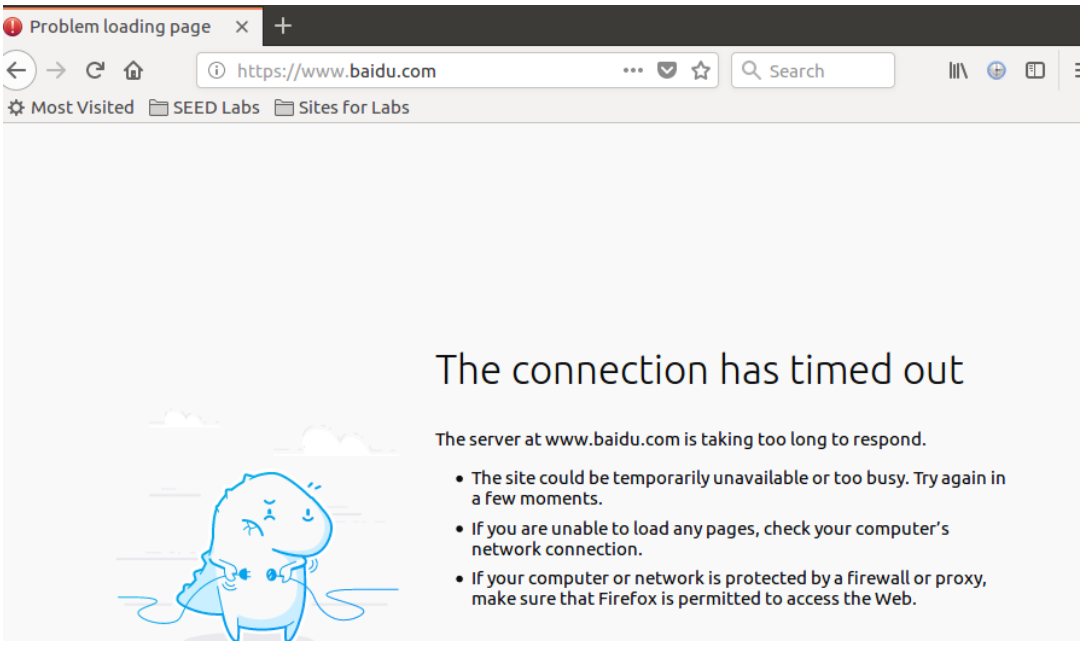


图 1.6 访问 ww.baidu.com

通过浏览器访问 www.baidu.com，连接超时。

Task 2: Implementing a Simple Firewall

在 192.168.220.129 主机上利用防火墙实现以下功能：禁止本机被 telnet 连接，禁止本机发出 telnet 连接，禁止本机被 ICMP 访问，禁止本机与 192.168.220.134 主机之间的连接。

简单起见，截取处理到达报文的代码，如下：

```c
if(iph->protocol == IPPROTO_ICMP){
    printk(KERN_INFO "post ICMP is banned\n");
    return NF_DROP;
}
else if(iph->protocol == IPPROTO_TCP && tcph->dest == htons(23)){
    printk(KERN_INFO "post Dropping telnet packet to %d.%d.%d.%d\n",
            ((unsigned char *)&iph->daddr)[0],
            ((unsigned char *)&iph->daddr)[1],
            ((unsigned char *)&iph->daddr)[2],
            ((unsigned char *)&iph->daddr)[3]);
            return NF_DROP;
}
else if(iph->daddr == -2032359232){
    printk(KERN_INFO "post Connection with 192.168.220.134 is forbidden\n");
    return NF_DROP;
}
else{
    return NF_ACCEPT;
}
```

图 2.1 核心代码

测试效果如下：

主机 192.168.220.129（运行内核模块的主机）：

```
[09/17/20]seed@VM:~/code$ ping 192.168.220.134
PING 192.168.220.134 (192.168.220.134) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 192.168.220.134 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1021ms

[09/17/20]seed@VM:~/code$ telnet 192.168.220.134
Trying 192.168.220.134...
telnet: Unable to connect to remote host: Connection timed out
```

主机 192.168.220.133（禁止 ping129,禁止 telnet 连接 129）：

```
[09/17/20]seed@VM:~$ ping 192.168.220.129
PING 192.168.220.129 (192.168.220.129) 56(84) bytes of data.
^C
--- 192.168.220.129 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1001ms

[09/17/20]seed@VM:~$ telnet 192.168.220.129
Trying 192.168.220.129...
telnet: Unable to connect to remote host: Connection timed out
```

主机 192.168.220.134（禁止与 129 有任何来往）：

```
[09/17/20]seed@VM:~$ ping 192.168.220.129
PING 192.168.220.129 (192.168.220.129) 56(84) bytes of data.
^C
--- 192.168.220.129 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1028ms

[09/17/20]seed@VM:~$ telnet 192.168.220.129
Trying 192.168.220.129...
telnet: Unable to connect to remote host: Connection timed out
```

截取日志如下：

```
[09/17/20]seed@VM:~/code$ dmesg | tail -20
[  539.349284] pre Dropping telnet packet from 192.168.220.129
[  555.477534] pre Dropping telnet packet from 192.168.220.129
[  556.259621] pre Dropping telnet packet from 192.168.220.129
[  557.280889] pre Dropping telnet packet from 192.168.220.129
[  559.297334] pre Dropping telnet packet from 192.168.220.129
[  563.393680] pre Dropping telnet packet from 192.168.220.129
[  571.584912] pre Dropping telnet packet from 192.168.220.129
[  587.712739] pre Dropping telnet packet from 192.168.220.129
[  588.500751] pre Dropping telnet packet from 192.168.220.129
[  591.526213] post ICMP is banned
[  592.547766] post ICMP is banned
[  598.312269] post Dropping telnet packet to 192.168.220.134
[  599.331038] post Dropping telnet packet to 192.168.220.134
[  601.346863] post Dropping telnet packet to 192.168.220.134
[  605.410995] post Dropping telnet packet to 192.168.220.134
[  613.603217] post Dropping telnet packet to 192.168.220.134
[  620.481305] pre Dropping telnet packet from 192.168.220.129
[  629.730721] post Dropping telnet packet to 192.168.220.134
[  661.584763] post Connection with 192.168.220.134 is forbidden
[  662.499445] post Dropping telnet packet to 192.168.220.134
```

Task 3: Evading Egress Filtering

```
[09/17/20]seed@VM:~/code$ sudo ufw status numbered
Status: active

     To                         Action      From
     --                         ------      ----
[ 1] 23                         DENY OUT    192.168.220.129            (out)
[ 2] 120.92.147.239             DENY OUT    Anywhere                   (out)
```

图 3.1 ufw 规则

第一条规则限定了本机不可访问外部主机的 23 端口（telnet）。

第二条规则，实验中本要求禁止访问 www.facebook.com，但由于某些原因此网站本就不可访问，因此改为 www.bilibili.com。为了防止服务器负载均衡的干扰，将 www.bilibili.com 通过/etc/hosts 绑定到一个固定的服务器 120.92.147.239，从而形成第二条规则。

Task 3.a: Telnet to Machine B through the firewall

```
[09/17/20]seed@VM:~$ ssh -L 8000:192.168.220.134:23 seed@192.168.220.133
seed@192.168.220.133's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.
```

```
/bin/bash 78x21
Connection closed by foreign host.
[09/17/20]seed@VM:~$ telnet localhost 8000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Thu Sep 17 07:20:24 EDT 2020 from 192.168.220.133 on pts/4
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

[09/17/20]seed@VM:~$ ifconfig
ens33     Link encap:Ethernet  HWaddr 00:0c:29:73:43:db
          inet addr:192.168.220.134  Bcast:192.168.220.255  Mask:255.255.255.0
```

图 3.1.1 建立 SSH 隧道

上方的终端与 192.168.220.133 建立了 ssh 隧道。下方的终端通过隧道使用 telnet 连接到 192.168.220.134，运行 ifconfig 可以看到 IP 确实是 192.168.220.134。

```
192.168.220.133  192.168.220.129     126 SSHv2  Server: Encrypted packet (len=60)
192.168.220.129  192.168.220.133     166 SSHv2  Client: Encrypted packet (len=100)
192.168.220.133  192.168.220.129     110 SSHv2  Server: Encrypted packet (len=44)
192.168.220.134  192.168.220.133      78 TELNET Telnet Data ...
192.168.220.133  192.168.220.129     118 SSHv2  Server: Encrypted packet (len=52)
192.168.220.129  192.168.220.133     118 SSHv2  Client: Encrypted packet (len=52)
192.168.220.133  192.168.220.134      78 TELNET Telnet Data ...
192.168.220.134  192.168.220.133      90 TELNET Telnet Data ...
192.168.220.133  192.168.220.129     126 SSHv2  Server: Encrypted packet (len=60)
192.168.220.129  192.168.220.133     158 SSHv2  Client: Encrypted packet (len=92)
192.168.220.133  192.168.220.134     123 TELNET Telnet Data ...
```

图 3.1.2 流量分析

从 wireshark 获取的流量中可以看到 129 和 133 建立了 SSH 连接，133 和 134 建立了 telnet 连接。而 telnet 报文发出前后都有 SSH 加密数据在 129 和 133 之间传送。虽然无法解密 SSH 负载，但是可以猜测 129 将“与 134 建立 telnet 连接“的信息和必要数据封装在 SSH 隧道中交给了 133，133 作为代理和 134 进行了通信。

Task 3.b: Connect to Bilibili using SSH Tunnel

图 3.2.1 使用代理连接网站

建立 SSH 连接并设置代理后，浏览器可正常访问被 ufw 禁止的网站。

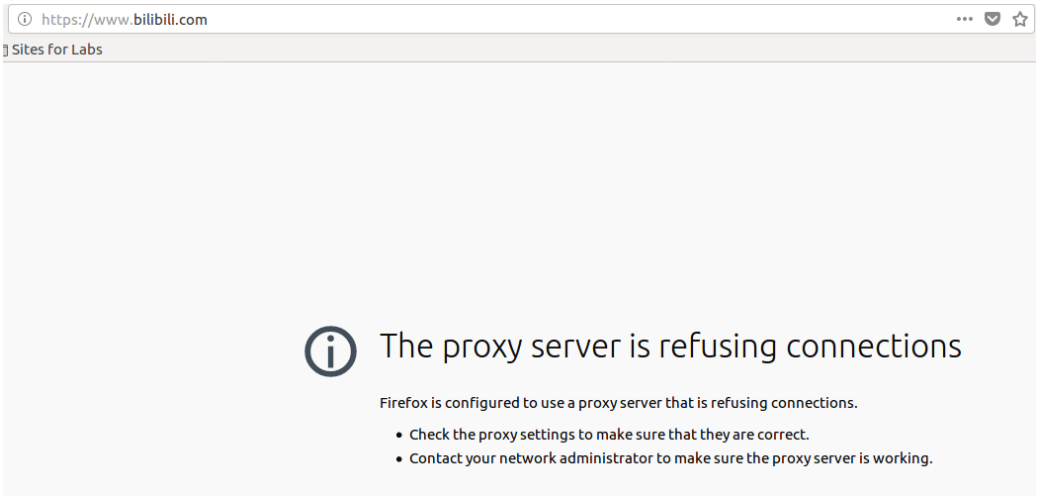


图 3.2.2 断开 SSH 连接

再断开 SSH 连接，清空浏览器缓存并重新连接，可以看见浏览器报错“代理拒绝连接“。

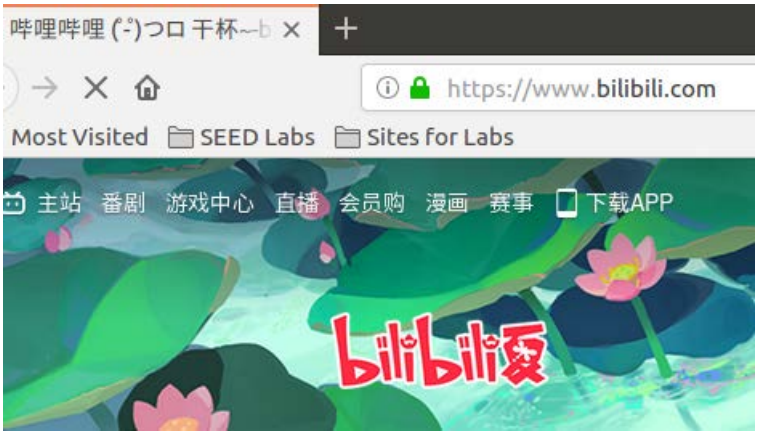


图 3.2.3 重建 SSH 连接

重建 SSH 连接后网页访问恢复。

因为 ufw 并没有深入的检测报文负载，所以可以借用代理来绕过防火墙。将针对目标网站 www.bilibili.com 的访问请求与网页数据加密封装在 SSH 数据段，由代理完成于网站的交互。

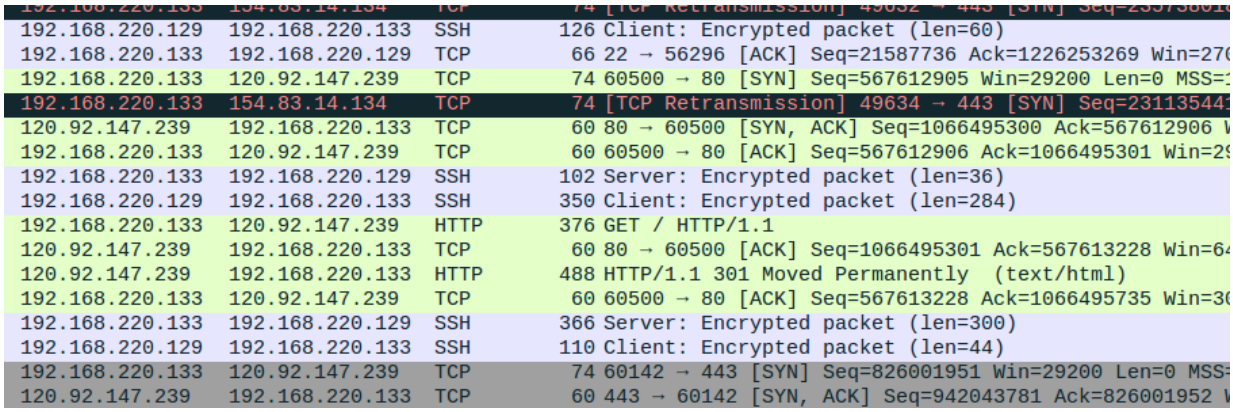


图 3.2.3 报文抓取

从报文中也可以看出，192.168.220.129 和 192.168.220.133 之间建立 SSH 连接，然后 192.168.220.133 与网站进行 HHTP 等报文交互。

当隧道断开时，相当于代理离线，所以浏览器无法通过代理访问先前的网站。

Task 4: Evading Ingress Filtering

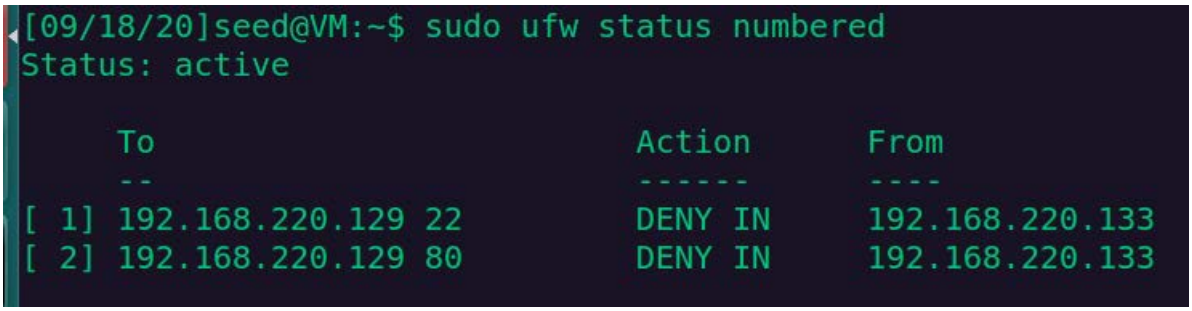


图 4.1 阻断对 VMB 22 端口与 80 端口的访问

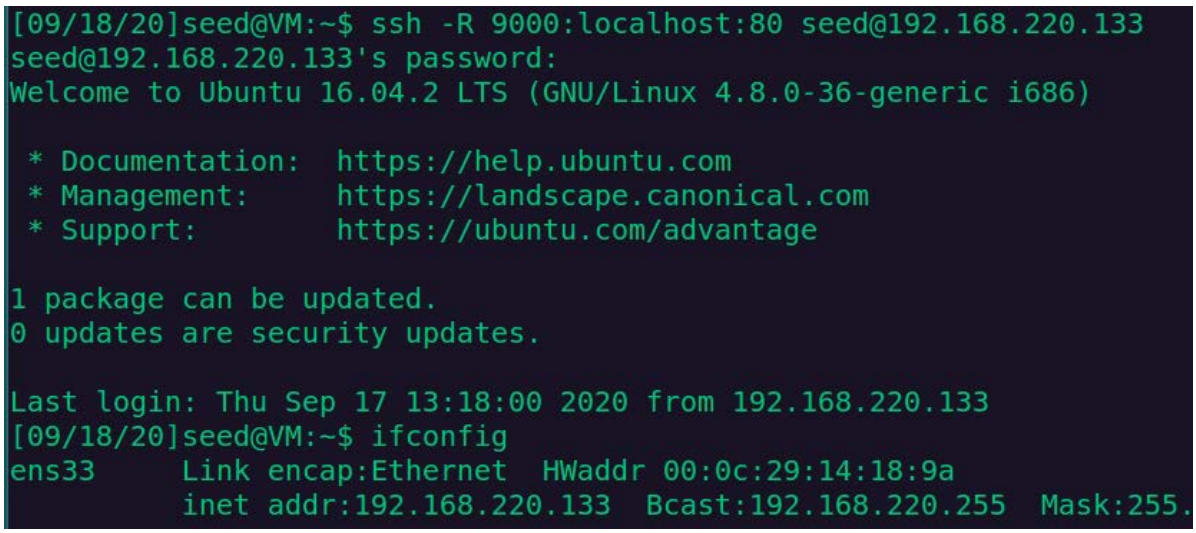


图 4.2 创建反向 SSH 隧道

在 VM B 上通过 SSH 连接 VM A（seed@192.168.220.133），并构造从任意地址 9000 端口到 VM B 80 端口的隧道。

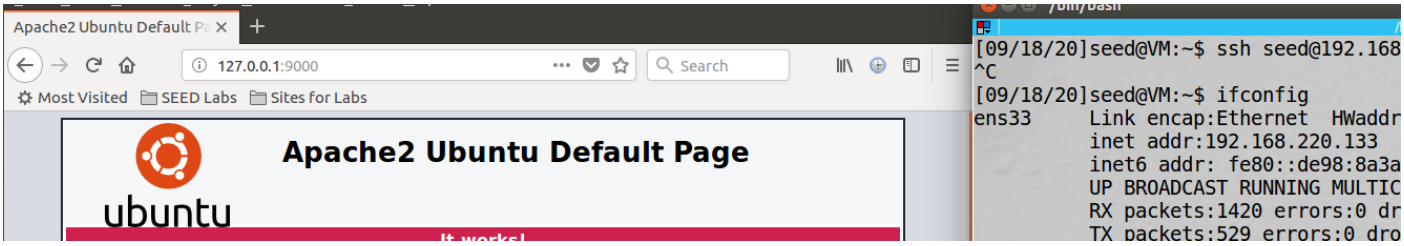


图 4.3 VM2 访问本地 9000 端口

在 VM A 上访问本地 9000 端口（右侧终端显示 IP 地址为 192.168.220.133，确实为 VM A）。此时浏览器显示的是 VM B 的 Apache 主页，即从外部访问内网主机 80 端口成功。

Firewall Evasion Lab: Bypassing Firewalls using VPN

Task 1: VM Setup

实验环境：
VM1（VPN Client）IP：192.168.220.129
VM2（VPNServer）IP：192.168.220.133

Task 2: Set up Firewall

图 2.1 设置 iptables 规则

此实验中，我们禁止 VM1 访问 www.baidu.com。通过多次 ICMP 探测发现域名对应的 IP 地址在 61.135.0.0/16 网段下。因此用 iptables 禁止 VM1 访问此网段，此时 ping www.baidu.com 发现操作被禁止。

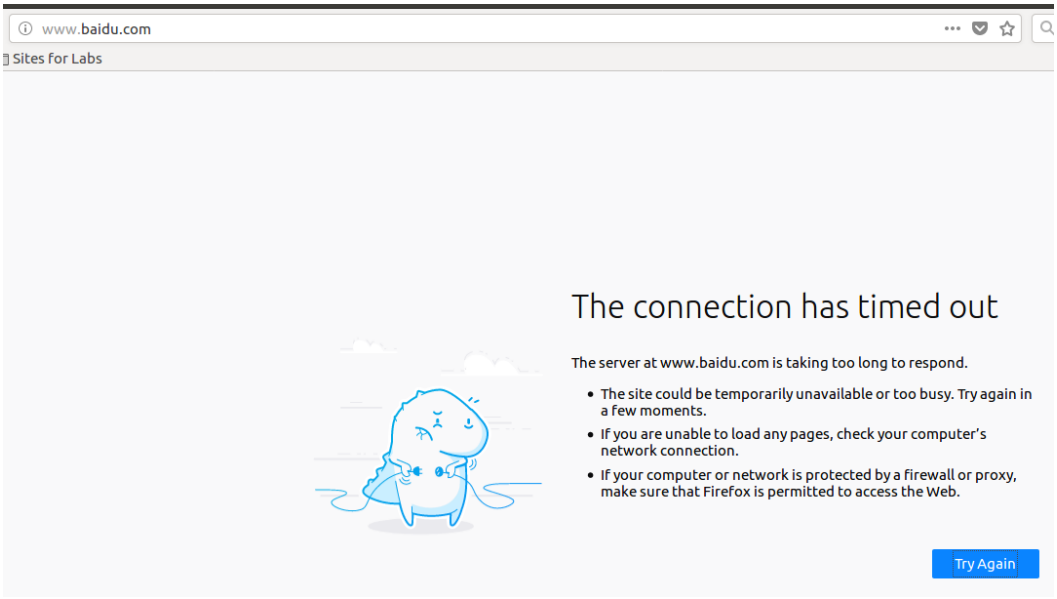


图 2.2 访问超时

Task 3: Bypassing Firewall using VPN

首先在 VM1 和 VM2 上分别运行 vpnclient 和 vpnserver，两台主机上分别虚拟出网卡 tun0，给 VM1 的 tun0 分配 IP 地址 192.168.53.5，给 VM2 的 tun0 分配 IP 地址 192.168.53.1，并设置 192.168.53.0/24 网段的报文指向 tun0 网卡。设置结束后在 VM1 上 ping 192.168.53.1，结果如下：

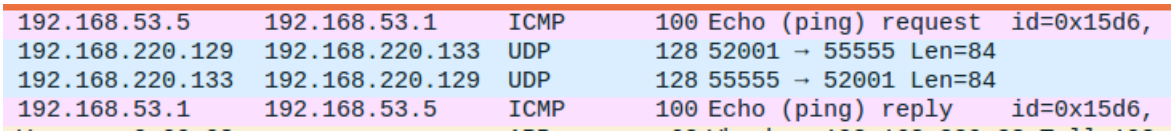


图 3.1 VM1 wireshark

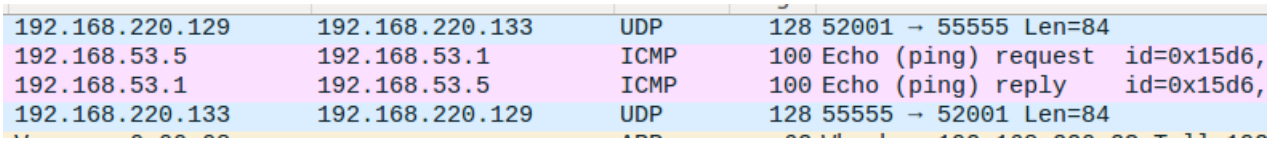


图 3.2 VM2 wireshark

可以看到 VM1 首先通过 tun0 向 192.168.53.1 发出 ICMP request，然后这个报文被封装进 UDP 数据段由 ens33 网卡（IP 地址为 192.168.220.129）发往 VM2（IP 地址为 192.168.220.133）。应答报文装在 UDP 中发回来，程序解开数据段发现是发往 192.168.53.5 的 ICMP reply，因此交给 tun0 处理，形成图 3.1 中的最后一条报文。

对于 VM2，则嵌套过程刚好相反，首先接到 UDP 报文，拆开之后交给 ICMP 程序处理然后原路返回。

为了让 VM1 能够访问 61.135.0.0/16 网段（百度），我们设置路由让发往此网段的报文由 tun0 网卡处理。

图 3.3 设置路由

此时用 ping 探测 www.baidu.com，虽然没有“操作禁止”的限制，但是仍无应答报文。通过在 VM1 上抓包发现 ICMP 确实从 VM1 的网卡上发出，也经过 UDP 封装到达了 VM2，但是对应的 ICMP 应答报文的目的地址是 192.168.53.5，VMware 虚拟机并不知道这个地址应该发给谁（设置的私有网络），因此该报文被丢弃，没有程序处理。

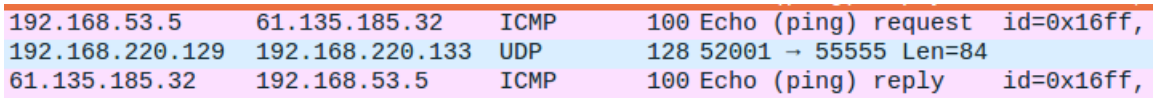


图 3.4 设置路由后在 VM1 上抓包

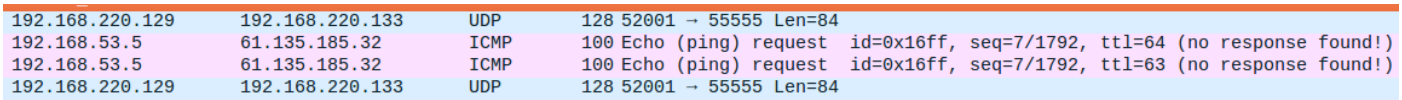


图 3.5 设置路由后在 VM2 上抓包

因此，在 VM2（VPN server）的对外网卡 ens33 上设置 nat，即在报文发给 VMware 之前进行 nat 转换。这样应答报文首先发给 VM2，再经过 nat 转换后才发往 192.168.53.1，而 VM2 是知道此网段的路由的。

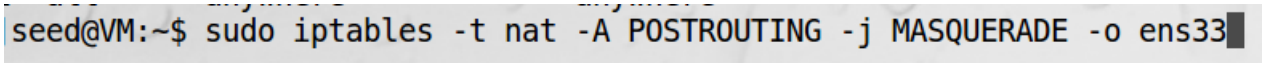
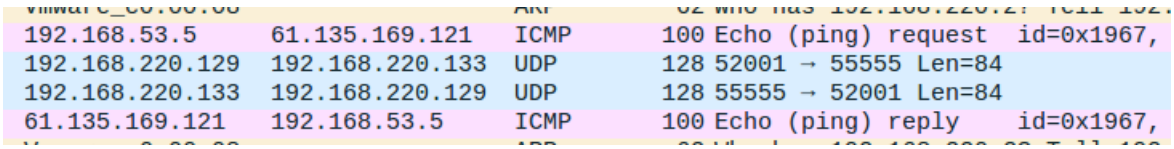


图 3.6 设置 nat

再次探测 www.baidu.com，可以连接，已绕过 iptables 的拦截。

图 3.7 ping www.baidu.com

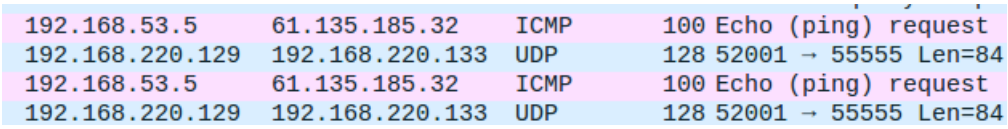


192.168.53.5	61.135.169.121	ICMP	100	Echo (ping) request id=0x1967,
192.168.220.129	192.168.220.133	UDP	128	52001 → 55555 Len=84
192.168.220.133	192.168.220.129	UDP	128	55555 → 52001 Len=84
61.135.169.121	192.168.53.5	ICMP	100	Echo (ping) reply id=0x1967,

图 3.8 VM1 wireshark

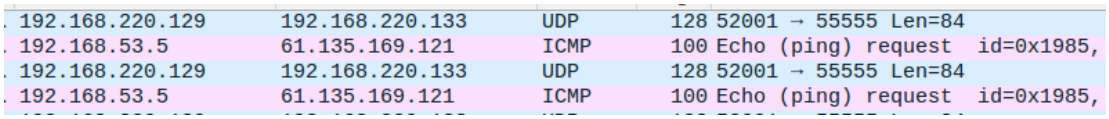
图 3.9 firefox 访问 www.baidu.com

需要说明的是，实验指导手册上要求用 sudo iptables -F 清空 VM2 的 iptables 规则，而这样做会带来错误。



192.168.53.5	61.135.185.32	ICMP	100	Echo (ping) request
192.168.220.129	192.168.220.133	UDP	128	52001 → 55555 Len=84
192.168.53.5	61.135.185.32	ICMP	100	Echo (ping) request
192.168.220.129	192.168.220.133	UDP	128	52001 → 55555 Len=84

图 3.10 VM1 wireshark



192.168.220.129	192.168.220.133	UDP	128	52001 → 55555 Len=84
192.168.53.5	61.135.169.121	ICMP	100	Echo (ping) request id=0x1985,
192.168.220.129	192.168.220.133	UDP	128	52001 → 55555 Len=84
192.168.53.5	61.135.169.121	ICMP	100	Echo (ping) request id=0x1985,

图 3.11 VM2 wireshark

可以看到，根本就没有 ICMP reply（即使是目的地址不可达的 ICMP reply 也没有），VM1 更不可能得到响应。猜测 VMware 在 iptables 中写入了一些默认规则，贸然删除使得来自 192.168.53.3 的 ICMP request 发不出去。因此，只要添加 sudo iptables -t nat -A POSTROUTING -j MASQUERADE -o ens33 即可。