Jinfeng Liu
Liujinfeng1209@gmail.com

Linux Firewall Exploration Lab Report

Task 1: Using Firewall

In this task, we enable the firewall on Linux machine and try to achieve 3 tasks, which are Prevent A from doing telnet to Machine B; Prevent B from doing telnet to Machine A; Prevent A from visiting an external web site(www.nyit.edu). The main command we used is: 1) ***sudo ufw deny out from IP_ADDR to IP_ADDR***

Task 2: Loadable Kernel Module & Netfilter

In this task, we mainly need to use LKM and Netfilter to implement the packet filtering module. We firstly created *Makefile* and *Myfilter.c*. Then used ***sudo make*** to compile it into an LKM. Then we used command ***sudo insmod Myfilter.ko*** – load *Myfilter.ko* module into Linux Kernel. After verification that our filter worked, then we use ***sudo rmmod Myfilter.ko*** – to remove *Myfilter.ko* module from Linux Kernel.

```
[06/09/19]seed@VM:~$ sudo insmod Myfilter.ko
[06/09/19]seed@VM:~$ lsmod
Module                  Size  Used by
Myfilter               16384  0
[06/09/19]seed@VM:~$ telnet 10.0.2.4
Trying 10.0.2.4...
telnet: Unable to connect to remote host: Connection timed out
```

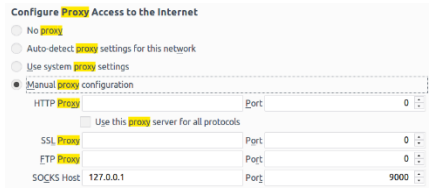
In the Myfilter.c file, we converted IP addresses into String datatype so that we can compare it in the hook. And we implemented 5 hooks in total. The hook we used on the right for example, we blocked SSH outgoing traffic from 10.0.2.4 to 10.0.2.15. Therefore, any SSH traffic that is going from 10.0.2.4 to 10.0.2.15 will be dropped by the code: ***return NF_DROP***.

```
snprintf(source, 16, "%pI4", &ip_header->saddr);
snprintf(destination, 16, "%pI4", &ip_header->daddr);
```

```
//REJECT SSH from 10.0.2.15 to 10.0.2.4
if ((strcmp(source, "10.0.2.4") == 0) && (strcmp(destination, "10.0.2.15") == 0) && (dst_port == 22))
{
        return NF_DROP;
}
printk(KERN_INFO "--------------------");
```

Task 3a, b: Telnet to Machine B through the firewall; Connect to Facebook using SSH Tunnel

In these tasks, we are trying to bypass the Linux firewall by establishing an SSH tunnel and all traffic will be going through this tunnel to avoid inspection. In order to reach port 23 (telnet) at Machine C (10.0.2.5) as destination, Machine A should first go through Machine B (10.0.2.4) as an intermedia using command ***ssh -L 8000:10.0.2.5:23 seed@10.0.2.4***. After verification, it is successful. Then we dynamically forwarding all traffic through port 9000 to Machine B using command ***ssh -D 9000 -C seed@10.0.2.15***. But we need to set up proxy at Machine A to reach the website.



Task 4: Evading Ingress Filtering

In this task, we set up a reverse tunnel between A and B on port 22 and 80 using command ***ssh -R 9000:10.0.2.15:80 seed@10.0.2.4; ssh -R 9001:10.0.2.15:22 seed@10.0.2.4.*** Modifying the localhost website using ***sudo subl /var/www/html/index.html.*** We then viewed it on Machine B's browser typing ***localhost:9000***. We are also able to SSH to Machine A from Machine B.