SEED LAB 5 (Firewall)

Andrew Simon
N00695969

Task 1: Using a Firewall

The first task of this lab is to work with Linux's **iptables** and **ufw** programs to gain a basic understanding of how packet filter firewalls operate. After setting up my two machines (A: 10.0.2.4 and B: 10.0.2.5), I make sure that the default ufw settings will accept traffic:

```
# Set the default input policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_INPUT_POLICY="ACCEPT"
```

The first part of the task is to prevent Machine A from communicating with Machine B through telnet. Before writing the rule, I can telnet into Machine B from Machine A:

```
[03/26/24]seed@VM:.../default$ telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sun Mar 24 19:58:42 EDT 2024 from 10.0.2.5 on
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic

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

Once I implement the ufw rule, I am no longer able to telnet from Machine A to B:

```
[03/26/24]seed@VM:~$ sudo ufw deny out from 10.0.2.4 to 10.0.2.5 po
rt 23
Rule updated
[03/26/24]seed@VM:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
23                         ALLOW OUT   10.0.2.5
23                         DENY OUT    10.0.2.4
10.0.2.5 23                DENY OUT    10.0.2.4

[03/26/24]seed@VM:~$ telnet 10.0.2.5
Trying 10.0.2.5...
```

The next part of the task is to deny Machine B from Telnetting to Machine A. Before the rule, the functionality works perfectly:

```
[03/26/24]seed@VM:~/Labs$ sudo ufw status
Status: inactive
[03/26/24]seed@VM:~/Labs$ sudo ufw enable
Firewall is active and enabled on system startup
[03/26/24]seed@VM:~/Labs$ sudo ufw status
Status: active
[03/26/24]seed@VM:~/Labs$ telnet 10.0.2.5
Trying 10.0.2.5...
Connected to 10.0.2.5.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
```

Once implementing the proper ufw rule, I can no longer telnet from Machine B to Machine A:

```
[03/26/24]seed@VM:~/Labs$ sudo ufw deny out from 10.0.2.5 to 10.0.2
.4 port 23
Rule updated
[03/26/24]seed@VM:~/Labs$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
10.0.2.4 23                DENY OUT    10.0.2.5

[03/26/24]seed@VM:~/Labs$ telnet 10.0.2.4
Trying 10.0.2.4...
```

The final part of this task is to prevent Machine A from reaching a specific website. Before implementing this rule, my machine is able to reach walmart.com:

```
[03/26/24]seed@VM:~$ dig +short walmart.com
23.44.192.164
[03/26/24]seed@VM:~$ ping 23.44.192.164
PING 23.44.192.164 (23.44.192.164) 56(84) bytes of data.
64 bytes from 23.44.192.164: icmp_seq=1 ttl=51 time=36.7 ms
64 bytes from 23.44.192.164: icmp_seq=2 ttl=51 time=24.7 ms
64 bytes from 23.44.192.164: icmp_seq=3 ttl=51 time=20.6 ms
64 bytes from 23.44.192.164: icmp_seq=4 ttl=51 time=13.2 ms
64 bytes from 23.44.192.164: icmp_seq=5 ttl=51 time=22.7 ms
^C
--- 23.44.192.164 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 13.227/23.640/36.756/7.635 ms
[03/26/24]seed@VM:~$
```

After making the appropriate rule to block access to this address, I am no longer able to reach this domain from this machine:

```
[03/26/24]seed@VM:~$ ping 23.44.192.164
PING 23.44.192.164 (23.44.192.164) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 23.44.192.164 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4101ms

[03/26/24]seed@VM:~$
```

With these ufw rules implemented, Machine A is no longer allowed to reach the walmart.com domain:

```
[03/26/24]seed@VM:~$ sudo ufw deny out from 10.0.2.4 to 52.44.165.2
26
Rule added
[03/26/24]seed@VM:~$ sudo ufw deny out from 10.0.2.4 to 52.55.180.7
0
Rule added
[03/26/24]seed@VM:~$ sudo status ufw
status: Unable to connect to Upstart: Failed to connect to socket /
com/ubuntu/upstart: Connection refused
[03/26/24]seed@VM:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
23                         ALLOW OUT   10.0.2.5
23                         DENY OUT    10.0.2.4
10.0.2.5 23                DENY OUT    10.0.2.4
52.44.165.226              DENY OUT    10.0.2.4
52.55.180.70               DENY OUT    10.0.2.4
```

Task 2: Implementing a Basic Firewall Program

This task primarily focuses on using **Loadable Kernal Module (LKM)** and **Netfilter** to modify the existing Linux kernel to create a firewall. When writing this program, I wanted to enable the following five rules:

- Deny 10.0.2.4 from telnetting to 10.0.2.5
- Deny 10.0.2.5 from telnetting to 10.0.2.4
- Deny 10.0.2.4 from reaching walmart.com
- Deny 10.0.2.5 from reaching target.com
- Allow 10.0.2.4 to ssh to 10.0.2.5

C: > Users > andre > Downloads > C andrewFilter (1).c

```c
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/inet.h>

static struct nf_hook_ops nfho;

// IP addresses to be filtered
unsigned int blocked_ip1 = 0x0402a8c0; // 10.0.2.4
unsigned int blocked_ip2 = 0x0502a8c0; // 10.0.2.5
unsigned int target_ip1 = 0x3b1930ac;  // walmart.com (23.207.49.224)
unsigned int target_ip2 = 0xd431c0ac;  // target.com (44.208.147.61)

static unsigned int hook_func(void *priv, struct sk_buff *skb, const struct nf_hook_state *state)
{
    struct iphdr *iph;
    struct tcphdr *tcph;

    if (!skb)
        return NF_ACCEPT;

    iph = ip_hdr(skb);
    if (!iph)
        return NF_ACCEPT;

    // Block telnet from 10.0.2.4 to 10.0.2.5
    if (iph->saddr == blocked_ip1 && iph->daddr == blocked_ip2 && iph->protocol == IPPROTO_TCP)
        return NF_DROP;

    // Block telnet from 10.0.2.5 to 10.0.2.4
    if (iph->saddr == blocked_ip2 && iph->daddr == blocked_ip1 && iph->protocol == IPPROTO_TCP)
        return NF_DROP;

```

```c
    // Block access to walmart.com from 10.0.2.4
    if (iph->saddr == blocked_ip1 && ntohl(iph->daddr) == target_ip1 && iph->protocol == IPPROTO_TCP)
        return NF_DROP;

    // Block access to target.com from 10.0.2.5
    if (iph->saddr == blocked_ip2 && ntohl(iph->daddr) == target_ip2 && iph->protocol == IPPROTO_TCP)
        return NF_DROP;

    // Allow SSH from 10.0.2.4 to 10.0.2.5
    if (iph->saddr == blocked_ip1 && iph->daddr == blocked_ip2 && iph->protocol == IPPROTO_TCP) {
        tcph = (struct tcphdr *)((__u32 *)iph + iph->ihl);
        if (ntohs(tcph->dest) == 22)
            return NF_ACCEPT;
    }

    return NF_ACCEPT;
}

static int __init init_func(void)
{
    nfho.hook = hook_func;
    nfho.pf = PF_INET;
    nfho.hooknum = NF_INET_PRE_ROUTING;
    nfho.priority = NF_IP_PRI_FIRST;

    nf_register_hook(&nfho);

    return 0;
}

static void __exit exit_func(void)
{
    nf_unregister_hook(&nfho);
}
```

```c

module_init(init_func);
module_exit(exit_func);

MODULE_LICENSE("GPL");

```

To compile this C code correctly, I needed to create a “Makefile” that looked like this:

```
obj-m += andrewFilter.o

all:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

Once having this “Makefile” and my C code in the same directory, I could use the “make” command. This command built a kernel module for my program titled “andrewFilter.ko”. I could then load this module with the “insmod” command.

```
[04/06/24]seed@VM:~/.../firewall$ sudo insmod andrewFilter.ko
[04/06/24]seed@VM:~/.../firewall$ ls
andrewFilter.c      andrewFilter.mod.o  modules.order   telnetFilter.c
andrewFilter.ko     andrewFilter.o      Module.symvers
andrewFilter.mod.c  Makefile            seedFilter.c
```

Once loading the module, my firewall rules were in place:

```
[04/06/24]seed@VM:~/.../firewall$ telnet 10.0.2.5
Trying 10.0.2.5...
^C
[04/06/24]seed@VM:~/.../firewall$ ping 59.25.49.59
PING 59.25.49.59 (59.25.49.59) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 59.25.49.59 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3108ms

[04/06/24]seed@VM:~/.../firewall$
```

Task 3: Evading Egress Filtering

The final task of this lab concerns **evading egress filtering**, which is commonly used by companies and schools to restrict access to certain websites or resource usage. We will be focused on using **SSH as a tunnel mechanism** to bypass this egress filter.

For this task, I will make adjustments to my firewall code from Task 2 that aim to restrict the following:

- **Block all outgoing traffic to external telnet servers**
- **Block all outgoing traffic to www.facebook.com**

My adjustments are as follows:

```
unsigned int telnet_server1 = 0x0a01a8c0;  // External telnet server 1 (192.168.1.10)
unsigned int telnet_server2 = 0x0a02a8c0;  // External telnet server 2 (192.168.2.10)
unsigned int facebook_ip1 = 0x2311431f;  // Facebook IP address 1 (31.13.67.35)
unsigned int facebook_ip2 = 0x23f0e595;  // Facebook IP address 2 (157.240.229.35)
unsigned int facebook_ip3 = 0x23f1f135;  // Facebook IP address 3 (157.240.241.35)
```

I found multiple addresses for www.facebook.com through dig and online searches. I added all of the current addresses I found to be safe.

```
// Block outgoing traffic to external telnet servers
if ((iph->saddr == blocked_ip1 || iph->saddr == blocked_ip2) &&
    (iph->daddr == telnet_server1 || iph->daddr == telnet_server2) &&
    iph->protocol == IPPROTO_TCP)
    return NF_DROP;

// Block outgoing traffic to Facebook.com
if ((iph->saddr == blocked_ip1 || iph->saddr == blocked_ip2) &&
    (ntohl(iph->daddr) == facebook_ip1 || ntohl(iph->daddr) == facebook_ip2 || ntohl
    (iph->daddr) == facebook_ip3) &&
    iph->protocol == IPPROTO_TCP)
    return NF_DROP;
```

Before attempting to bypass these rules, I wanted to first ensure they were in place:

```
[04/07/24]seed@VM:~/.../EgressFilter$ telnet 10.0.2.5
Trying 10.0.2.5...
```

```
[04/07/24]seed@VM:~/.../EgressFilter$ wget https://facebook.com
--2024-04-07 09:15:42--  https://facebook.com/
Resolving facebook.com (facebook.com)... 31.13.67.35, 2a03:2880:f12c:83:face:b00
c:0:25de
Connecting to facebook.com (facebook.com)|31.13.67.35|:443... ^C
[04/07/24]seed@VM:~/.../EgressFilter$ curl https://facebook.com
```

Once these rules were in place, my first job was to Telnet to Machine B (10.0.2.5) from Machine A (10.0.2.4) while the firewall program was running on Machine A. The strategy will be using SSH to tunnel and encrypt the the traffic so that the firewall program does not detect it. The Linux command I used to bypass was:

ssh -L 8000:10.0.2.5:23 seed@10.0.2.5

```
[04/07/24]seed@VM:~/.../EgressFilter$ ssh -L 8000:10.0.2.5:23 seed@10.0.2.5
telnet localhost 8000
^C
[04/07/24]seed@VM:~/.../EgressFilter$ ssh -L 8000:10.0.2.5:23 seed@10.0.2.5
The authenticity of host '10.0.2.5 (10.0.2.5)' can't be established.
ECDSA key fingerprint is SHA256:p1zAio6c1bI+8HDp5xa+eKRi561aFDaPE1/xq1eYzCI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.5' (ECDSA) to the list of known hosts.
seed@10.0.2.5'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.

Last login: Sun Apr  7 09:22:53 2024 from localhost
[04/07/24]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:47:e6:1e
          inet addr:10.0.2.5  Bcast:10.0.2.255  Mask:255.255.255.0
```

The final part of the task was to bypass the egress filter to access www.facebook.com. The first technique I tried was similar to the last, using a static IP in my ssh line:

ssh -L 8000:31.13.57.35:80 seed@10.0.2.5

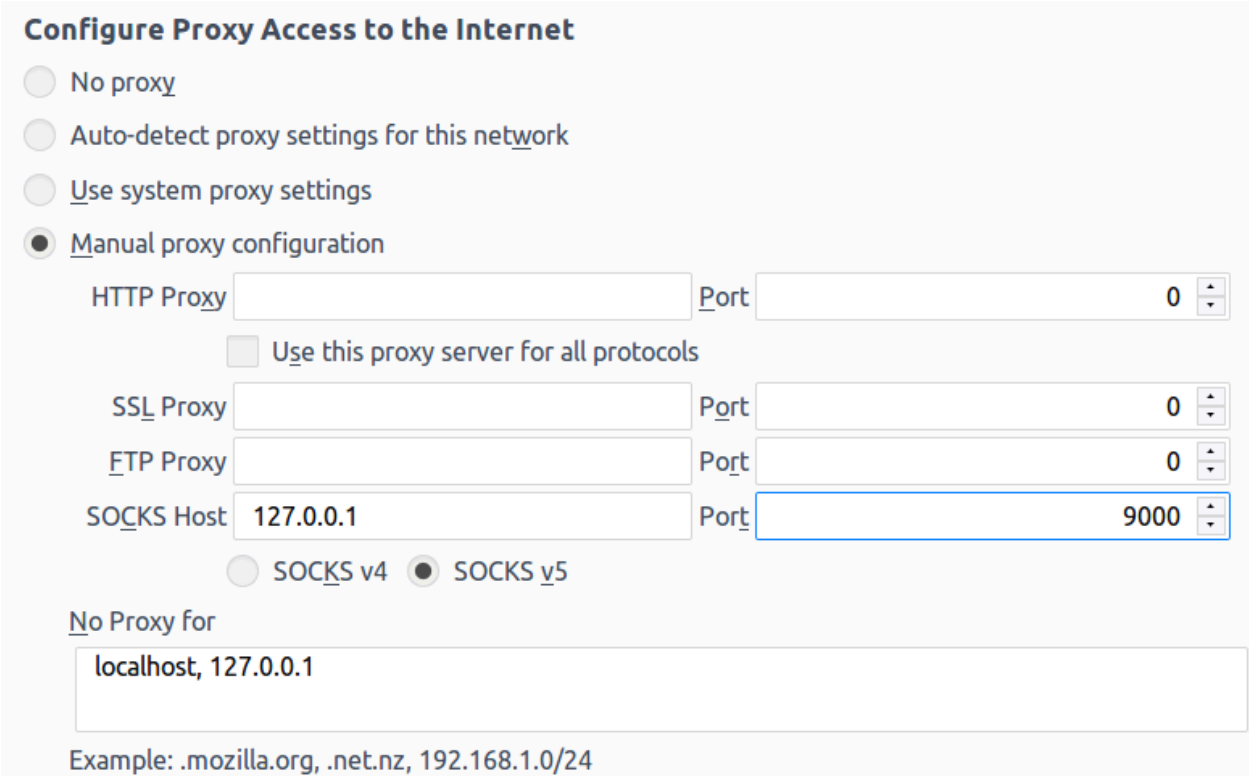
```
[04/07/24]seed@VM:~/.../EgressFilter$ dig +short facebook.com
157.240.14.35
[04/07/24]seed@VM:~/.../EgressFilter$ ssh -L 8000:31.13.67.35:80 seed@10.0.2.5
seed@10.0.2.5's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * 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.

Last login: Sun Apr  7 09:30:45 2024 from 10.0.2.4
[04/07/24]seed@VM:~$ wget https://www.facebook.com
--2024-04-07 09:32:36--  https://www.facebook.com/
Resolving www.facebook.com (www.facebook.com)... 31.13.67.35, 2a03:2880:f12c:183
:face:b00c:0:25de
Connecting to www.facebook.com (www.facebook.com)|31.13.67.35|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://www.facebook.com/unsupportedbrowser [following]
--2024-04-07 09:32:36--  https://www.facebook.com/unsupportedbrowser
Reusing existing connection to www.facebook.com:443.
```

While this worked, the instructions suggested a more generic approach, utilizing dynamic port forwarding. To support dynamic port forwarding, I needed to set up a SOCKS proxy in Firefox:




```
[04/07/24]seed@VM:~/.../EgressFilter$ ssh -D 9000 -C seed@10.0.2.5
seed@10.0.2.5'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.

Last login: Sun Apr  7 09:32:20 2024 from 10.0.2.4
[04/07/24]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:47:e6:1e
          inet addr:10.0.2.5  Bcast:10.0.2.255  Mask:255.255.255.0
```

With this ssh tunnel enabled, I am able to access www.facebook.com. Once the tunnel is closed, I am not able to reach the site and am presented with the following error message:

The proxy server is refusing connections

Firefox is configured to use a proxy server that is refusing connections.

- Check the proxy settings to make sure that they are correct.
- Contact your network administrator to make sure the proxy server is working.

Try Again