Department of Computer Science

California State University, Channel Islands

COMP-524: Security

Lab Report

Lab Number: 7
Lab Topic: Firewalls

Student Name: Sandipta Subir Khare
Student Major: Masters in Computer Science

Task 1:

With the help of iptables commands, we create simple iptables limitations to restrict outgoing and then inbound telnet connections using fine-grained soured and destination, as well as concrete port definitions to prevent telnet connection requests. BTW, in a real-world production system, I believe that restricting outgoing telnet connections is less wise than restricting access to incoming telnet requests.

```
[03/30/22]seed@VM:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[03/30/22]seed@VM:~$ ifconfig
Files     Link encap:Ethernet  HWaddr 08:00:27:e7:37:a6
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::ac7c:3d2a:8daf:79eb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:163 errors:0 dropped:0 overruns:0 frame:0
          TX packets:102 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:27051 (27.0 KB)  TX bytes:11867 (11.8 KB)
```

```
          RX bytes:42183 (42.1 KB)  TX bytes:42183 (42.1 KB)
[03/30/22]seed@VM:~$ sudo iptables -a INPUT -p tcp -s 10.0.2.5 --dport 23 -j DROP
iptables v1.6.0: unknown option "-a"
Try `iptables -h' or 'iptables --help' for more information.
[03/30/22]seed@VM:~$ sudo iptables -A INPUT -p tcp -s 10.0.2.5 --dport 23 -j DROP
```

```
[03/30/22]seed@VM:~$ telnet 10.0.2.4
Trying 10.0.2.4...
^C
```

```
[03/30/22]seed@VM:~$ sudo iptables -A OUTPUT -p tcp -d 10.0.2.5 --dport 23 -j DROP
[03/30/22]seed@VM:~$ telnet 10.0.2.5
Trying 10.0.2.5...
^C
[03/30/22]seed@VM:~$
```

Prohibiting access to specific websites is also an excellent way for system and devops administrators to prevent certain computers from connecting to the external network, which is important from a security standpoint. By digging the concrete ip address of particular web sites, we are restricting the connection of our system to our Professor's rabdolee.com website.

```
[03/30/22]seed@VM:~$ ping rabdolee.com
PING rabdolee.com (72.167.58.62) 56(84) bytes of data.
64 bytes from ip-72-167-58-62.ip.secureserver.net (72.167.58.62): icmp_seq=1 ttl=46 time=26.0 ms
64 bytes from ip-72-167-58-62.ip.secureserver.net (72.167.58.62): icmp_seq=2 ttl=46 time=26.1 ms
64 bytes from ip-72-167-58-62.ip.secureserver.net (72.167.58.62): icmp_seq=3 ttl=46 time=23.9 ms
^C
--- rabdolee.com ping statistics ---
```

```
[03/30/22]seed@VM:~$ sudo iptables -A OUTPUT -p tcp -d 72.167.58.62 -j DROP
[03/30/22]seed@VM:~$
```

```
[03/30/22]seed@VM:~$ sudo iptables -A OUTPUT -p tcp -d 72.167.58.62 -j DROP
[03/30/22]seed@VM:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  10.0.2.5             anywhere             tcp dpt:telnet

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  10.0.2.5             anywhere             tcp dpt:telnet
DROP       tcp  --  anywhere             ip-72-167-58-62.ip.secureserver.net  tcp dpt:telnet
DROP       tcp  --  anywhere             10.0.2.5             tcp dpt:telnet
DROP       tcp  --  anywhere             ip-72-167-58-62.ip.secureserver.net
[03/30/22]seed@VM:~$
```

BTW, with only one command, we can quickly erase all previously configured iptables rules.

```
[03/30/22]seed@VM:~$ sudo iptables -F
[03/30/22]seed@VM:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[03/30/22]seed@VM:~$
```

Another method of restricting machines to prevent certain network connections and actions is to create and use custom and third-party libraries such as Netfilter to restring connections more comprehensively at the Machine Kernel level, which is more similar to today's primitive prototypes such as Palo-Alto and Forcepoint. I loved how simple Netfilter libraries in C could be used to do this.

```
[03/30/22]seed@VM:~/Desktop$ make
make -C /lib/modules/4.8.0-36-generic/build M=/home/seed/Desktop modules
make[1]: Entering directory '/usr/src/linux-headers-4.8.0-36-generic'
  CC [M]  /home/seed/Desktop/telnetFilter.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/seed/Desktop/telnetFilter.mod.o
  LD [M]  /home/seed/Desktop/telnetFilter.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.8.0-36-generic'
[03/30/22]seed@VM:~/Desktop$ vim telnetFilter.c
```

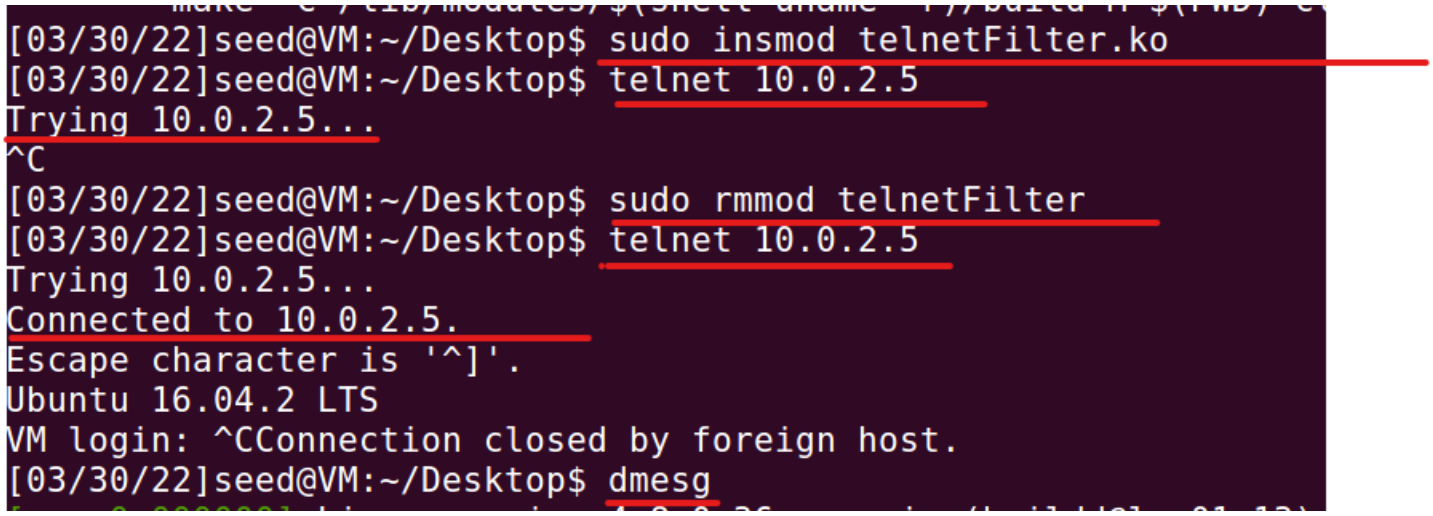
First, we'll create pre-compiler instructions for building an executable that will be used to generate a forged C executable later.

```
[03/30/22]seed@VM:~/Desktop$ cat Makefile
obj-m += telnetFilter.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
[03/30/22]seed@VM:~/Desktop$
```

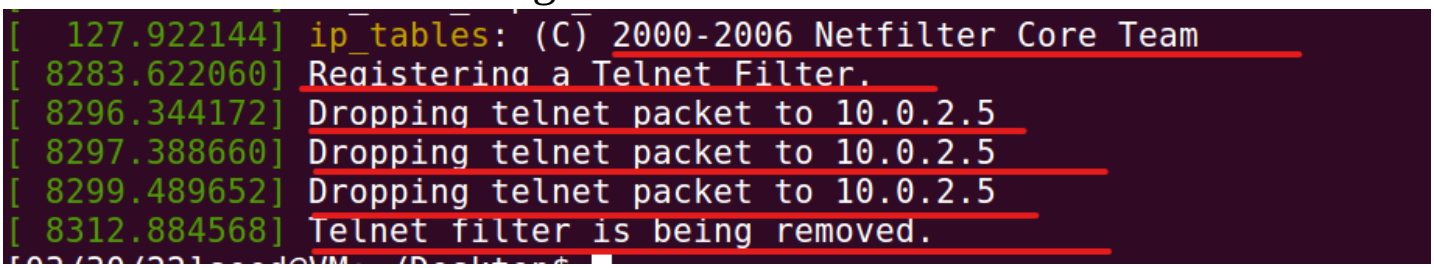
Then, using netfilter hooks, we create custom netfilter restrictions, which interact with the machine kernel to prevent specific connections from being made.

```c
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
#include <linux/tcp.h>
static struct  nf_hook_ops telnetFilterHook;
unsigned int telnetFilter(void *priv, struct sk_buff *skb, const struct nf_hook_state *state){
  struct iphdr *iph;
  struct tcphdr *tcph;
  iph = ip_hdr(skb);
  tcph = (void *)iph+iph->ihl*4;
  if (iph->protocol == IPPROTO_TCP && tcph->dest == htons(23)) {
        printk(KERN_INFO "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 {
    return NF_ACCEPT;
  }
}
int setUpFilter(void) {
        printk(KERN_INFO "Registering a Telnet Filter.\n");
        telnetFilterHook.hook = telnetFilter;
        telnetFilterHook.hooknum = NF_INET_POST_ROUTING;
        telnetFilterHook.pf = PF_INET;
        telnetFilterHook.priority = NF_IP_PRI_FIRST;
        nf_register_hook(&telnetFilterHook);
        return 0;
}
void removeFilter(void) {
        printk(KERN_INFO "Telnet filter is being removed.\n");
        nf_unregister_hook(&telnetFilterHook);
}
module_init(setUpFilter);
module_exit(removeFilter);
MODULE_LICENSE("GPL");
~
~
~
~
~
~
"telnetFilter.c" 39L, 1228C
```

We can easily ban Telnet connections to specific machines and erase this collection of kernel instructions by using the customized telnetFilter.ko created program.

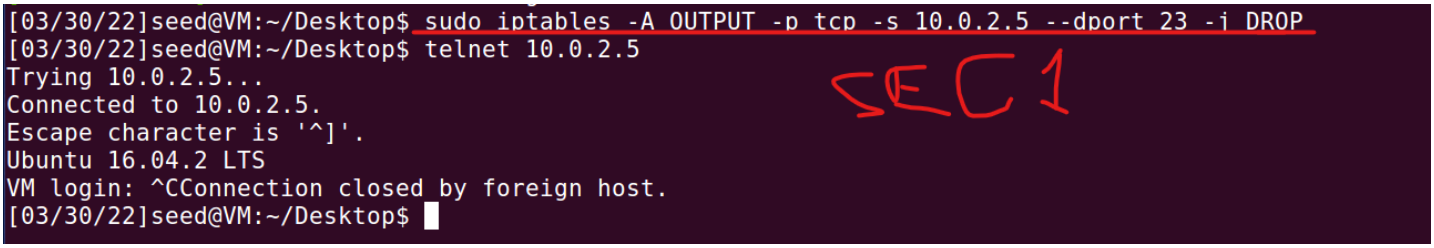


Dmesg displays us clear Kernel-related operations that occurred via telnetFilter on the Kernel logs.

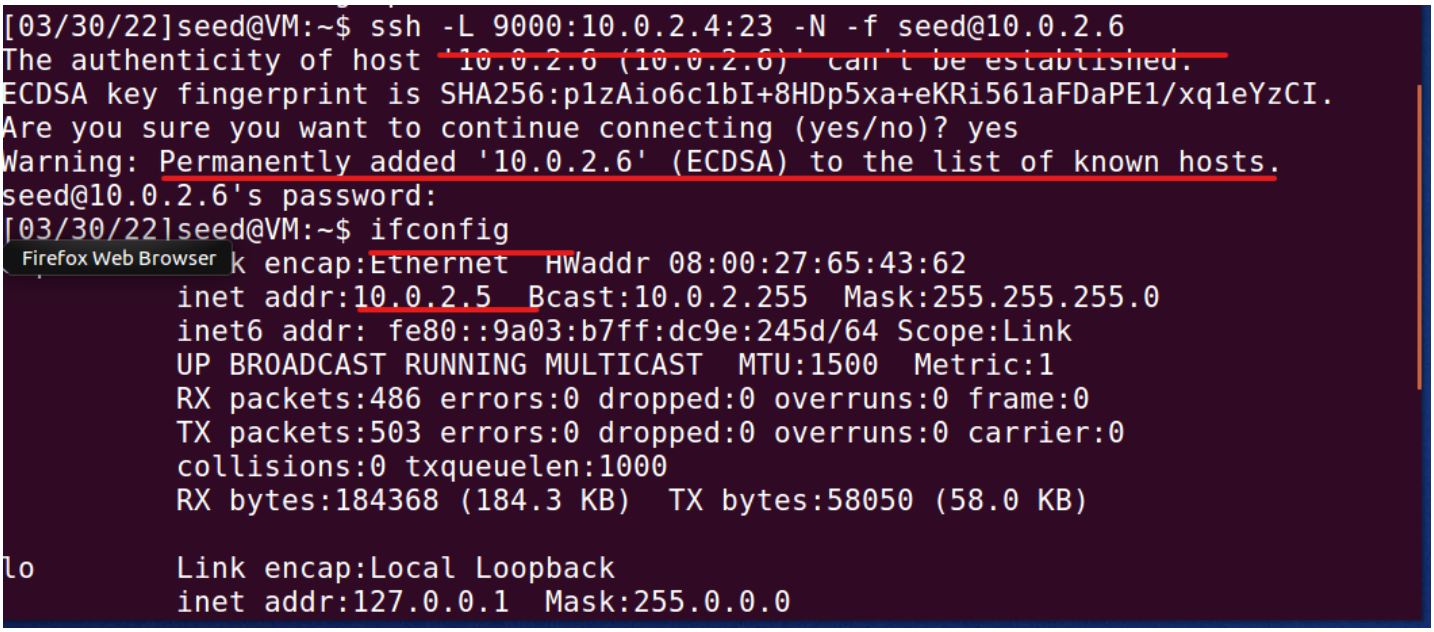


Evasion:

Of course, if we place certain restrictions on certain machines in our Production environment, some clever employees in our company can circumvent those restrictions by employing the Daisy Chaining exploitation vector in perimeter machines to reclaim prohibited resources that can only be obtained by using specific machines directly.



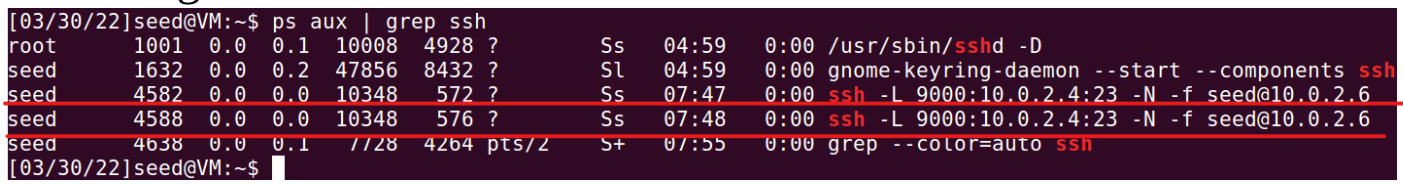
In the first security machine, we're imposing fine-grained outbound connection restrictions. However, by using an open port of 9000 on a third machine, the first machine can easily regain access to the second machine and request a telnet connection.



```
[03/30/22]seed@VM:~$ telnet localhost 9000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Files
.04.2 LTS
VM login: seed
Password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic

 * 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.
```

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

[03/30/22]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:e7:37:a6
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::ac7c:3d2a:8daf:79eb/64 Scope:Link
Files     UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6540 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1357 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8885330 (8.8 MB)  TX bytes:120625 (120.6 KB)

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:789 errors:0 dropped:0 overruns:0 frame:0
          TX packets:789 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:88962 (88.9 KB)  TX bytes:88962 (88.9 KB)

[03/30/22]seed@VM:~$
```

In the early 2010s, when Linux distributions had less sophisticated network segregation and infantile fine-grained customizations, this type of attack-evade vector was fairly popular. Exploiting such an attach in a current Production Network via proxy computers appears to be less practical nowadays, as it's nearly impossible to launch such an attack in a Production environment with strong Network Segregation and also Strong Hardening Policies.

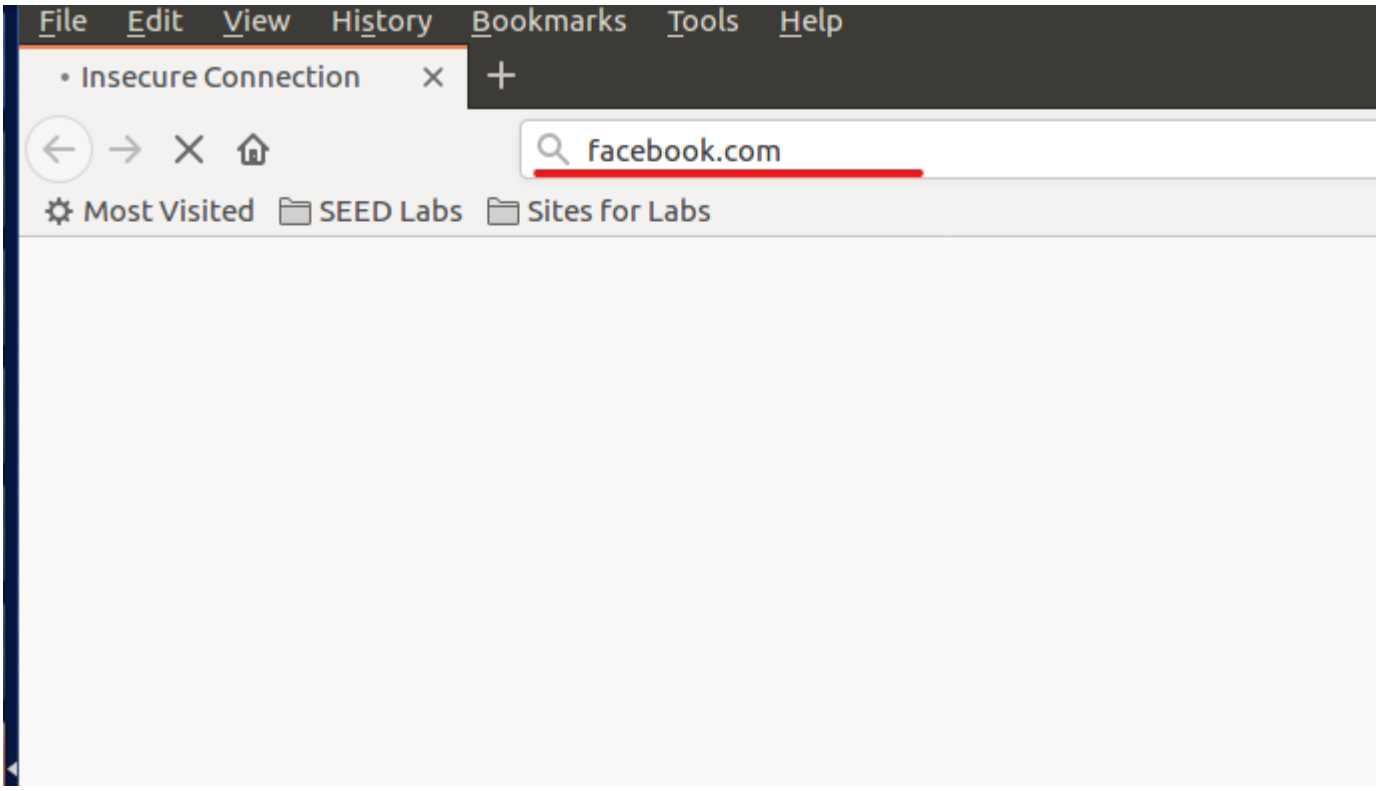


To exploit another machine, use the same daisy chaining or piggybacking method. In the network, it is also extremely feasible to access certain web resources that are blocked in our machine if certain vulnerabilities are intentionally left in our machines for igniting and employing proximising in between machines for our goals.

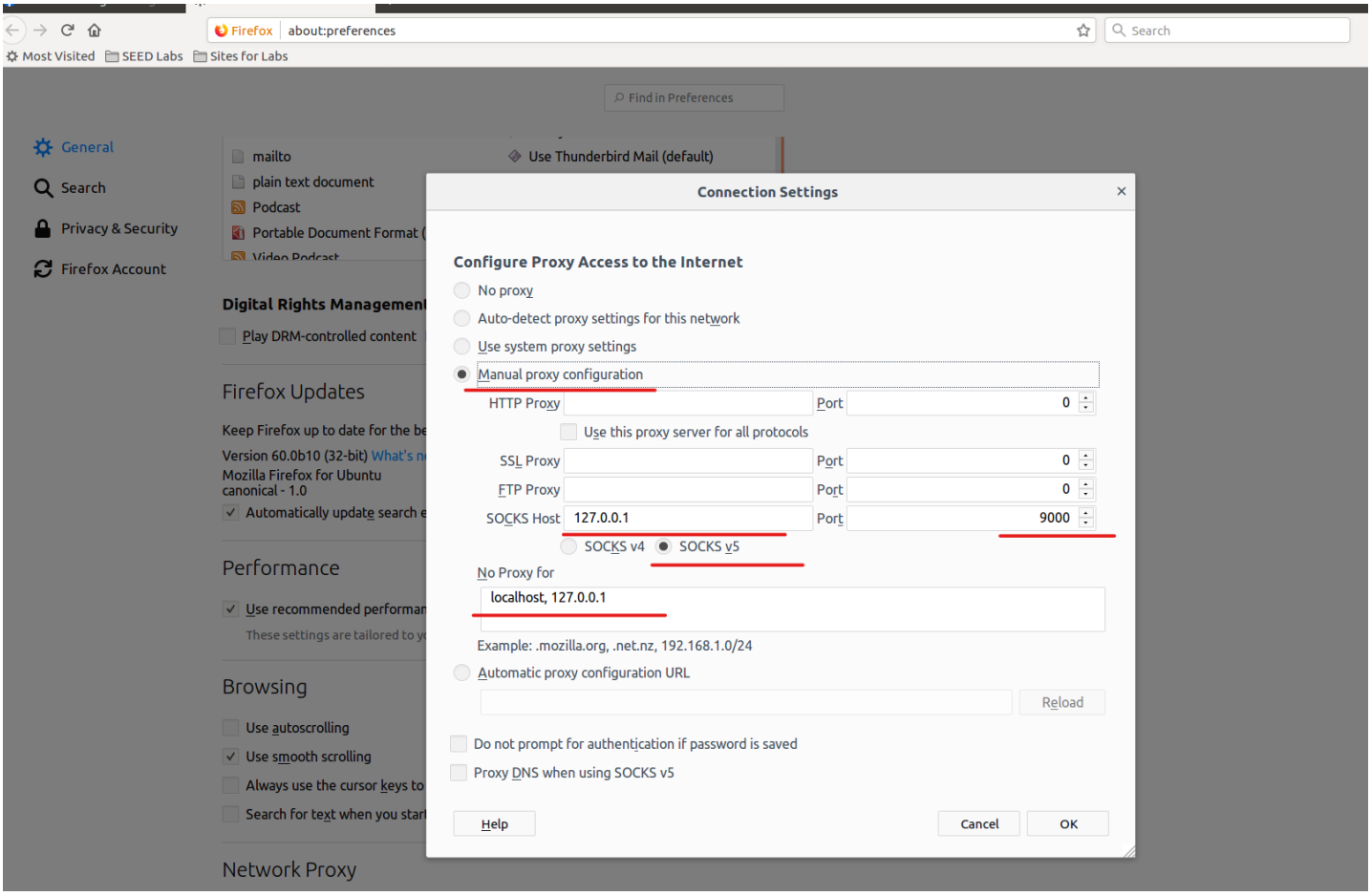
```
[03/30/22]seed@VM:~/Desktop$ dig facebook.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> facebook.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36236
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;facebook.com.                  IN      A

;; ANSWER SECTION:
facebook.com.           120     IN      A       31.13.70.36

;; Query time: 14 msec
;; SERVER: 127.0.1.1#53(127.0.1.1)
;; WHEN: Wed Mar 30 07:58:47 EDT 2022
;; MSG SIZE  rcvd: 57

[03/30/22]seed@VM:~/Desktop$
```

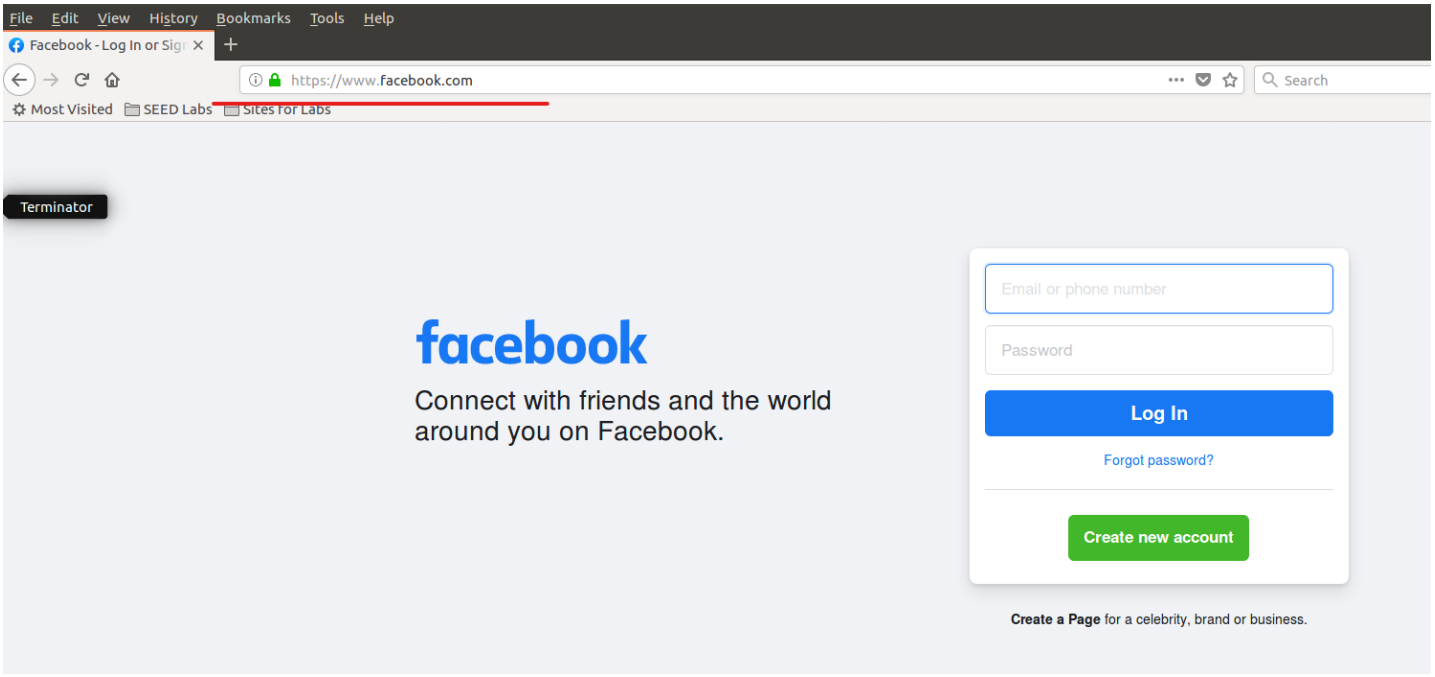
```
[03/30/22]seed@VM:~/Desktop$ sudo iptables -A OUTPUT -p tcp -d 31.13.70.36 -j DROP
[03/30/22]seed@VM:~/Desktop$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  10.0.2.5             anywhere             tcp dpt:telnet
DROP       tcp  --  anywhere             edge-star-mini-shv-01-lax3.facebook.com
```

System Settings


File Edit View History Bookmarks Tools Help
• Insecure Connection
facebook.com
Most Visited
SEED Labs
Sites for Labs

```
[03/30/22]seed@VM:~$ ssh -D 9000 -C seed@10.0.2.6
seed@10.0.2.6'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.

[03/30/22]seed@VM:~$
```




File Edit View History Bookmarks Tools Help
Facebook - Log In or Sig
https://www.facebook.com
Search
Most Visited
SEED Labs
Sites for Labs
Terminator
facebook
Connect with friends and the world around you on Facebook.
Email or phone number
Password
Log In
Forgot password?
Create new account
Create a Page for a celebrity, brand or business.