Kristianstad University
SE-291 88 Kristianstad
+46 44-250 30 00
www.hkr.se

Computer security, 7.5 credits
Semester Year e.g. Autumn Semester 2021
Faculty of Natural Science

Lab 3

Sandra Kaljula

Content

Introduction **3**
Method **4**
Results **5**
Discussion **12**
Appendix A ..... Fel! Bokmärket är inte definierat.

Introduction

This report is about lab 3 in the computer security course. Lab 3 is about establishing and making rules for the incoming and outcoming packets in the Linux firewall.

Method

For the labs I used my laptop that already had Linux Ubuntu on it. Then I installed the required packages for the tasks and moved on to learning about them from the internet. Finally, once everything was clear I moved on to playing around with the firewall and using Nmap on my solutions.

Results

Step 1 The environment

I already had an existing Linux Ubuntu dual boot on my laptop, so I decided to use that.

Step 2 Installing iptables

To install iptables in the terminal I wrote: “sudo apt-get install iptables”, but it seemed like it already existed.

sudo - superuser do (give administrator right).

apt-get - uses the etc/apt/sources.list to handle packages (install, remove, update).

iptables - utility in the firewall with policies for allowing or blocking traffic[1].

Step 3 - Packet filter firewall settings

Chains:

INPUT - incoming traffic

FORWARD - forward traffic (only for routers, my computer does not forward any packages)

OUTPUT - outgoing traffic

ACCEPT - accepts traffic
REJECT - rejects the traffic, but sends an error message to the sender
DROP - drops the traffic, no error message, just timeout at the sender side

The task is to block all the other traffic but the web traffic. For that we need to first block all the connections in, out and forward. We will leave the forward to always be DROP since my computer is not a router and not supposed to forward anything. We will in the next steps add rules to when to accept specific traffic.

-h or **-help** (more information)

-P or **- -policy** (INPUT/FORWARD/OUTPUT)

sudo iptables -P INPUT DROP

sudo iptables -P FORWARD DROP

sudo iptables -P OUTPUT DROP

-F (clear the changes made)

-L (list all rules)

-A or **- - append** (Appends to the chain)

-p (specifies a protocol)

-m (additional match) **multiport** (multiple ports), **-m conntrack** (store connection tracking information)

- -dports X,Y (destination ports X,Y)

- -sports X,Y (source ports X,Y)

-j or **- -jump** (jumps to the target, if valid: ACCEPT/ DELETE/ RETURN/ QUEUE) [2]

Then we need to add policies for accepting HTTP and HTTPS both inwards and outwards. HTTP port is 80 and HTTP is 443 and they use tcp.

sudo iptables -A OUTPUT -p tcp -m multiport --dports 80,443 -j ACCEPT

sudo iptables -A INPUT -p tcp -m multiport --sports 80,443 -j ACCEPT

We also need to have the DNS to get access, so we can just write in an URL address. DNS uses port 53 and udp.

sudo iptables -A OUTPUT -p udp --dport 53 -j ACCEPT

sudo iptables -A INPUT -p udp --sport 53 -j ACCEPT

Lastly, since we are using Ubuntu, we need to allow the localhost DNS connection in and out through port 53.

lo (localhost)

sudo iptables -A OUTPUT -o lo -p udp --sport 53 -j ACCEPT

sudo iptables -A INPUT -i lo -p udp --dport 53 -j ACCEPT

Figure 1 and 2 show the results from step 3 and saving them.

```
sandra@sandra-VivoBook-ASUSLaptop-X512DA-F512DA:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             multiport sports http,https
ACCEPT     udp  --  anywhere             anywhere             udp spt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             multiport dports http,https
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp spt:domain
```

Figur 1 List over the rules after step 3.

```
sandra@sandra-VivoBook-ASUSLaptop-X512DA-F512DA:~$ sudo /sbin/iptables-save
# Generated by iptables-save v1.8.4 on Fri Dec 17 11:41:14 2021
*filter
:INPUT DROP [69:6904]
:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]
-A INPUT -p tcp -m multiport --sports 80,443 -j ACCEPT
-A INPUT -p udp -m udp --sport 53 -j ACCEPT
-A INPUT -i lo -p udp -m udp --dport 53 -j ACCEPT
-A OUTPUT -p tcp -m multiport --dports 80,443 -j ACCEPT
-A OUTPUT -p udp -m udp --dport 53 -j ACCEPT
-A OUTPUT -o lo -p udp -m udp --sport 53 -j ACCEPT
COMMIT
# Completed on Fri Dec 17 11:41:14 2021
```

Figur 2 List over the rules after saving step 3.

This seems to be correct since I can visit all the websites.

Step 4 - Stateful firewall settings

In this step we look at the state of the connection, if the connection is new, established or related.

- -ctstate (NEW/ESTABLISHED/RELATED) (sets the states)

We still have a base case policy to DROP all the connections at the end of the chain.

This time we are going to track the connection and store it.

sudo iptables -A OUTPUT -p tcp -m multiport --dports 80,443 -m conntrack --ctstate **NEW,ESTABLISHED** -j ACCEPT

Only accept the established incoming connections.

sudo iptables -A INPUT -p tcp -m multiport --sports 80,443 -m conntrack --ctstate **ESTABLISHED** -j ACCEPT[3]

We also need to have the DNS to get access, so we can just write in an URL address. DNS uses port 53 and udp.

sudo iptables -A OUTPUT -p udp --dport 53 -m conntrack --ctstate **NEW,ESTABLISHED** -j ACCEPT

sudo iptables -A INPUT -p udp --sport 53 -m conntrack --ctstate **ESTABLISHED** -j ACCEPT[3]

Lastly, since we are using Ubuntu, we need to allow the localhost DNS connection in and out through port 53. I only left the OUTPUT for localhost to be ESTABLISHED because this could protect from a spoofing attack where someone else pretends to be the localhost.

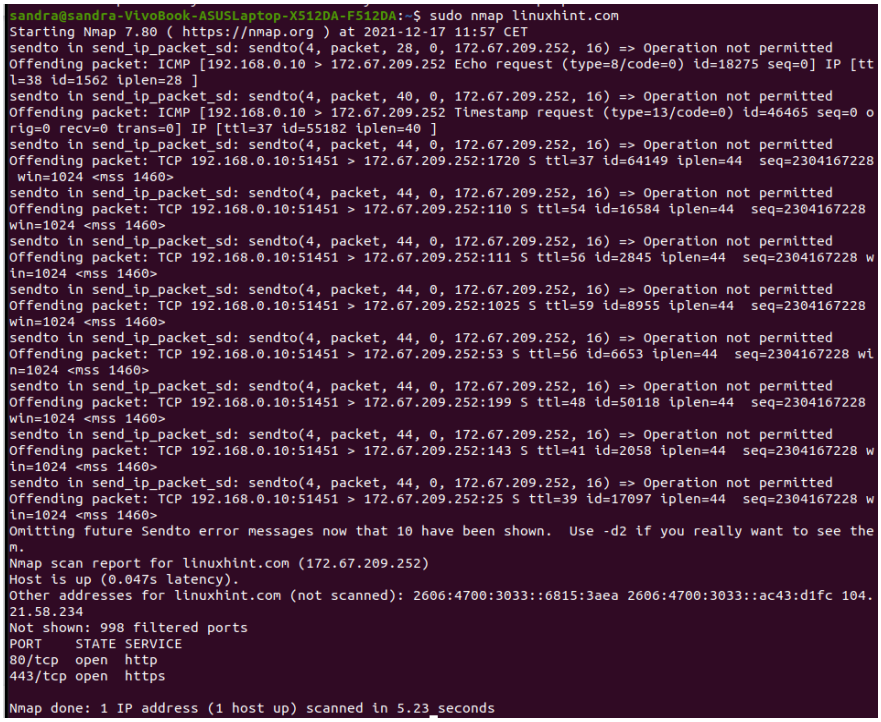
lo (localhost)

sudo iptables -A OUTPUT -o lo -p udp --sport 53 -m conntrack --ctstate **ESTABLISHED** -j ACCEPT

sudo iptables -A INPUT -i lo -p udp --dport 53 -m conntrack --ctstate **NEW,ESTABLISHED** -j ACCEPT [3]

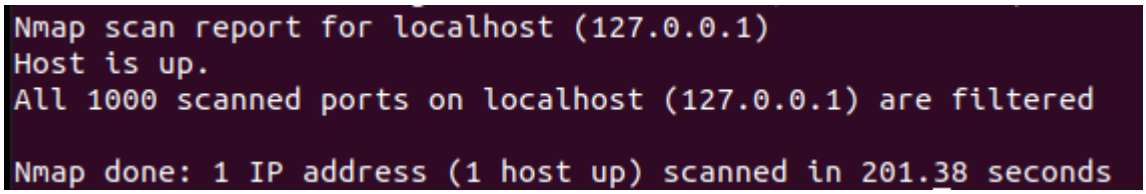
This seems to be correct since I can visit all the websites. I also decided to use Nmap. I installed it through the console once again with the command: “sudo apt install Nmap”. The other ports but 80 and 443 could not be accessed. Thus, the error messages.

I decided to first check one of the most used examples, which was scanning the website of linuxhint.com. See figure 3.



Figur 3 Nmap on linuxhint.com after completing step 4.

I also decided to do a scan with Nmap on my localhost and I got a response that all the ports were filtered and the operations the Nmap wanted to do were not permitted.



Figur 4 Nmap on my localhost.

Step 5

And finally, save the changes in the iptables after step 4 seen in figure 5. And print out the detailed rules in figure 6 after step 4.

-v (verbose mode: detailed information about)

sudo /sbin/iptables-save

```
sandra@sandra-VivoBook-ASUSLaptop-X512DA-F512DA:~$ sudo /sbin/iptables-save
[sudo] password for sandra:
# Generated by iptables-save v1.8.4 on Fri Dec 17 10:58:41 2021
*filter
:INPUT DROP [6135:622199]
:FORWARD DROP [0:0]
:OUTPUT DROP [91:26875]
-A INPUT -p tcp -m multiport --sports 80,443 -m conntrack --ctstate ESTABLISHED -j ACCEPT
-A INPUT -p udp -m udp --sport 53 -m conntrack --ctstate ESTABLISHED -j ACCEPT
-A INPUT -i lo -p udp -m udp --dport 53 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
-A OUTPUT -p tcp -m multiport --dports 80,443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
-A OUTPUT -p udp -m udp --dport 53 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
-A OUTPUT -o lo -p udp -m udp --sport 53 -m conntrack --ctstate ESTABLISHED -j ACCEPT
COMMIT
# Completed on Fri Dec 17 10:58:41 2021
```

Figur 5 List over the saved rules after task 4.

sudo iptables -L -v

```
sandra@sandra-VivoBook-ASUSLaptop-X512DA-F512DA:~$ sudo iptables -L -v
Chain INPUT (policy DROP 2 packets, 202 bytes)
 pkts bytes target     prot opt in     out     source               destination
   52 31782 ACCEPT     tcp  --  any    any     anywhere             anywhere             multiport sports http,https ctstate ESTABLISHED
   34  4340 ACCEPT     udp  --  any    any     anywhere             anywhere             udp spt:domain ctstate ESTABLISHED
   14  1056 ACCEPT     udp  --  lo     any     anywhere             anywhere             udp dpt:domain ctstate NEW,ESTABLISHED

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
   50  9077 ACCEPT     tcp  --  any    any     anywhere             anywhere             multiport dports http,https ctstate NEW,ESTABLISHED
 1596  122K ACCEPT     udp  --  any    any     anywhere             anywhere             udp dpt:domain ctstate NEW,ESTABLISHED
   14  1486 ACCEPT     udp  --  any    lo      anywhere             anywhere             udp spt:domain ctstate ESTABLISHED
```

Figur 6 List over the rules after task 4.

Conclusion

I think these labs were very good to get an insight to firewall methods and computer security overall. I encountered some problems. I had a bunch of confusion around the syntax at first. Therefore, I used the -help method for iptables and decided to write down all the useful information. This made me understand everything much clearer. I also forgot to take screenshots of step 3, so I had to go back to it. But at the end I got it all figured out. I had not used the command prompt a lot neither had I used Linux for that before. But I am now glad that I can now implement a firewall and think of computer security a little bit out of the box.

References

[1] Brown K. The beginner's guide to iptables, the Linux firewall[Internet]. HowToGeek. [updated: 2020-08-27; cited date: 2021-12-17] Available from: https://www.howtogeek.com/177621/the-beginners-guide-to-iptables-the-linux-firewall/

[2] Red Hat Enterprise. Linux 4: Reference Guide. iptables[Internet]. Red Hat Enterprise[cited date: 2021-12-17]. Available from: https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-en-4/s1-iptables-options.html

[3] Drist. Why must loopback traffic be authorized using iptables to get web access?[Internet]. StackExchange[updated: 2016-07-14; cited date 2021-12-17] https://unix.stackexchange.com/questions/81107/why-must-loopback-traffic-be-authorized-using-iptables-to-get-web-access