Lab – Iptables The Linux Firewall

Introduction

Iptables is a packet filter-based implementation of the Linux kernel firewall, It defines tables that contain a chain of rules that specify how packets should be treated. The hierarchy is iptables -->tables --> chains --> rules. There may be built-in tables and chains as well as user-defined ones.

There are three independent tables (the presence of a table depends on the kernel configuration options): **filter, nat and mangle**. You can specify the table to be used through the **-t** option. This lab is based on the "filter" table, you can try other tables as well.

Required resources

We are going to use all three VMs, the following demonstration is only for practice purpose.

Task

Configure iptables firewall rules on Cyber-client

Step 1 Check iptables is installed in your system

a. We already have iptables pre-installed. You can use iptables -V to check the package version of iptables. The capital V is for version, the lower-case v is for verbose.

```
cybersec-client@ubuntu:~$ iptables -V
iptables v1.4.21
```

b. If you need to update/install iptables, just retrieve the iptables package:

   sudo apt-get install iptables

c. Use iptables -h to check the usage of iptables.

```
cybersec-client@ubuntu: ~
cybersec-client@ubuntu:~$ iptables -h
iptables v1.4.21

Usage: iptables -[ACD] chain rule-specification [options]
       iptables -I chain [rulenum] rule-specification [options]
       iptables -R chain rulenum rule-specification [options]
       iptables -D chain rulenum [options]
       iptables -[LS] [chain [rulenum]] [options]
       iptables -[FZ] [chain] [options]
       iptables -[NX] chain
       iptables -E old-chain-name new-chain-name
       iptables -P chain target [options]
       iptables -h (print this help information)
```

Step 2 Check the build-in chain

The **“filter”** table has three built-in chain: input, forward, and output.

Input – This chain is used to control the behaviour for incoming connections. For example, if a user attempts to SSH into your PC/server, iptables will attempt to match the IP address and port to a rule in the input chain.

Forward – This chain is used for incoming connections that aren’t actually being delivered locally. Think of a router – data is always being sent to it but rarely actually destined for the router itself; the data is just forwarded to its target. Unless you’re doing some kind of routing, NATing, or something else on your system that requires forwarding, you won’t even use this chain.

Output – This chain is used for outgoing connections.

We can using sudo iptables -L to check the current iptables chains, we can add -v (for verbose) to check the number of packer be accepted or denied.

```
cybersec-client@ubuntu:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 14 packets, 1176 bytes)
 pkts bytes target     prot opt in     out     source               destination


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


Chain OUTPUT (policy ACCEPT 14 packets, 1176 bytes)
 pkts bytes target     prot opt in     out     source               destination

cybersec-client@ubuntu:~$
```

“ACCEPT” highlighted in the screenshot indicates the default behaviour of the chains, it will be used to process traffic when there is no match in the existing rules.

Step 3 Change the default behaviour of iptables

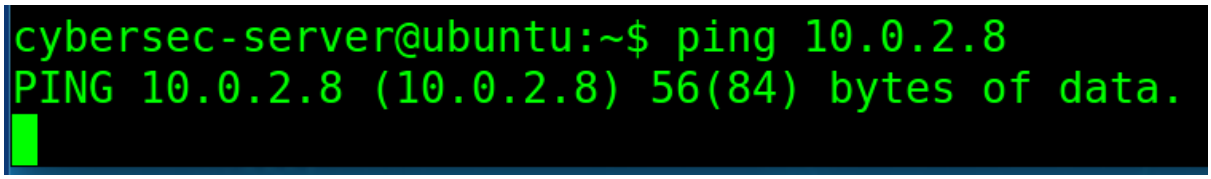
a. Drop all traffic
   We are going to change the behaviour of the chain to drop all traffic.

```
cybersec-client@ubuntu:~$ sudo iptables --policy INPUT DROP
cybersec-client@ubuntu:~$ sudo iptables --policy OUTPUT DROP
cybersec-client@ubuntu:~$ sudo iptables --policy FORWARD DROP
cybersec-client@ubuntu:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
cybersec-client@ubuntu:~$
```

The ping from CyberServer to CyberClient failed after we changed the default behaviour of iptables to “DROP”



Note: a lot of protocols will require two-way communication, so both the input and output chains will need to be configured properly.

b. Permit the ping from server to client.
The three most basic and commonly used action is:

Accept – Allow the connection.

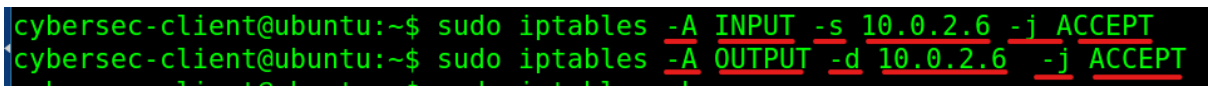
Drop – Drop the connection with no error message. (Refer to the above ping, there is no feedback message.)

Reject – Don’t allow the connection but send back an error message.

iptables -A (for append)to append rules to the existing chain. iptables behave like ACL, starts at the top of its list and goes through each rule until it finds one that it matches. If you need to insert a rule above another, you can use iptables -I [chain] [number] to specify the number it should be in the list, the -I is for insert. You can combine -A option with other options, such as:

- **-i** (interface) — the network interface whose traffic you want to filter, such as eth0, lo, ppp0, etc.
- **-p** (protocol) — the network protocol where your filtering process takes place. It can be either tcp, udp, udplite, icmp, sctp, icmpv6, and so on. Alternatively, you can type all to choose every protocol.
- **-s** (source) — the address from which traffic comes from. You can add a hostname or IP address.
- **--dport** (destination port) — the destination port number of a protocol, such as 22 (SSH), 443 (https), etc.
- **-j** (target) — the target name (ACCEPT, DROP, RETURN). You need to insert this every time you make a new rule.

Now we are going to add rules to permit the connection from CyberServer, see the screenshot below:



After add this rule, the ping from server is successful, but the ping from attacker still fail.

```
cybersec-server@ubuntu:~$ ping 10.0.2.8
PING 10.0.2.8 (10.0.2.8) 56(84) bytes of data.
64 bytes from 10.0.2.8: icmp_seq=1 ttl=64 time=0.638 ms
64 bytes from 10.0.2.8: icmp_seq=2 ttl=64 time=0.655 ms
64 bytes from 10.0.2.8: icmp_seq=3 ttl=64 time=7.53 ms
64 bytes from 10.0.2.8: icmp_seq=4 ttl=64 time=0.978 ms
```

```
cybersec-attacker@ubuntu: ~
cybersec-attacker@ubuntu:~$ ping 10.0.2.8
PING 10.0.2.8 (10.0.2.8) 56(84) bytes of data.
```

Now check the rules again.

```
cybersec-client@ubuntu:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  10.0.2.6            anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere            10.0.2.6
```

c. Let's permit SSH connections from the Cybersec-Attacker to Cybersec-Client(Only for demonstration purpose).

   Check SSH from Cybersec-Attacker to Cybersec-Client before we add the rules.

```
cybersec-attacker@ubuntu:~$ ssh cybersec-client@10.0.2.8


^C
```

   Add the rules in Cybersec-Client to permit the SSH connection.

```
cybersec-client@ubuntu:~$ sudo iptables -A INPUT -p tcp --dport ssh -s 10.0.2.7 -j ACCEPT
cybersec-client@ubuntu:~$ sudo iptables -A OUTPUT -p tcp --sport ssh -d 10.0.2.7 -j ACCEPT
```

We have added rules to the input and output chains in the above configuration. What if we only want SSH coming into our system, we can use connection states which is similar to "established" keyword in ACL. Let allow SSH connection from Attacker, but not to attacker VM.

Use -D to remove the previous rules:

```
cybersec-client@ubuntu:~$ sudo iptables -D OUTPUT -p tcp --sport ssh -d 10.0.2.7 -j ACCEPT
cybersec-client@ubuntu:~$ sudo iptables -D INPUT -p tcp --dport ssh -s 10.0.2.7 -j ACCEPT
```

Reconfigure the rules:

cybersec-client@ubuntu:~$ sudo iptables -A INPUT -p tcp --dport ssh -s 10.0.2.7 -m state --state NEW,ESTABLISHED -j ACCEPT

cybersec-client@ubuntu:~$ sudo iptables -A OUTPUT -p tcp --sport ssh -d 10.0.2.7 -m state --state NEW,ESTABLISHED -j ACCEPT

Refer to the link for more match options.

```
cybersec-client@ubuntu:~$ sudo iptables -A INPUT -p tcp --dport ssh -s 10.0.2.7
-m state --state NEW,ESTABLISHED -j ACCEPT
cybersec-client@ubuntu:~$ sudo iptables -A OUTPUT -p tcp --sport ssh -d 10.0.2.7
 -m state --state NEW,ESTABLISHED -j ACCEPT
```

Verify SSH connection from attacker to client and vice versa.

```
cybersec-attacker@ubuntu:~$ ssh cybersec-client@10.0.2.8
cybersec-client@10.0.2.8's password:
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 4.2.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

381 packages can be updated.
309 updates are security updates.

Last login: Thu Jun 30 02:22:14 2022 from 10.0.2.7
cybersec-client@ubuntu:~$
```

```
cybersec-client@ubuntu:~$ ssh cybersec-attacker@10.0.2.7


^C
cybersec-client@ubuntu:~$
```

You can add -v to check the number of packeted been accepted or dropped.

Step 4 delete rules and saving changes

To delete/flush all rules in chains, we can use -F, the command sudo iptables -F will erase all current rules. To delete a specific rule, we can use -D option. We can see all the available rules with numbers by entering sudo iptables -L --line-numbers

```
cybersec-client@ubuntu:~$ sudo iptables -L --line-numbers
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     all  --  10.0.2.6             anywhere
2    ACCEPT     tcp  --  10.0.2.7             anywhere             tcp dpt:ssh state NEW,ESTABLISHED

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             10.0.2.6
2    ACCEPT     tcp  --  anywhere             10.0.2.7             tcp spt:ssh state NEW,ESTABLISHED
```

To delete a rule, insert the corresponding chain and the number from the list. For example, to delete number 2 of the INPUT rule.

```
cybersec-client@ubuntu:~$ sudo iptables -D INPUT 2
cybersec-client@ubuntu:~$ sudo iptables -L --line-numbers
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     all  --  10.0.2.6             anywhere
```

The change we did will be removed once we restart the VM, to save the change, use /sbin/iptables-save. We are not going to save the change for other labs.

Challenge:

Configure iptables on Cybersec-server to permit HTTP and HTTPS request, drop all other traffic.