# Lab-7-1: Network Security – Firewall

In today’s lab, we are investigating how firewalls work and how we can use firewalls to improve the security of our server: Metasploitable2.

In today’s lab, we need to **start the following two virtual machines**:

1. Kali Linux
2. Metasploitable2

**Exercise-7-1-1: Initial port scan of Metasploitable2**

Make sure you are in Kali Linux. Open a terminal and run a quick port scan. We are going to use this information to compare to once we have set up a basic firewall.

Use the following command to perform a port scan on the first 200 ports:

**root@kali**:**~**# nmap –sV –p 1-1000 192.168.19.10 –oN portscan1.txt

The –oN argument save the port scan to a file named portscan1.txt. We can use this file to compare our results to after we have set up a functional firewall.

**Exercise-7-1-2: Setting up SSH**

So far, in the Security labs we have directly accessed the Metasploitable2 system directly. However, it would make sense that we practice some remote system administration. Today we are going to use SSH to establish a remote terminal to configure our firewall. Make sure you are logged into your Kali Linux machine and start an SSH session with Metasploitable2.

Start a new terminal window using: Ctrl + Shift + T

Now, use the following command to connect to Metasploitable2 using SSH:

**root@kali**:**~**# ssh msfadmin@192.168.19.10

When logging into SSH for the first time you will receive a message similar to below:

The authenticity of host '192.168.19.10 (192.168.19.10)' can't be established.

RSA key fingerprint is SHA256:BQHm5EoHX9GCiOLuVscegPXLQOsuPs+E9d/rrJB84rk.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '192.168.19.10' (RSA) to the list of known hosts.

You need to accept the RSA public key by entering “yes”. This key is stored on the Kali Linux machine for any subsequent SSH sessions. Login to the SSH session using the password: msfadmin

Please note that we use the SSH command, followed by the user and then the target IP address. If we did not specify a user (in this case msfadmin) the SSH command would try to log us in as the root user.

Switch back to the first Terminal tab, and view the file of SSH connections using the following command:

**root@kali**:**~**# cat ~/.ssh/known\_hosts

**Q1.** Document the SSH public key stored in this file.

|1|ngRZ26Wr+w04DXpPCmfxq1p/6+c=|oIcoOpITzFg4HuJa5IN6+St7+7o= ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAstqnuFMBOZvO3WTEjP4TUdjgWkIVNdTq6kboEDjteOfc65TlI7sRvQBwqAhQjeeyyIk8T55gMDkOD0akSlSXvLDcmcdYfxeIF0ZSuT+nkRhij7XSSA/Oc5QSk3sJ/SInfb78e3anbRHpmkJcVgETJ5WhKObUNf1AKZW++4Xlc63M4KI5cjvMMIPEVOyR3AKmI78Fo3HJjYucg87JjLeC66I7+dlEYX6zT8i1XYwa/L1vZ3qSJISGVu8kRPikMv/cNSvki4j+qDYyZ2E5497W87+Ed46/8P42LNGoOV8OcX/ro6pAcbEPUdUEfkJrqi2YXbhvwIJ0gFMb6wfe5cnQew==

**Q2.** What does the using the servers public key achieve for our SSH connection?

Allows us to connect to the VM remotely. The connection is secure as only I have the private key used for this public key.

**Exercise-7-1-3: Setting up iptables**

So far, in the Security paper we have exploited and broken the security of the Metasploitable2 server. Today we are going to fix some security issues by implementing a simple firewall. To achieve this we are going to use the iptables command to setup the firewall.

First let us see what iptables configuration in configured on our system. Switch to the SSH connection tab in your terminal, and try the following command:

**msfadmin@metasploitable:~$** sudo iptables -L

As you can see from the output, we have no rules and nothing configured in our iptables firewall. Since we are connected using SSH we should make sure we explicitly add a rule for SSH so we are not locked out of our system.

To achieve this, we need to specify that any existing connections should be maintained:

**msfadmin@metasploitable:~$** sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

This command looks highly complex, but looking at each component makes it much easier to understand.

|  |  |
| --- | --- |
| -A INPUT | Append rule to end of chain  And INPUT means the rule if for incoming traffic |
| -m conntrack | This provides stateful filtering in our firewall. Conntrack is basically “connection tracking” |
| --ctstate ESTABLISHED,RELATED | This sections states that we will allow any already established or related connections (like our SSH session that we have “established”) |
| -j ACCEPT | If any packets match the above, we accept them! |

This rule is used only to allow active connections to be kept. Since we are using SSH to connect to Metasploitable2 this is exceptionally important! Check the rule you added is there by using:

**msfadmin@metasploitable:~$** sudo iptables -L

NOTE: If you make any mistakes during this lab, you can flush (delete) any firewall settings using the following command:

**msfadmin@metasploitable:~$** sudo iptables –F (do not run this command)

**Q3.** Based on our first firewall rule, what type of firewall (packet filtering, stateful or application) are we using? Why did you come to this conclusion?

Stateful. Because In the command we used ‘ctstate’ and the firewall is aware of the state of each individual connection. For example, ESTABLISHED is an ongoing, previously established connection, which a stateless firewall does not know.

Now, we should set some rules to allow specific traffic... Try adding SSH to be allowed:

**msfadmin@metasploitable:~$** sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT

Maybe we should probably add port 80 (aka HTTP) to our rule set as well:

**msfadmin@metasploitable:~$** sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT

OK, we have some firewall rules added, we should try another nmap scan to see if we have achieved our goal of securing our system (NOTE: make sure you are on the Kali Linux terminal, not the SSH session):

**root@kali**:**~**# nmap –sV –p 1-1000 192.168.19.10 –oN portscan2.txt

Compare the portscan1.txt and portscan2.txt documents. Open the documents using the leafpad text editor (go Applications > Favourites > Leafpad, or just double click the text files in your home directory).

**Q4.** Has our firewall closed any ports on Metasploitable2? Why not?

No, because we have not implemented any rules in iptables to drop or close any ports.

We are going to add one more firewall rule today. Use the following command:

**msfadmin@metasploitable:~$** sudo iptables -A INPUT -j DROP

Try one more portscan to see if our firewall is dropping unspecified packets:

**root@kali**:**~**# nmap –sV –p 1-1000 192.168.19.10 –oN portscan3.txt

So, it looks like we have managed to successfully set up a very basic firewall on our Metasploitable2 server.

**Q5.** Document all of you firewall rules using “sudo iptables –L”?

Chain INPUT (policy ACCEPT)

target prot opt source destination

ACCEPT all -- anywhere anywhere ctstate RELATED,ESTABLISHED

ACCEPT tcp -- anywhere anywhere tcp dpt:ssh

ACCEPT tcp -- anywhere anywhere tcp dpt:www

DROP all -- anywhere anywhere

**Q6.** It looks like we have filtered all traffic apart from SSH (port 22) and HTTP (port 80). What now happens to all the other services (FTP, telnet, SMTP etc) on our server? Write some rules to allow traffic incoming to these essential services.

All other services will be dropped because they were not explicitly accepted in the iptables rules

FTP: sudo iptables -A INPUT -p tcp --dport 21 -j ACCEPT

Telnet: sudo iptables -A INPUT -p tcp --dport 23 -j ACCEPT

SMTP: sudo iptables -A INPUT -p tcp --dport 25 -j ACCEPT