EN4720: Security in Cyber-Physical Systems
Exercise — Infrastructure Security

Name: Thalagala B. P.
Index No: 180631J

June 20, 2023

This is an individual exercise!
Due Date: 20 June 2023 by 11.59 PM

This exercise has to be carried out using a Linux-based PC/virtual machine. Read all the instructions and questions before attempting the exercise. Add answers under each question and submit the resulting PDF.

Section 1

In this section, you will implement Firewall rules using **iptables** and **ufw** Linux commands. Moreover, you will scan network ports of a remote device using **nmap** Linux command.

For all the questions in this section, add a screenshot of the terminal (including all the commands you ran to perform the task) unless specified otherwise. The evaluator should be able to see each step that you followed to perform each task.

1. View the currently logged in user.

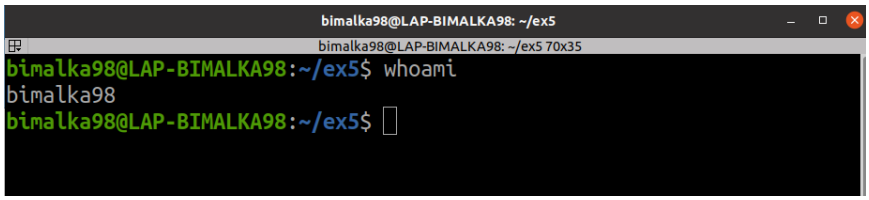

Figure 1: Currently logged-in user

Creating Firewall Rules with iptables

2. Use `dpkg -l | grep iptables` command to check whether iptables is installed on your system. If it does not existing in your system, install it by running `sudo apt-get install iptables`.

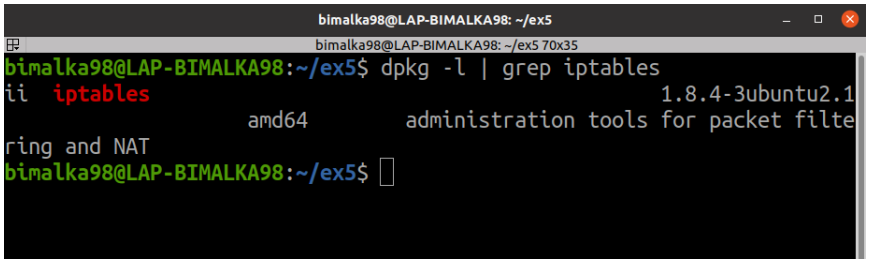

Figure 2: Checking whether `iptables` is installed on the system

3. Check all available iptables rules in your system using the command `/sbin/iptables -n -L` .

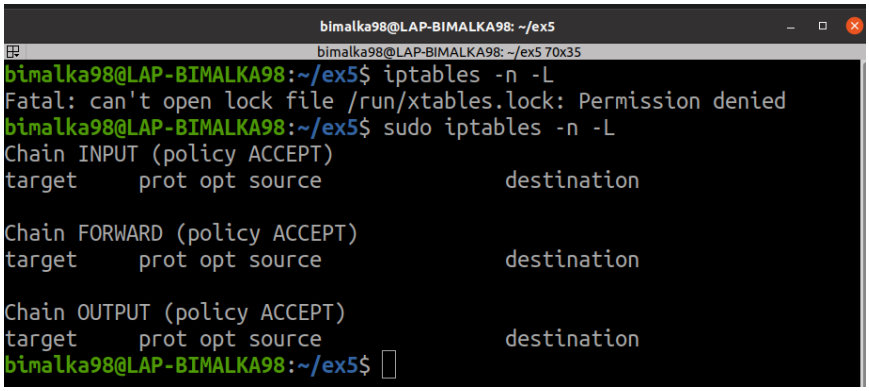

Figure 3: Checking all available `iptables` rules

4. Save all available iptables rules to a file named **iptablesRule.v4** using `iptables-save` command.

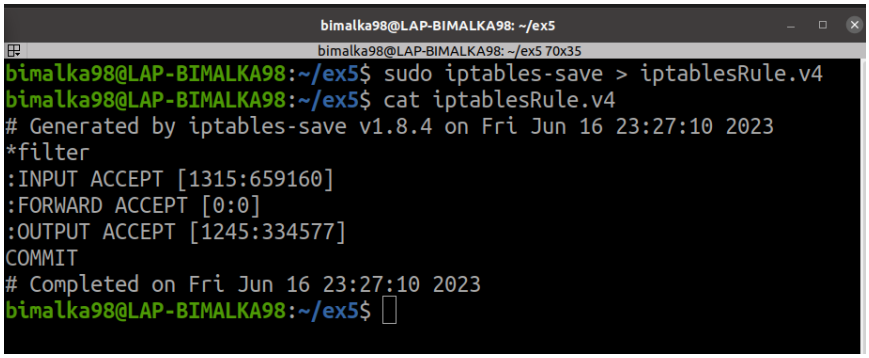

Figure 4: Saving all available `iptables` rules

5. Flush all the iptables rules that exist in your system and set a default policy to drop packets.

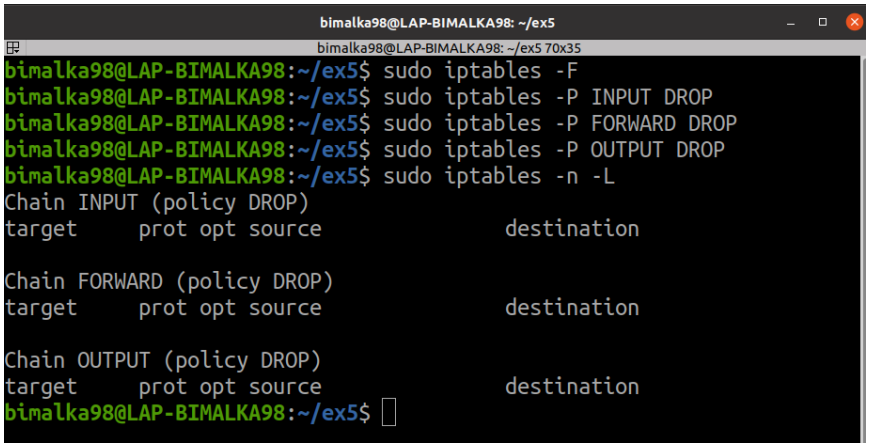

Figure 5: Flushing all the rules and setting a default policy to drop packets

6. Set iptables rules to permit input and output DNS traffic in your system.

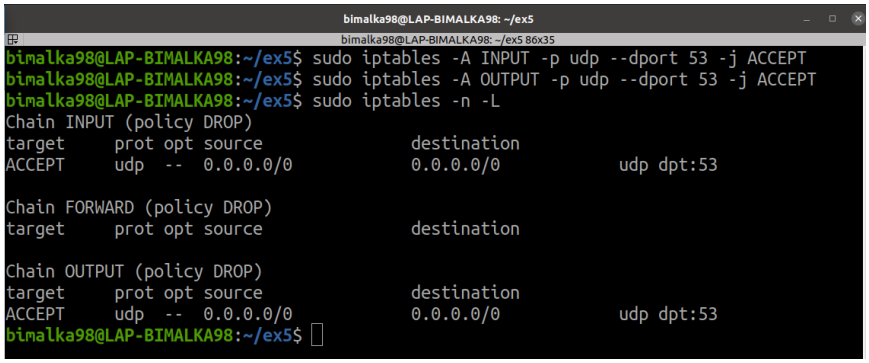

Figure 6: Permitting input and output DNS traffic

7. Add iptables rules to accept local network incoming and outgoing traffic from the network 192.168.1.0/24.

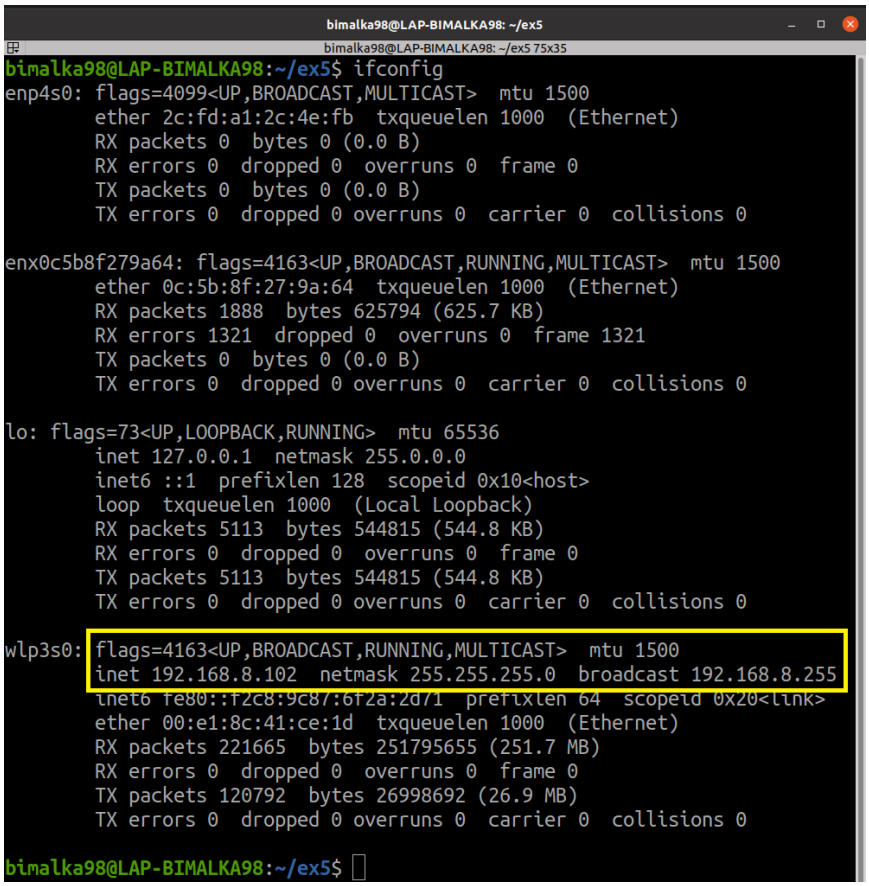

Figure 7: Checking the CIDR of the current local network

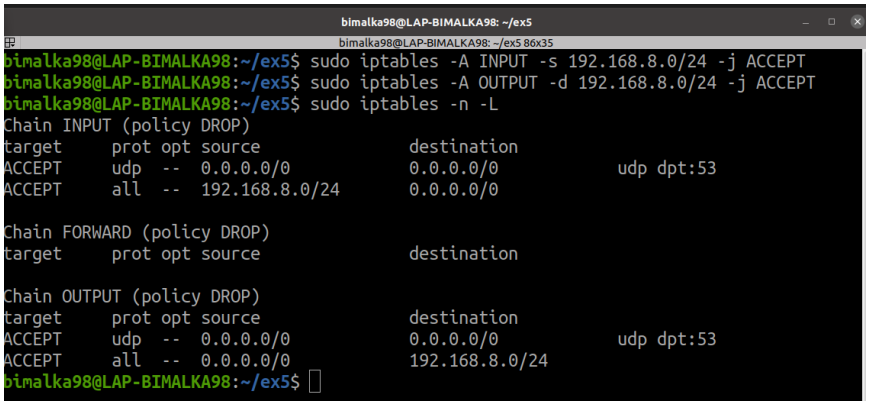

Figure 8: Adding rules to accept local network incoming and outgoing traffic

8. Configure iptables rules to allow all HTTP traffic.

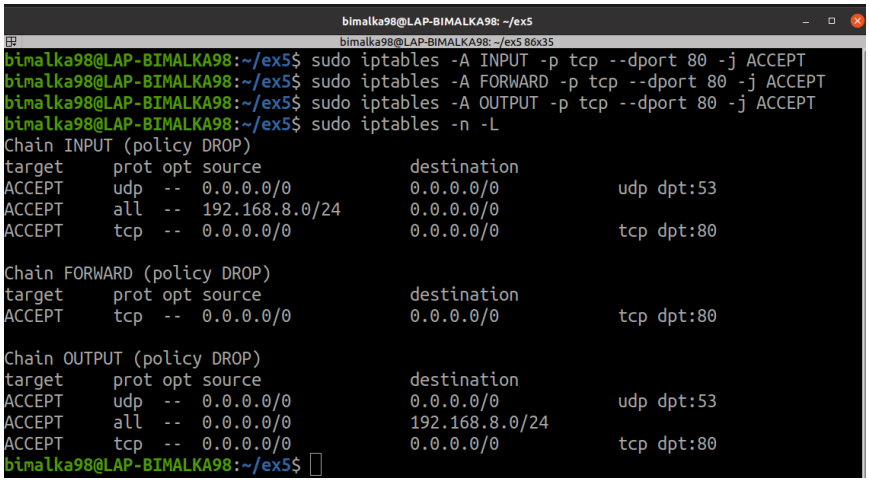

Figure 9: Configuring rules to allow all HTTP traffic

9. View all iptables rules in your system and save them to a file **iptablesRuleNew.v4**.

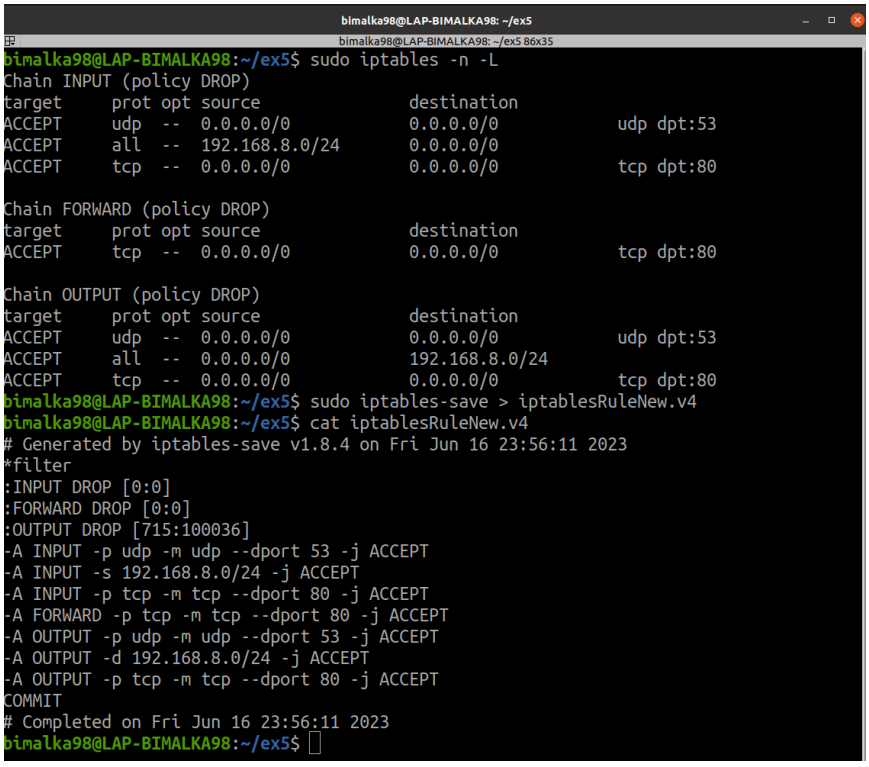

Figure 10: Viewing all rules and saving them to a file

10. Create a file called **iptablesCommands.sh** and put all commands you ran from steps 4, 5 and 6 in the file. After creating the file, flush your iptables commands again and run **iptablesCommands.sh** file. View the iptables rules now and compare with the previous result.

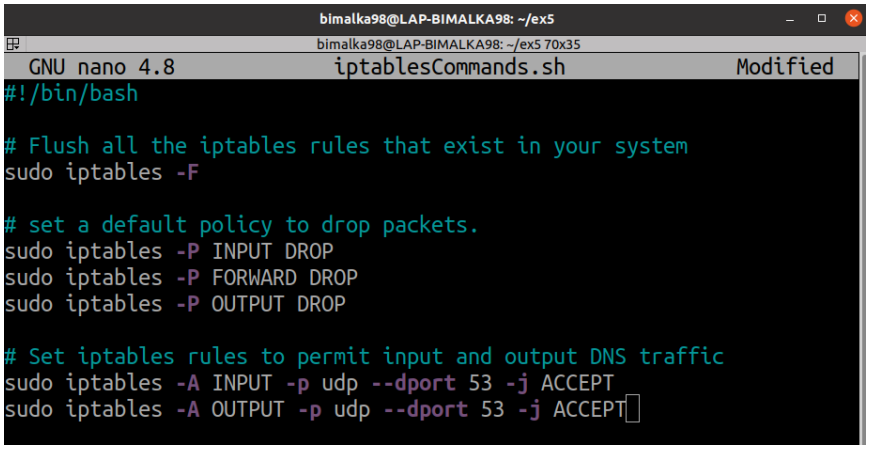

Figure 11: Creating the shell script

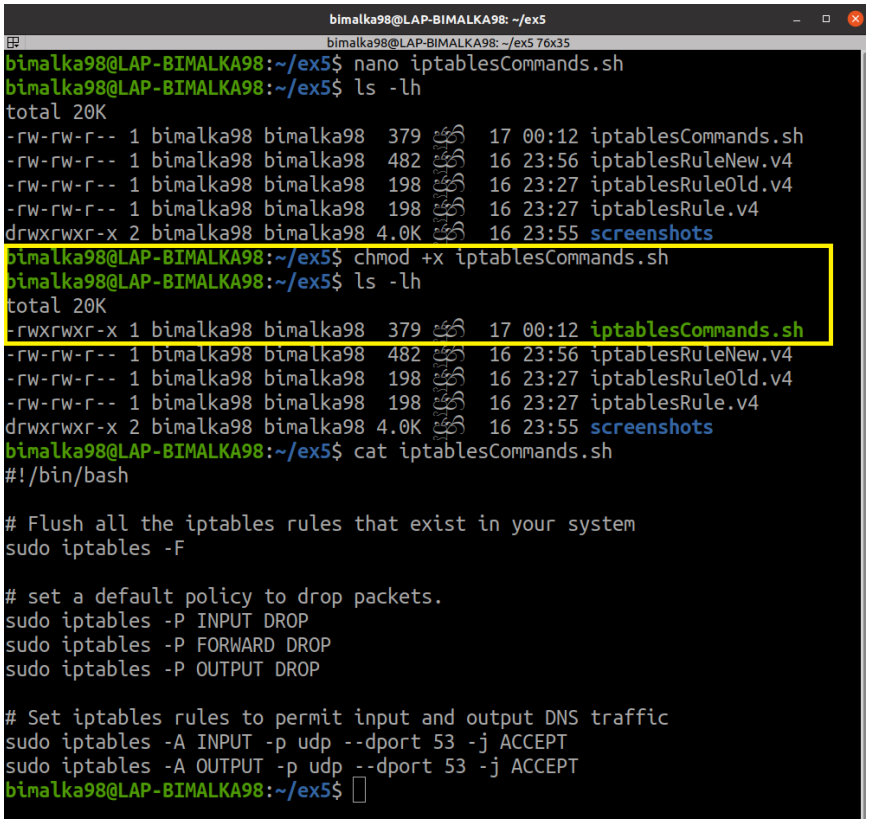

Figure 12: Making the script executable

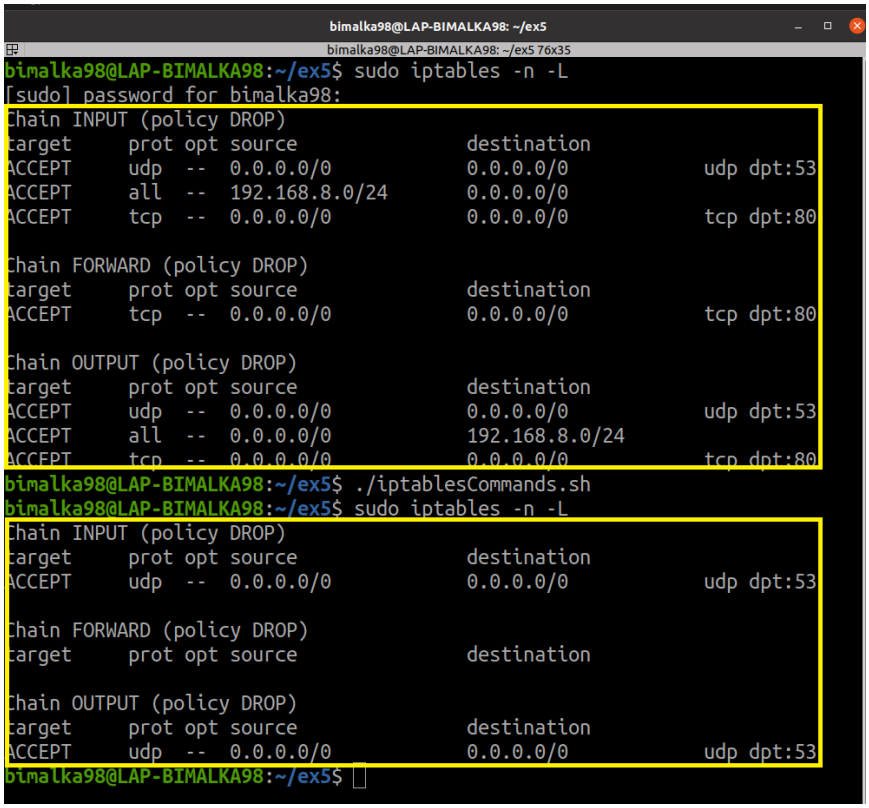

Figure 13: Executing the shell script

As it can be observed in the Figure 13, it is possible to set `iptables` *rules through shell scripts, which is an efficient way when it comes to setting rules in multiple host devices. In terms of the differences there is no difference between setting the rules manually and setting the rules through the script. The reason for the change in the number of rules is just because some rules were not included in the script (only the rules from item 4, 5 and 6 are there).*

11. Finally, flush your iptables rules again. But this time, load the saved iptables rules from the file **iptablesRuleNew.v4** using `iptables-restore` command. View the iptables rules and compare them with the ones you have in step 8.

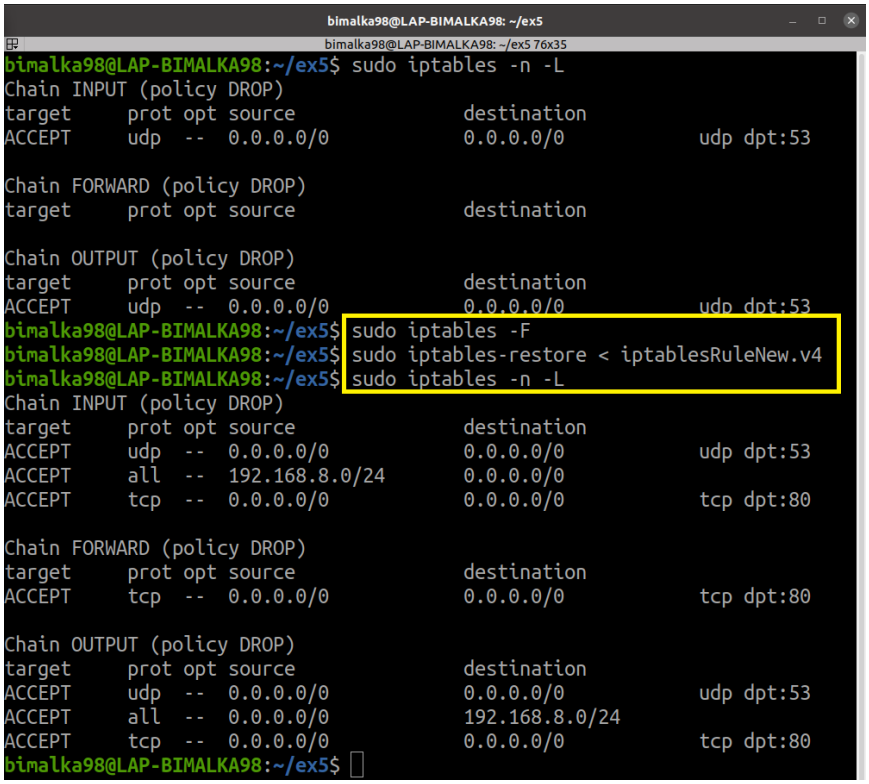

Figure 14: Flushing current rules and restoring saved rules

When comparing the output shown in the Figure 14 and the Figure 9, it can be observed that there is no difference. This means it is possible to save `iptables` *of a host at a given moment and restore them at another moment as required. This can be a useful tool when it comes to backing up rules before any significant change to the rules, because if anything goes wrong we have a fallback position.*

Creating Firewall Rules with UFW

The scenario comprises of two virtual machines (VM1 IP - 192.168.46.140 and VM2 IP - 192.168.46.141) running on a host (HOST IP - 192.168.46.1) machine. VM1 is an Ubuntu virtual machine that has a firewall implemented/configured.

The current firewall ruleset is as below.

```
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
```

All chain policies are set to drop traffic. To implement base rules, you can use the following commands:

- Delete any current rules associated with UFW using `sudo ufw reset`
- Disable UFW using `sudo ufw disable`
- Flush all iptables rules using `sudo iptables -F`
- Enable UFW using `sudo ufw enable`
- Deny outgoing traffic using `sudo ufw default deny outgoing`

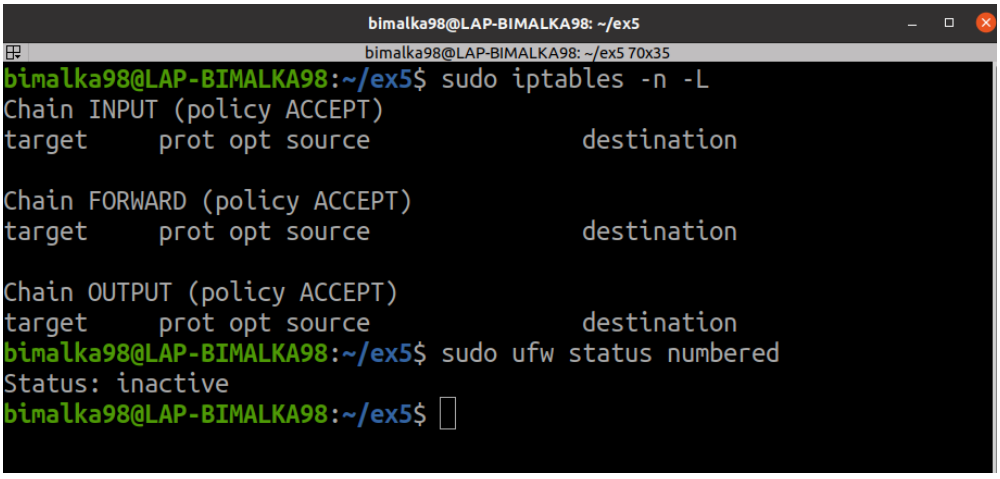

Figure 15: Current firewall rules

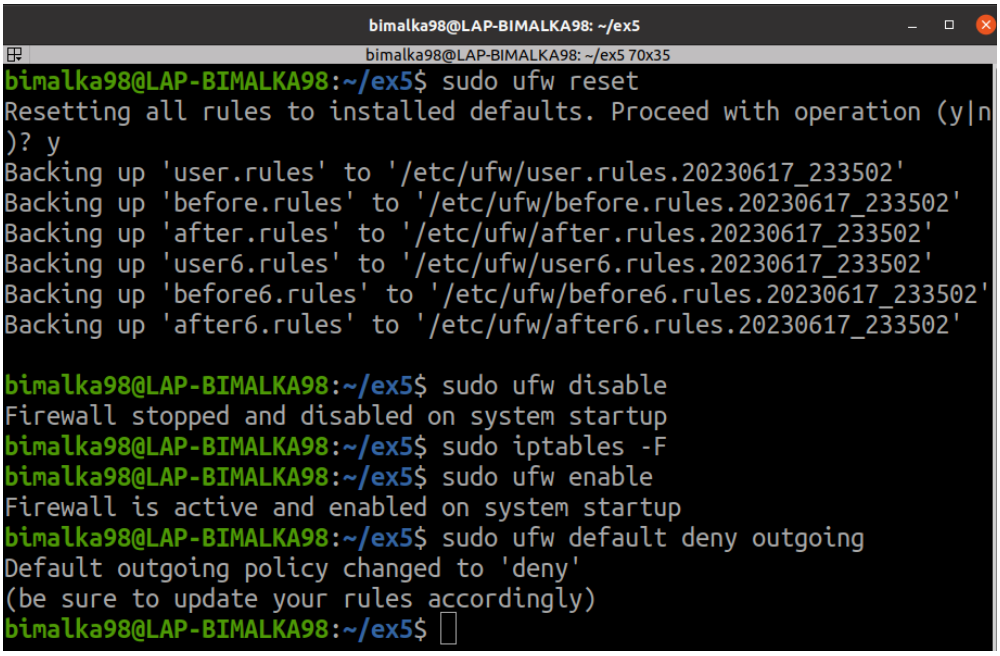

Figure 16: Implementing the base rules

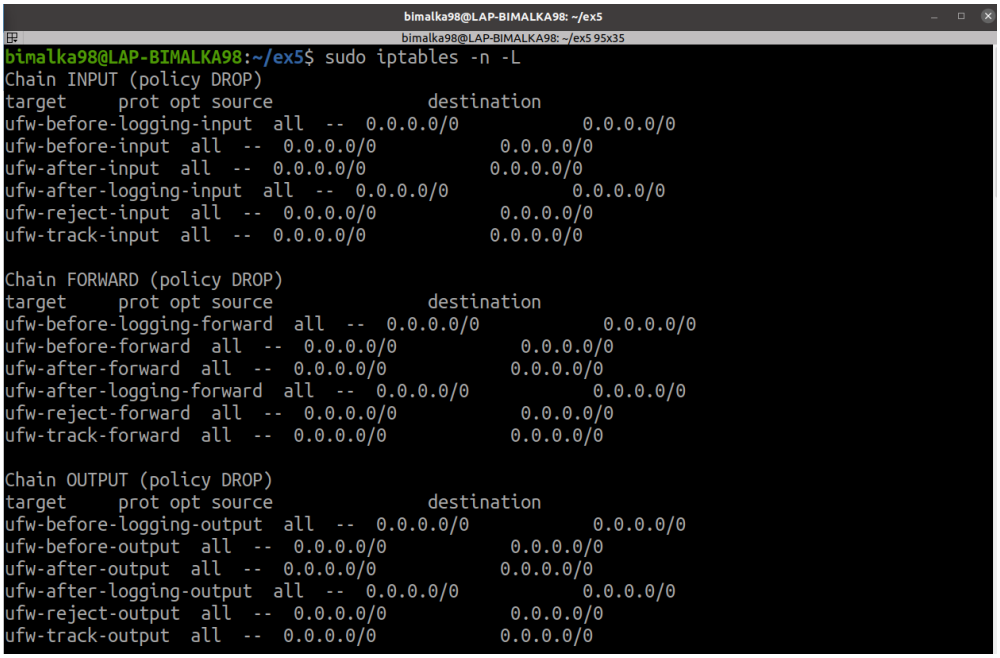

Figure 17: `iptables` after setting up the base rules

12. Implement the following network administration in VM1:

    - Access to VM1 from VM2 must only be allowed over FTP and Telnet.
    - Access to VM1 from HOST must only be allowed over SSH
    - Allow all outgoing traffic from VM1 with the exception of access to HTTP websites

    In this task, you are asked to implement UFW rules on the ubuntu machine. You can pretend that VM2 and HOST exist in your network. List the commands you used to achieve the above. Add a screenshot of the terminal output after running the command `sudo ufw status numbered`.

    If the firewall is physically implemented, you could have tested the connections using PuTTY or the command line.

    Following network administration in VM1 is configured using the associated commands given below.

    - *Access to VM1 from VM2 must only be allowed over FTP and Telnet*

      ```
      sudo ufw allow in from 192.168.46.141 to any port 21 proto tcp
      sudo ufw allow in from 192.168.46.141 to any port 23 proto tcp
      ```

    - *Access to VM1 from HOST must only be allowed over SSH*

      ```
      sudo ufw allow in from 192.168.46.1 to any port 22 proto tcp
      ```

    - *Allow all outgoing traffic from VM1 with the exception of access to HTTP websites*

      ```
      sudo ufw default allow outgoing
      sudo ufw deny out to any port 80 proto tcp
      ```

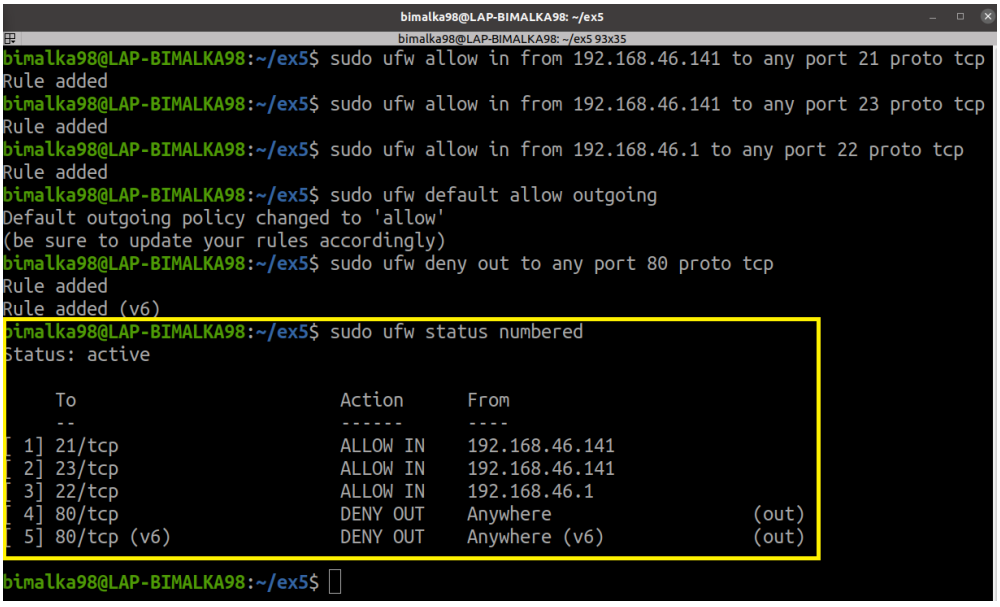

Figure 18: Output after running the command `sudo ufw status numbered`

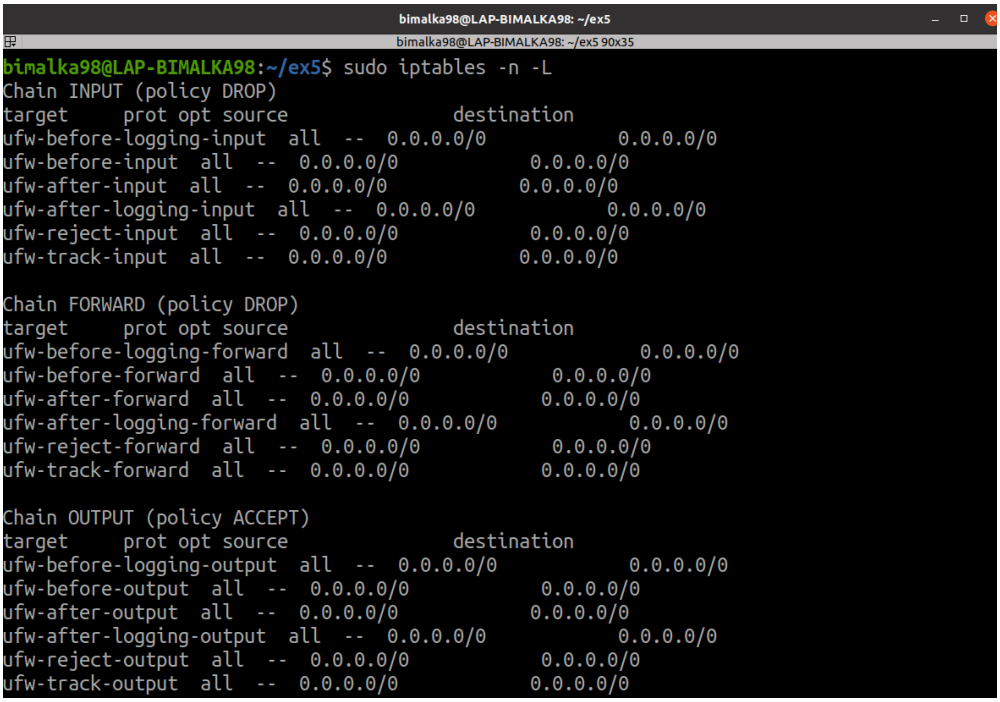

Figure 19: `iptables` ruleset after mentioned configurations

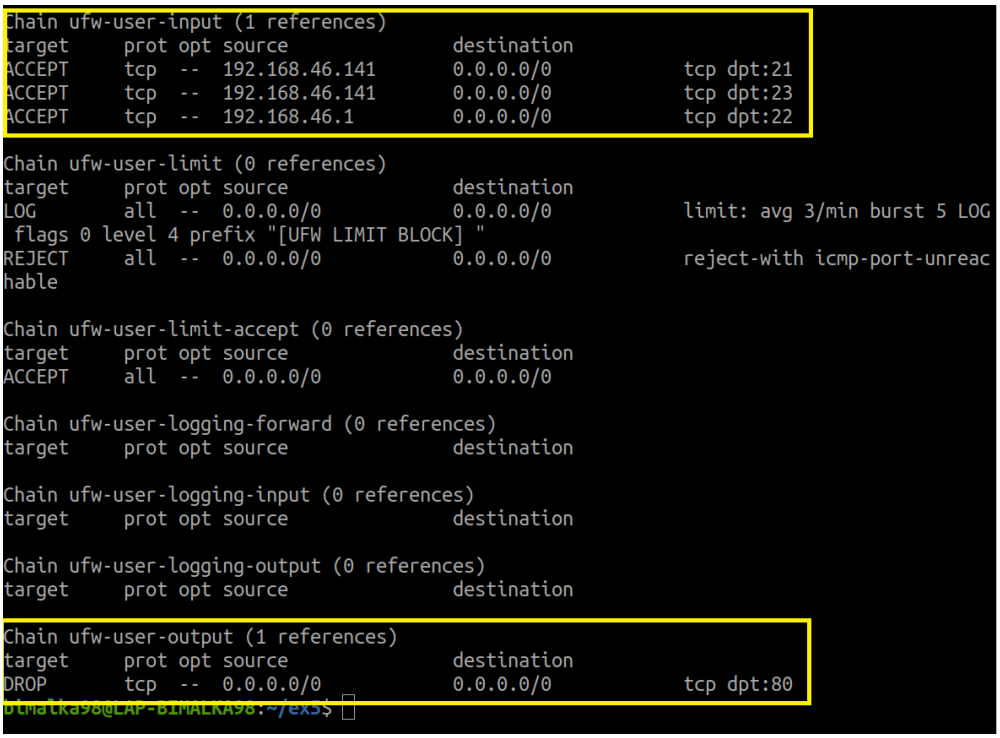

Figure 20: Mentioned configurations in the `iptables`

Scan systems with NMAP

In this section, you will scan for the Ports of a remote host. You will need to have two devices connected to the same local network to perform this task.

13. View ip addresses of both devices using `hostname -I` command.

Figure 21: IP addresses of the two devices (Laptop & a Raspberry Pi)

14. Scan one host from the other host for TCP and UDP ports using `nmap` command.

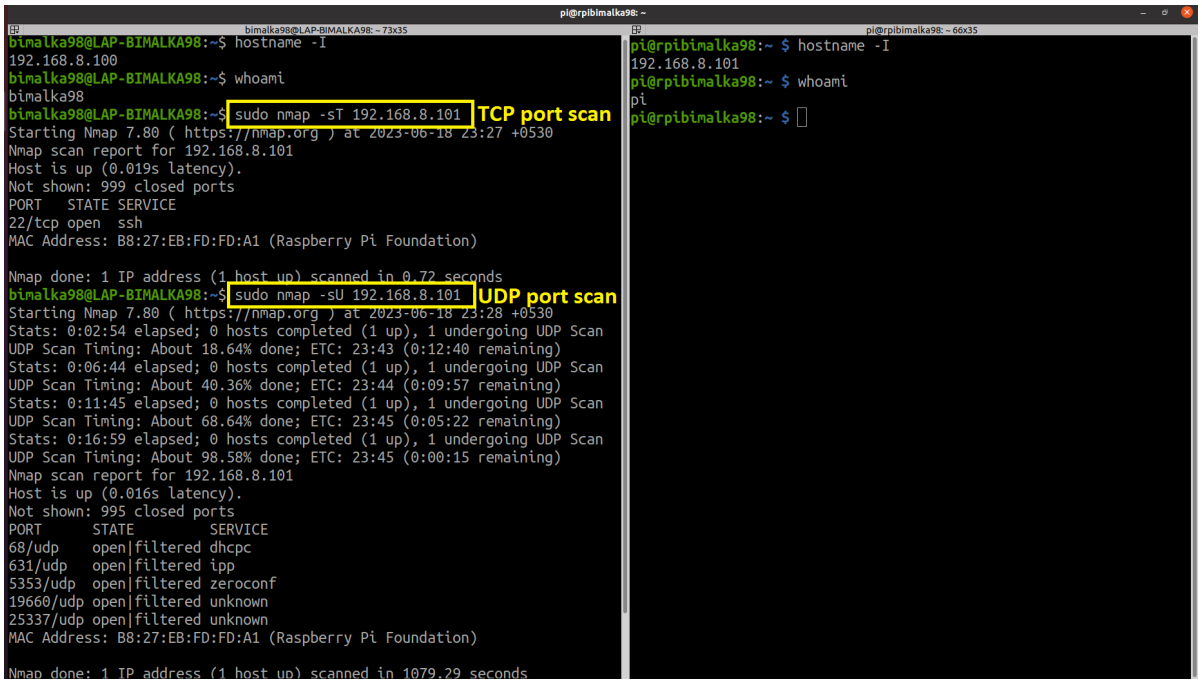

Figure 22: Scan Raspberry Pi device from the Laptop for TCP and UDP ports

Section 2

15. Briefly explain VLANs, VPNs, DMZs and Network Segmentation concepts outlining their similarities and differences.

    VLAN (a virtual local area network *allows multiple logical networks to exist on the same physical network. This improves the security and the performance of a given network by limiting the access to the resources within the given VLAN and isolating the traffic of a given VLAN from other VLANs respectively. VLANs assigned for each department of an organization is an example for this.*

    VPN (a virtual private network) *allows an encrypted tunnel to be created between two end points in an unsecured network (eg: internet) for secure communication. This minimize the possibility of being tracked by a malicious third party. VPNs are used to connect remote employees or branches of an organization, to its corporate network for secure communication.*

    DMZ (a demilitarized zone) *or otherwise known as a perimeter network/ screened subnet is a physical or logical network segment (a buffer) that separates an organization's internal LAN from the other unsecured networks (eg: internet). This allows an additional layer of security to be maintained, as it restrict the direct access from the internet to the LAN. The services/ servers provided to public are placed in the DMZ, while organization's private resources are kept in the internal LAN network.*

    Network Segmentation *is a method to isolate/ control the traffic of a given network. As in the VLANs, improved performance due to reduced traffic congestion and improved security due to virtual isolation of networks are the benefits.*

16. Perform a comparison between IPsec and SSL.

    Answer was adapted from:
    `https://www.geeksforgeeks.org/difference-between-ipsec-and-ssl/`

Table 1: Comparison of IPsec and SSL

IPsec	**SSL**
A set of protocols that provide security for IP (Internet Protocol) protocol	A protocol developed for secure communication
Operates in the *Internet/ Network* layer of the TCP/IP stack	Operates in between *Transport* and *Application* layers of the TCP/IP stack
Used in securing VPNs	Used in securing web transactions
Involves a complex configurations	Comparatively less complex to set up
Installation is not specific to the vendor	Installation is specific to the vendor

17. Explain the differences between an IDS, an IPS, and a firewall?

*An **IDS (Intrusion Detection System)** is a passive network monitoring system, which is only capable of detecting threats and alerting the network administrator or the SOC (the security operations center).*

*Whereas **IPS (Intrusion Prevention System)** is an active network monitoring system which not only detect the threats, but also take actions to prevent or block them in real time. This is done through analyzing the traffic and checking them against known attack patterns or policies of the network.*

*In contrast to the above, a **Firewall** provides a general protection by simply controlling the access to a given network, based on predefined set of rules. This can include the rules to allow or block traffics from a given subnet, allow or block traffics through a specific port and etc.*

18. What is the difference between anomaly detection and signature or heuristic-based intrusion detection?

Both of the methods are used in intrusion detection systems to identify potential threats. However, the operating principles are different.

*When it comes to the anomaly detection, system evaluates the network traffic with respect to a normal/ usual behavior, and reports if anything deviates from that baseline. This method is effective at **identifying new or unknown threats**.*

*However, signature based system evaluates the traffic by comparing them with a database of known threats (signatures/ patterns). This kind of system is effective at **identifying known threats**.*