Unit 2: Introduction To Windows And Linux Firewall

2

Unit Structure

2.1. Learning Objectives

2.2. Introduction

2.3. Windows Firewall

2.4. Linux Firewall

2.5. Let Us Sum Up

2.6. Check your Progress: Possible Answers

2.1 LEARNING OBJECTIVE

After studying this unit student should be able to:

- Understand Linux and window firewalls.

2.1 INTRODUCTION

In this chapter, we will see the details of the firewall in the different operating system such as Linux and windows and how it works. As both Windows and Linux are completely different operating systems, supports different file types, Linux is an open source and windows is a propriety one.

Both operating systems can be used as a standalone machine as well as can be used for the server machine based on the requirements. But there is something common set of requirements for any user is to secure from the external threats.

Many of the network administrators think that Linux has many advantages over windows, not just only freely available. But it is more stable than windows, less often crashing than windows. Easy configuration and less downtime required. Can be easily used for a file server, web server, email server, can be used in an intranet. Also as a router and firewall to help to connect to the network.

2.3 WINDOWS FIREWALL

Over the period of time windows operating system has grown much in providing its core functionality as an operating system. Windows Firewall was first included in Windows XP (back in 2001), and since then it has been improved in each new version of Windows. For every operating system, it is important to provide the core security infrastructure inbuilt within the operating system which handles implementing security protocols, enforcing security policies by providing dedicated firewall as software to monitor the network traffic.

One of its roles is to block unauthorized access to your computer. The second role is to permit authorized data communications to and from your computer.

Firewall Functions in Windows Operating System: Windows firewall with advanced security in windows server operating systems blocks unauthorized

network traffic flowing into or out of a local device by providing host-based, two-way network traffic filtering.

While the old Windows Firewall allowed you to configure only a single set of inbound and outbound rules (a profile), Windows Firewall with Advanced Security includes three profiles (Domain, Private and Public), so you can apply the appropriate rules to each server based on its connection to the network.

- Domain networks. Networks at a workplace that are attached to a domain.
- Private networks. Networks at home or at work where you trust the people and devices on the network. When private networks are selected, network discovery is turned on but file and printer sharing is turned off.
- Guest or public networks. Networks in public places. This location keeps the computer from being visible to other computers. When a public network is the selected network location, network discovery and file and printer sharing are turned off.

You can also configure the following options for each of the three network profiles in advance windows firewall settings:

- Firewall State. You can turn the firewall on or off independently for each profile.
- Inbound Connections. You can block connections that do not match any active firewall rules (this is the default), block all connections regardless of inbound rule specifications, or allow inbound connections that do not match an active firewall rule.
- Outbound Connections. You can allow connections that do not match any active firewall rules (this is the default) or block outbound connections that do not match an active firewall rule.
- Protected Network Connections. You can select the connections — for example, the Local Area Connection — that you want Windows Firewall to help protect.
- You can configure display notifications and unicast responses, and merge rules that are distributed through Group Policy.
- You can configure and enable logging.
- IPsec Settings. You can configure the default values for IPsec configuration.

Apart from the packet filtering, IP security windows also provide the functionality of the VPN(Virtual Private Network).

> **Virtual Private Network:** A VPN, or Virtual Private Network, allows you to create a secure connection to another network over the Internet. VPNs can be used to access region-restricted websites, shield your browsing activity from getting traceback while on the public network.
>
> They originally were just a way to connect business networks together securely over the internet or allow you to access a business network from home. Most operating systems have integrated VPN support.

You can use a VPN to:

- Bypass geographic restrictions on websites or streaming audio and video.
- Protect yourself from snooping on untrustworthy Wi-Fi hotspots.
- Gain at least some anonymity online by hiding your true location.
- Protect yourself from being logged while torrenting.

By default Windows Firewall is on and can be found as Goto Control Panel then goes to Windows Firewall. Click on the Windows Firewall and you will see the current status of the firewall and types of networks. Active connections if there are any.

On the left side, there are several options to configure the default firewall setting to change it as per requirements.

On left side panel there in the above image, there is an option of Advance Setting on clicking that option will lead you to open another window of Windows Firewall with Advanced Security as shown below; Where you can set the inbound and outbound rules.

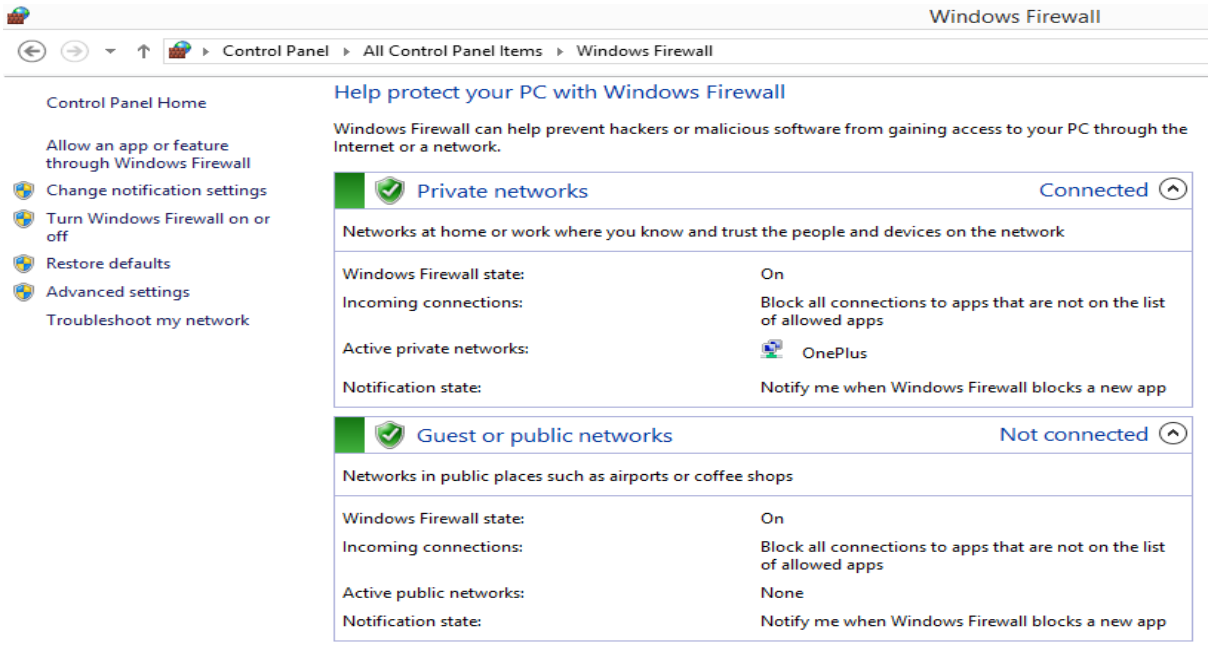

Figure 2.1 Windows Firewall

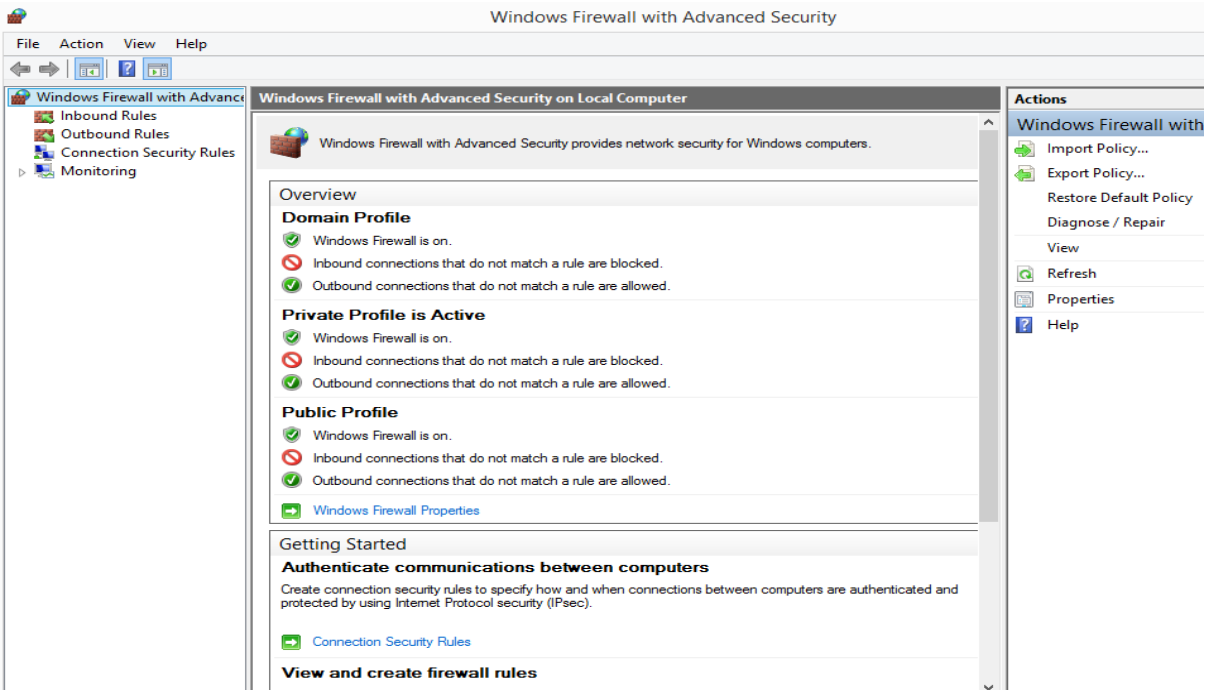

Figure 2.2 Windows Firewall with Advanced Security

2.4 LINUX FIREWALL

Before getting deep into the Linux Firewall we will go through some of the highlights of the Linux Operating System. Linux was created in 1991 by Linus Torvalds when he was an undergraduate student at the University of Helsinki in Finland. He has first

created his own operating system based on Unix. After that nearly two decades Linux has become a full-featured operating system which is fast and reliable. Linux has got a solid reputation for efficiency and security.

Linux is a multiuser operating system. Which means more than one user can log on into the system and can use the system at the same time; Where mostly all versions of windows are the single-user system. Only one user at a time can log in in a windows machine and can use it.

Linux has a very different way of using the file system, unlike windows. There is no concept of “C:/” Drive in Linux. Instead, Linux combines all drives and partitions into a single directory hierarchy. In Linux, one partition is designated as “root” partition. It is similar to the C:/ drive in the windows system. There are many distributions available based on the package manager which is either Debian based or RPM-based operating system in Linux.

Though there are common components which are present in all different distributions which are Linux Kernal, administrative tools and packages. An operating system such as Ubuntu, Lubuntu, are Debian based operating system which supports .deb packages. Another one is RPM-based package manager operating system such as Fedora, CentOS. Which supports .rpm based packages. Redhat is one operating which is the most stable version of RPM-based operating system and which doesn't come freely. Let us understand the security features of the Linux operating system.

Netfilter: Netfilter is a framework provided by Linux Kernal which allows networking operations to be implemented in the form of handlers. Netfilter supports different operations and functions for packet filtering, network address translation, a port translation which allows or rejects the network packets in the network.

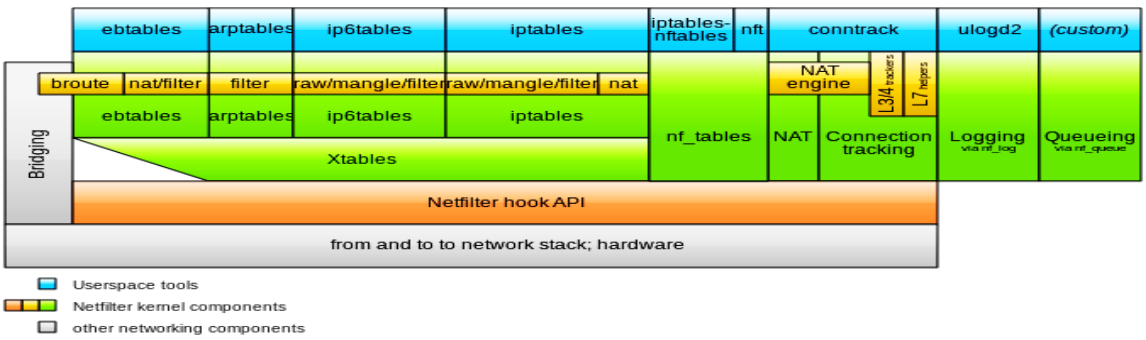

Figure 2.3 Netfilter Components Source: Wikipedia

The *Netfilter* framework included in the Linux kernel restricts incoming and outgoing network connections according to a set of rules that have been defined by the administrator. Several Linux distributions configure firewall rules by default and offer utilities for managing simple firewall configurations. You may also manage the firewall rules on any Linux system with the standard iptables and ip6tables command-line utilities. Use of iptables will only configure restrictions for IP version 4 connections and that you will need to use ip6tables to set up rules for IP version 6 as well.

Fedora, Red Hat, and SUSE automatically enable the firewall and supply their own graphical configuration utilities. You must manually configure and enable the firewall on Debian and Ubuntu systems. Current releases of Ubuntu include a command-line utility called *ufw* for firewall configuration.

Those Linux distributions that enable a firewall by default use a netfilter configuration that blocks connections from other systems. Any attempt by a remote system to access a service on a blocked port simply fails. This means that no other system may connect to an installed service unless you specifically choose to unblock the relevant port.

Let us try to understand the basic functionality such as NAT(network address translation), Port forwarding

Network Address Translation: Network Address Translation (NAT) is a process in which one or more local IP address is translated into one or more Global IP address and vice versa in order to provide Internet access to the local hosts. Also, it does the translation of port numbers i.e. masks the port number of the host with another port number, in the packet that will be routed to the destination. It then makes the corresponding entries of IP address and port number in the NAT table. NAT generally operates on router or firewall.

Port Forwarding: Port forwarding or port mapping is an application of network address translation (NAT) that redirects a communication request from one address and port number combination to another while the packets are traversing a network gateway, such as a router or firewall. This technique is most commonly used to make services on a host residing on a protected or masqueraded (internal) network available to hosts on the

opposite side of the gateway (external network), by remapping the destination IP address and port number of the communication to an internal host.

IPtables: IPtables which is an inbuilt firewall in Linux system. It is a user based application for configuring the tables provided by the Linux kernel firewall. iptables is the default firewall installed with Red Hat, CentOS, Fedora distributions.

Different modules and programs are used for different protocols such as iptables for IPv4, ip6tables for IPv6 and so on. It uses the concept of IP addresses, protocols (TCP, UDP, ICMP, etc) and ports.

IPtables is a command line firewall that uses the concept of chains to handle the network traffic. It places the rules into chains, i.e., INPUT, OUTPUT, and FORWARD, which are checked against the network traffic. Decisions are made as to what to do with the packets based on these rules, i.e., whether the packet should be accepted or dropped.

These actions are referred to as targets. DROP and ACCEPT are commonly used predefined targets used for dropping and accepting the packets, respectively.

IPtable architecture comprises groups of network packets, processing rules into tables and chains for processing the rules.

Rules consist of matches to determine which packet the rule will apply to and the targets. They operate at the network layer.

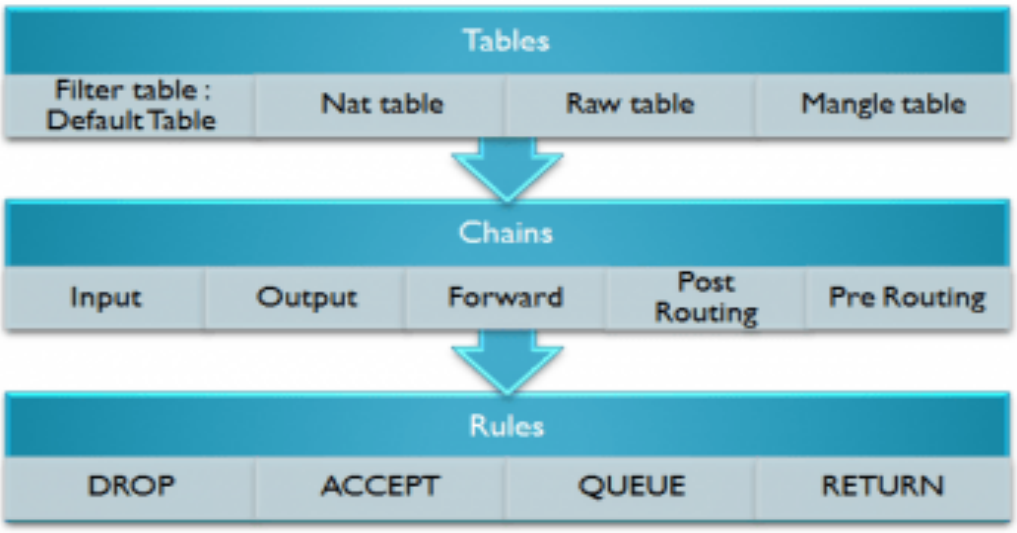

Figure 2.4 Iptables Architecture

Before you can configure rules with the iptables command, you have to understand a few concepts and Linux-specific terms:

Tables: This is a default table that Linux firewall stores and maintains sets of rules. The main table is the filter table, where you define most rules that apply to incoming and outgoing traffic. The nat table contains rules that define how Linux performs NAT. The mangle table is used for advanced packet routing.

Chains: Linux uses this term to refer to a set of rules that Linux applies when filtering network traffic. Here are the three main chains, each of which is part of the filter table:

The three predefined chains in the filter table to which rules are added for processing IP packets are:

- INPUT: These are packets destined for the host computer.
- OUTPUT: These are packets originating from the host computer that leaves from the firewall.
- FORWARD: These packets are neither destined for nor originate from the host computer, but pass through (routed by) the host computer. This chain is used if you are using your computer as a router.

Let us see with the commands used on the Linux terminal, on how to work with iptables. Linux includes many different numbers of iptables commands. We will start with the basic syntax of the command-line options.

>>iptables [-t table] CMD [chain] [filter_match] [target]

Iptables commands must specify the table where the command will be applied, the command itself, the chain to which the command will belong, an expression that defines what type of traffic the filter will apply to, and what Linux should do with the packet. For example, to add a simple rule to the input chain of the filter table that would drop all ICMP traffic, your command-line would look something like this:

>>iptables -t filter -A INPUT –p icmp DROP

Above command tells the Linux system that, when the filter table's input chain receives the packet which uses ICMP protocol, send the packet to DROP target. Which simply means that to dump all ICMP protocol-related traffic.

Before moving to see more details we will first see the most commonly used iptables commands which are described in the below table.

Table 2.1 Iptables Commands

Command	Name	Description
-A	Append	This command appends a rule to the end of a chain.
-I	Insert	This command inserts a rule to the beginning of a chain.
-D <chain><rule number>	Delete Rule	This command deletes a rule.
-L [<chain>]	List	This command lists all rules in a chain. If you don't specify a chain, the command lists the rules in all chains.
-N <chain>	New	This command creates a new user-defined chain.
-X <chain>	Delete Chain	This command deletes a user-defined chain.
-F [<chain>]	Flush	This command deletes all rules in a chain. If you don't specify a chain, the command deletes all rules in all chains.
-h	Help	This command lists all iptables commands and options.

It is important to check the order of the commands used to prepare the rules. As Linux firewall. Once it matches the packet it no further checks for next defined rules.

Because of this, iptables has the convenient -A command that appends commands to the end of the processing chain and the equally convenient -I command that adds commands to the beginning of the processing chain or a user-specified location.

Now we will check the common iptables options available to prepare the rule.

Table 2.2 Iptables Options

-p protocol	Specifies the protocol. It can be TCP, UDP, ICMP or the protocol number which is listed in /etc/protocols.
-s source_address	Specifies source address of the packet. Also, specify the subnet mask along with the source address such as –s 192.168.1.2/24
-d destination_address	Specifies the destination address of the packet.
--source-port	Specifies the source port of a TCP or UDP packet.
--destination-port	It refers to the destination port of the TCP or UDP packet.
-i interface	Specifies the network interface for the incoming packet. For eg: -i eth0.
-o interface	Specifies an output interface on which packet is to be sent. For eg: -o eth1.
-j target	Specifies that packet should be sent to the specified target. For eg: -j DROP. Which means that the packet should be sent to DROP target(to discard the packet).

To check the status of the iptables, execute the following command.

>>service iptables status

```
Table: filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
2    ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
3    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
4    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
5    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

Figure 2.5 iptables status

To start and stop iptables service, use the following command on Linux terminal.

>>service iptables start / stop

To open the iptables files on the Linux system, use the following command.

>>gedit /etc/sysconfig/iptables

Here the gedit is the text editor in Linux as similar as notepad in windows for text editing operations.

Now let us create some iptables rules.

To allow SSH traffic, use the following command:

>> iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

>> iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT

Above command specifies that for all input packets received on the interface eth0 for the TCP protocol with the destination port 22 (for SSH traffic) for the new connection or established one, accept the request.

Similarly, for the traffic leaving the firewall from the interface eth0 where the source port is 22 and established connection allow it to leave from the firewall.

Check Your Progress 1

---

1. Provide the iptable rule to block the ip address 9.9.8.8.

2. Provide iptable rule to accept the ping request from outside to inside and inside to outside.

3. Give short details regarding the iptables chains.

SELinux: Security-Enhanced Linux (SELinux) is a Linux kernel security module that integrated into the 2.6.x kernel using the Linux Security Modules (LSM)which provides a mechanism for supporting access control security policies, including mandatory access controls (MAC).

It is a project of the United States National Security Agency (NSA) and the SELinux community. SELinux integration into Red Hat Enterprise Linux was a joint effort

between the NSA and Red Hat. SELinux is also implemented as a standard feature in Centos/Fedora-based distributions and widely deployed.

A Linux kernel integrating SELinux enforces mandatory access control policies that confine user programs and system services, as well as access to files and network resources.

Limiting privilege to the minimum required to work eliminates the ability of these programs and daemons to cause harm if faulty or compromised (for example via buffer overflows or misconfigurations).

This confinement mechanism operates independently of the traditional Linux (discretionary) access control mechanisms.

SELinux is set in three modes:

- Enforcing – SELinux security policy is enforced. If this is set SELinux is enabled and will try to enforce the SELinux policies strictly
- Permissive – SELinux prints warnings instead of enforcing. This setting will just give a warning when any SELinux policy setting is breached
- Disabled – No SELinux policy is loaded. This will totally disable SELinux policies.

SELinux is set in two levels:

- Targeted – Targeted processes are protected.
- Mls – Multi Level Security protection.

To check whether the SELinux is enabled or not in the Linux system you can use this below commands. You can use the CentOS or Fedora (RPM) based Linux distributions for the SELinux workings.

Lets understanding the working of SELinux:

>>getenforce

The output will be either “enabled” or “disabled”

To check the status of the SELinux we can use:

>>sestatus

The output of the above command will be like:

Sample output:

SELinux status: enabled

SELinux mount : /selinux

Current mode: enforcing

Mode from config file: enforcing

Policy version: 21

Policy from config file: targeted

From the above output, we can see that SELinux is enabled and it is in enforced mode and to see detailed status you can use -b option, this will give on which services SELinux is enabled and which services are disabled.

>>sestatus -b

Sample Output:

SELinux status: enabled

SELinuxfs mount: /selinux

Current mode: permissive

Mode from config file: enforcing

Policy version: 24

Policy from config file: targeted

Policy booleans:

abrt_anon_write off

allow_console_login on

allow_corosync_rw_tmpfs off

allow_cvs_read_shadow off

allow_daemons_dump_core on

allow_daemons_use_tty on

allow_domain_fd_use on

allow_execheap off

allow_execmem on

allow_execmod on

SELinux can potentially control which activities a system allows each user, process, and daemon, with very precise specifications. It is used to confine daemons such as database engines or web servers that have clearly defined data access and activity rights.

2.5 LET US SUM UP

In this chapter, we have seen the basic fundamentals of the Windows firewall and Linux firewall. Both the operating system is different in terms of the file format, networking, security management. Windows has an inbuilt Windows Firewall which provides security management using inbound and outbound rules. Also, the user can allow which application network traffic to accept and reject. Similarly, in Linux, there are concepts of Netfilter, iptable and SELinux which are used on RPM-based distributions. In which you can there are chains which contain the set of rules/ policies defined and can be enforced on the system.

2.6 CHECK YOUR PROGRESS: POSSIBLE ANSWERS

Check Your Progress 1

1. iptable rule to block the ip address 9.9.8.8:

iptables -A INPUT -s 9.9.8.8 -j DROP

2. iptable rule to accept the ping request:

iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT

iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT

iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT

iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT

3. iptables chains:

INPUT chain – Incoming to the firewall. For packets coming to the local server.

OUTPUT chain – Outgoing from the firewall. For packets generated locally and going out of the local server.

FORWARD chain – Packet for another NIC on the local server. For packets routed through the local server.