ITS 332 Networking Lab

Firewalls

Dr Steven Gordon

Revision 927

6 November 2009

1 Overview

This lab will introduce you to a common security mechanism used in networks: *firewalls*. A firewall is a device (usually implemented in software) that controls what traffic can enter and leave a network. If an organisation wants to *protect* there network, then a firewall between their internal network and all external networks ("the rest of the Internet") will be configured to inspect the traffic entering/leaving the network, and only allow the traffic that meets the organisations policies. We will setup our own simple firewall—in practice, real firewalls will be much more complex, and often require specialised network equipment.

2 Background

Firewalls are network devices that control what packets enter and leave a computer network. Typically a company (and more recently, a home user) will use a firewall to stop people *outside* the company network (that is, everyone on the external Internet) from accessing computers and resources *inside* the company network (e.g. the SIIT network). For example:

- Stop people on the Internet from connecting to and accessing files on a SIIT computer
- Stop people on the Internet sending viruses and spam to computers in the SIIT network

The firewall can also be used to control what computers inside the network access. For example:

- Stop SIIT users from access *inappropriate* web sites on the Internet
- Stop SIIT users from sending `ping` commands to routers on the Internet

The firewall is usually a specialised router that acts as a gateway between the local network and the outside networks. That is, all traffic goes through the firewall. However, in this lab, we will see that we can configure the Ubuntu Linux computers to act as a simple firewall.

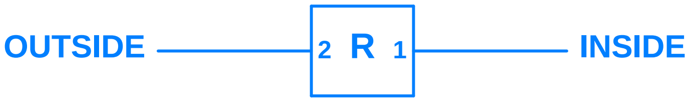


Figure 1: An organisation views their network as *inside*, and all other networks as *outside*

2.1 How Do Firewalls Work?

A gateway router (that is, the router between the inside and outside networks) normally receives an IP packet, looks at the destination IP address, looks up its routing table to determine where to sends the packet, and sends (or forwards) the packet.

A firewall is hardware or software running on the gateway router that provides additional functionality:

1. When the IP packet is received, the firewall looks at the packet and compares it to a set of rules stored in a firewall table. An example rule may be: "Drop all packets destined to IP address 64.233.189.104"

2. When a rule matches, the corresponding action is taken. The action is usually DROP (discard, do not let the packet through) or ACCEPT (forward, let the packet through). In the above rule, if the IP destination address was 64.233.189.104, then the packet would be dropped.

3. If the packet is not dropped, then the gateway router follows its normal procedures (e.g. look up routing table and send the packet).

2.2 Firewall Rules

The rules used by firewalls are the most important aspect. They can be very simple (e.g. "drop all packets destined to the local network") or very complex (e.g. 1000's of rules).

Packet-filtering firewalls usually create the rules using the following information:

- Packet match conditions:
  - IP source address
  - IP destination address
  - TCP/UDP source port number
  - TCP/UDP destination port number
  - Other IP/TCP/UDP header fields
- Direction of traffic:
  - Is the packet coming from outside (to inside) or is it coming from inside (to outside)
- Actions:
  - ACCEPT or DROP

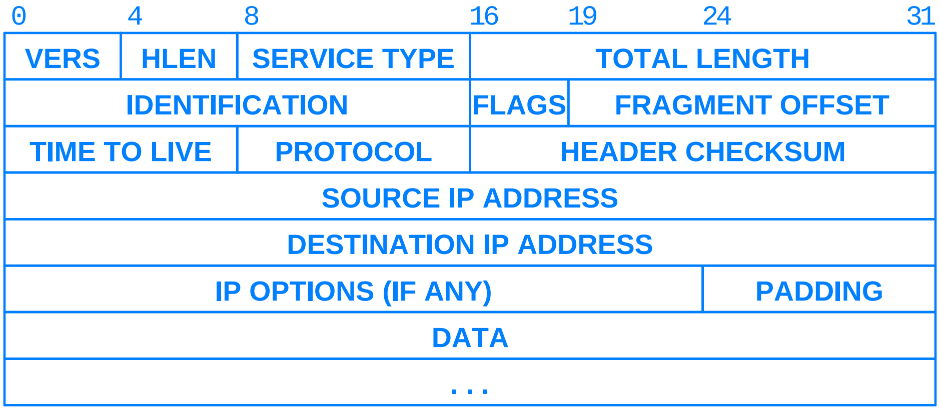


Figure 2: IP Datagram Header

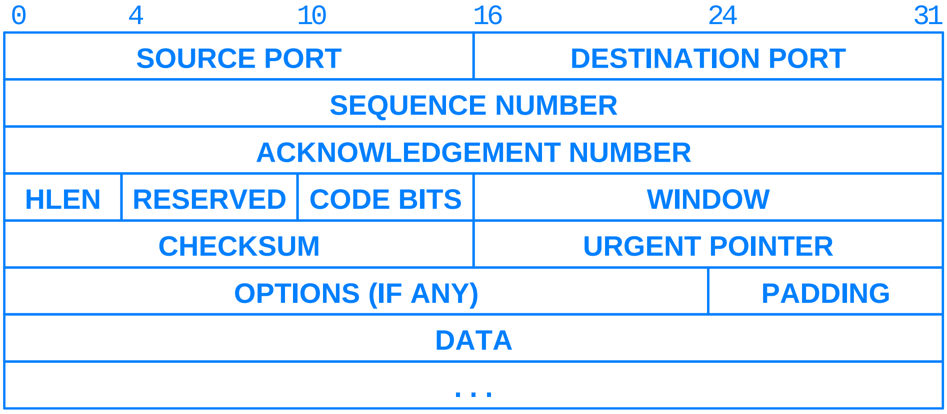


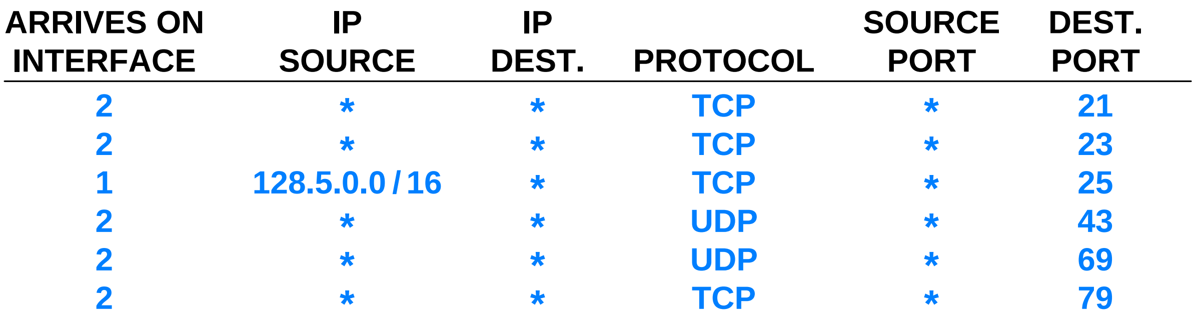
Figure 3: TCP Segment Header

Using the above conditions, a reasonably good firewall can be built that can filter packets based on where the packets are coming from, where they are going to, and what applications are being used (remember, if a destination port number is 80, we can assume that a web browsing application is being used--if SIIT wanted to stop all web browsing, then they could drop all packets destined to port 80).

More complex firewalls (application-level firewalls) can be created by not only looking at the TCP/IP packet information, but also looking at the content of the messages. For example:

- Does the packet contain an email virus or spam?
- Does the packet contain spam?
- Is the web request to an unacceptable server (e.g. www.illegal-site.com)?

Figure 4 shows an example set of rules for a firewall (on router R). Each row in the table specifies a rule. When a packet arrives at R (the firewall) the packet will be DROPPED if a rule matches. If no rules match, then the packet is ACCEPTED (forwarded).



ARRIVES ON INTERFACE	IP SOURCE	IP DEST.	PROTOCOL	SOURCE PORT	DEST. PORT
2	*	*	TCP	*	21
2	*	*	TCP	*	23
1	128.5.0.0 / 16	*	TCP	*	25
2	*	*	UDP	*	43
2	*	*	UDP	*	69
2	*	*	TCP	*	79

Figure 4: Example firewall rules

The example rules specify:

- Block all packets destined to following services (applications) on internal network: FTP (port 21); TELNET (23); WHOIS (UDP port 43); TFTP (69); FINGER (79).
- Block all packets coming from internal network 128.5.0.0 (subnet mask 255.255.0.0) and destined to external email server (port 25)

As a result, no-one outside the network could FTP to the inside network. And no-one inside the network using and address on the network 128.5.0.0 could send an email.

2.3 Firewalls and Servers

Most applications operate in a Client/Server mode, where a Client inside a network accesses a Server outside the network. Most computers inside the network DO NOT run servers accessible to the outside network. For example, there is no need for a SIIT staff or students PC to run a web server accessible to someone outside SIIT.

Therefore, it is common for firewalls to be setup that will:

- Allow computers inside the network to access specific services outside the network. This is done by allowing traffic to pass from inside to outside if it is destined to a specific port (e.g. port 80 for web traffic).
- Do not allow computers inside the network to access unauthorised servers outside the network (for example, SIIT may decide that no-one inside can access FTP servers on the Internet).
- Do not allow any computers outside the network to access any servers inside the network. The only exceptions are to allow access to dedicated servers (e.g. the SIIT website).

Although the above cases can become quite complex in practice, very basic rules can be used to implement a simplified firewall that performs this functionality. You will do this in the Lab tasks.

2.4 Firewalls on Linux: iptables

`iptables` is a program on Linux that can be used to create a firewall. It allows the user to create a set of rules. Then when packets are received by the computer, the rules are processed. The packet is only sent if accepted by the rules.

`iptables` defines three basic classes of rules (or chains), based on where the packet is from/going to:

1. INPUT: processed if a packet is destined to this computer (e.g. the destination is this computer)
2. OUTPUT: processed if a packet is created to be sent by this computer (e.g. this computer is the source)
3. FORWARD: processed if a packet is to be forwarded by this computer (e.g. the packet is not destined to or from this computer, but this computer is acting as a router).

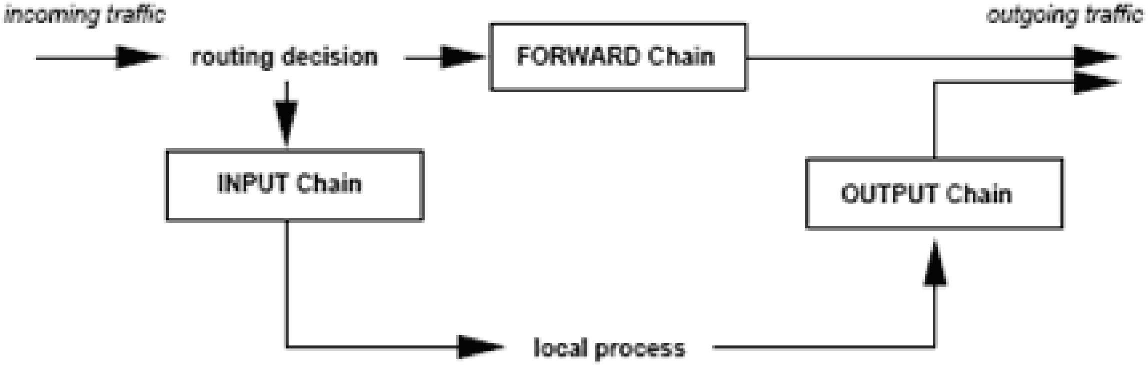


Figure 5: Chains in `iptables`

The most common way to use iptables is illustrated below (you need to execute with super-user privileges using sudo):

```
iptables -t filter -A chain [conditions]
```

where chain may be: INPUT, OUTPUT or FORWARD. Note that the `-t filter` is used by default, so you do not have to include it.

Some of the conditions are:

```
-s source_IP_address (e.g. 192.168.1.2)
-d destination_IP_address
-i input__interface (e.g. eth0)
-o output_ interface
-p protocol (e.g. tcp, udp, icmp)
-j action (e.g. ACCEPT, DROP)
```

Each protocol (e.g. `tcp`, `udp`) also have their own set of options: e.g. `--sport`, `--dport`.

There are many other options that you can read from the man pages.

So one way to create the 3rd rule/row in Figure 4 is:

```
iptables -A FORWARD -s 128.5.0.0/16 -p tcp -dport 25 -i eth1 -j DROP
```

The `-A` option specifies to *append* the rule to the table. You can also use a `-I` (*insert*) and `-D` (*delete*) options in a similar way.

To view all the rules in your table run:

```
iptables -L [chain]
```

where `[chain]` is INPUT, OUTPUT, FORWARD—if omitted then all rules are shown.

To delete (or flush) all rules in your table, run:

```
iptables -F [chain]
```

3 Tasks

3.1 Using iptables

This task aims to give you quick experience using the `iptables` command. The following task can be performed on your own computer. You should record the commands that you use to complete the tasks, and make any notes where necessary.

Task 1. *Flush all rules from the tables.*

Task 2. *Create a rule that will drop ICMP packets*

Task 3. *List the rules to check you have created it correctly.*

Task 4. *Try to ping another computer on the Lab network. Observe what happens.*

Task 5. *What happens when you use different chains in the rule created in part (2)? That is, observe and record what happens when you use INPUT, OUTPUT and FORWARD. Describe the effect of using different chains.*

Task 6. *Delete the rule using the D option (DO NOT flush the rules)*

Task 7. *Create a rule that will drop TCP packets destined to a specific computer on the Lab network (e.g. your neighbours computer).*

Task 8. *Try to connect (e.g. secure shell or web browser) to the other computer. Observe what happens.*

Task 9. *Flush all rules*

3.2 Setting up a Firewall

In pairs (e.g. using two computers, PC1 and PC2), complete the following steps:

1. Flush all rules.
2. Run a web server (Apache) and secure shell server (Openssh-server) on PC1. Test that they work.
3. Configure a firewall so that:
   (a) Any computer can connect to the web server on PC1
   (b) Only PC2 can connect to the SSH server on PC1
   (c) No computers can connect to any other servers (e.g. FTP, Email) on PC1
4. Record your rules and explain why you chose those rules.

A Notes

Record any additional notes from this lab here (e.g. important points made by the instructor, summary of things you learned, mistakes you made). You should use this in future labs, as well as in preparation for assessment items like exams.