The Linux Firewall

Laboratory Report in EDA491/DIT071 Network Security

Simon Fransson
Henrik Hugo

Group 12

Version no: 0.1

May 15, 2014

Contents

1 Introduction **1**

2 System Configuration and Requirements **2**

3 Firewall Configuration **4**
3.1 Chain INPUT 4
3.2 Chain OUTPUT 4
3.3 Chain LOG_DROP 4
3.4 Chain FORWARD 5
3.5 Chain CTH 5

4 Firewall Correctness **7**
4.1 Scan protection 7
4.2 Loopback 7
4.3 Ping flood 7
4.4 Stateful inspection 7
4.5 Local services 7

5 Discussion **8**
5.1 Further protection 8
5.2 Lessons learned 8

6 Conclusion **9**

References **10**

A Initial Firewall Configuration **11**

B Final Configuration Script **13**

C Answers to Assignment Questions **16**

1 Introduction

Today people take Internet and being connected for granted. It is such a common part of our everyday life that if Internet would stop working tomorrow many experts claim that it would result in a societal disaster as Danny Hillis speaks of in his TED talk [1]. This is also why it is very important to keep the systems that depend on Internet secure. Otherwise malicious users can launch attacks on these systems. Everyone can use Internet, it is very convenient and user-friendly. What people seem to forget is that the Internet is NOT secure. Without proper protection it is very easy for an attacker to gain access to a private network. A first perimeter of defense against attackers would be to implement a firewall. A firewall help protect against attackers by denying access and blocking security holes in systems that attackers could exploit.

This report is written to formulate and describe the problems and thoughts about how to set up a firewall using `iptables` on a Linux Redhat system. The purpose of this report is to introduce the reader to security threats and how these threats can partially be prevented using a firewall. It also aims to present the results obtained from the laboratory work done with `iptables` on Chalmers Linux Redhat systems. The configurations and thoughts about this laboratory work will also be presented.

The rest of the report is organized as follows: Section 2 presents the system configuration and requirements. Section 3 presents the firewall configuration done by the writers of this report. Section 4 aims to present reasoning as to why the specific configuration is correct. Section 5 provides some discussion regarding the firewall configuration, raising questions such as 'what could be done different?'. Section 6 presents the conclusions obtained from this labs and reflections upon the subject of firewalls.

2 System Configuration and Requirements

The system that was used for the laboratory work was a Linux Redhat computer which had the netfilter framework installed. It contains the program `iptables` to configure the firewall in Linux. The firewall was already configured to allow the NFS traffic required for being able to work in the user directory. The firewall was also configured to drop XMAS and NULL packets. This is shown in Listing 1. Otherwise, all traffic, in or out, was accepted. The system was set to run three different services, the names, service types and ports can be found in Table 1. The laboratory work consisted of configuring the firewall to meet certain security requirements stated by the lab PM. These requirements were implemented to make the system more secure. The specific requirements can be found in Table 2 below.

```
Chain INPUT (policy ACCEPT 9 packets, 2244 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 CTH        all  --  eth0   *       129.16.0.0/16        0.0.0.0/0
          /* Fix NFS traffic */
2        0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
          tcp flags:0x29/0x29
3        0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
          tcp flags:0x3F/0x00

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
          tcp flags:0x29/0x29
2        0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
          tcp flags:0x3F/0x00

Chain OUTPUT (policy ACCEPT 5 packets, 1207 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 ACCEPT     all  --  *      eth0    0.0.0.0/0            129.16.0.0/16

Chain CTH (1 references)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 ACCEPT     all  --  *      *       129.16.20.26         0.0.0.0/0
          ctstate RELATED,ESTABLISHED /* NFS server soleil */
2        0     0 RETURN     all  --  *      *       129.16.20.0/23       0.0.0.0/0
          /* Dont look at CE */
3        0     0 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0
          ctstate RELATED,ESTABLISHED /* Allow the rest to Chalmers */
```

Listing 1: Initial firewall configuration

Table 1: The services run on the lab system

Name	Service	Port
OpenSSH	SSH Server	SSH (22)
Apache	Web Server	http (8080)
Rpcbind	portmapper	sunrpc (111)

Table 2: The security requirements

Requirement	Description
Activate logging	Log packets that match specific rules
Default policy: DROP	Change the default policy of the firewall to drop all packets that does not match a chain in the firewall
Allow traffic from loopback	Make sure to let the firewall let traffic from loopback through as it can be used to troubleshoot network connection failures.
Allow traffic from host	Allow all outgoing traffic from the host, otherwise the host can not communicate.
Drop spoofing packets	Make sure to drop all packets from private networks, that the host is not a part of, as these addresses are not used by real hosts. Attackers can spoof their addresses by using these.
Stateful inspections	Make the firewall stateful by making it keep track of its current connections. In other words to let through connections that are already established.
Ping flood protection	Add some protection against ping flooding by limiting the number of packets the firewall will let through to one packet per second.
Application specific rules	Add rules to let hosts connect to the services displayed in Table 1.
Activate logging for unmatched packets	Add rules that log all packets that does not match any of the above requirements

3 Firewall Configuration

This chapter delves into the details of what the firewall configuration looked like when the lab was finished. The chapter will be divided into subchapters explaining each 'chain' such as INPUT, OUTPUT and so on.

3.1 Chain INPUT

This is the chain of the firewall that inspect incoming packets destined for the host. Rule number 1-3 have not been changed. From rule 4 and onwards are the new rules defined according to the firewall specification from the lab PM. Rule number 4 makes sure to accept all incoming traffic on the loopback interface. The rule can be found on line 69 in Appendix B. Rules 5-8 makes sure to drop and log all incoming traffic that comes from private/unused addresses as these may be spoofed packets. The rules can be found on lines 70-73 in Appendix B. Rule 9-10 limits the rate of incoming ICMP echo request packets to 1 packet per second. The rule is stated on line 79-80 in Appendix B. Rule 11 accepts all TCP connections that are already established or related to some other TCP connection. In other words, it makes sure that the connections which are already established with the host are allowed through the firewall. The rule can be found on line 83 in Appendix B. Rule 12-15 makes sure the firewall accepts connections for the services displayed in Table 1. Since sunrpc also accepts UDP connections there is an extra rule added for this, the rules can be found on lines 85-88 in Appendix B. Rule 16-18 makes sure to log all incoming TCP, UDP and ICMP traffic that did not match any of the other rules. The rules can be found on lines 90-92 in Appendix B.

3.2 Chain OUTPUT

This chain inspects all outgoing packets sent from the host. Rule 1 is not changed from the initial firewall script. Rule 2 accepts outoing loopback traffic, statement in script is found on line 68 in Appendix B. Rule 3-6 drop and logs all traffic sent to private/unused addresses. The statement of the rules can be found on lines 74-77 in Appendix B. Rule 7 accepts all other outgoing traffic, definition in the script can be found on line 82. Rule 8-10 make sure to log all outgoing TCP, UDP and ICMP traffic that did not match any of the other rules. The rules can be found on lines 94-96 in Appendix B.

3.3 Chain LOG_DROP

This chain was created only to achieve a clean and more readable script, the number of script lines would have otherwise been doubled (since one line for DROP and one line for LOG would have been required). The chain simply logs and then drops the packets that enter the chain. Definition can be found on lines 53-55.

3.4 Chain FORWARD

The forwarding chain is for packets that come to the host but is not destined for the host. The host will only forward these packets much like the behaviour of a router. The firewall can inspect these packets as well and create rules as for any other chain. It was not within the scope of the lab assignment to configure this chain and thus will this report not delve into any further details regarding this chain.

3.5 Chain CTH

This was a preconfigured chain done by the lab assistants. It simply makes sure to enable the network file storage traffic used for the home folders of the users. Without these rules it would not have been possible to reach the home folders and thus not being able to do the assignment. This chain was not in the scope of the lab and will thus not be explained any further.

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 CTH        all  --  em1    any     129.16.0.0/16        anywhere
          /* Fix NFS traffic */
2        0     0 DROP       tcp  --  any    any     anywhere             anywhere
          tcpflags: FIN,PSH,URG/FIN,PSH,URG
3        0     0 DROP       tcp  --  any    any     anywhere             anywhere
          tcpflags: FIN,SYN,RST,PSH,ACK,URG/NONE
4       16  1100 ACCEPT     all  --  lo     any     anywhere             anywhere
5        0     0 LOG_DROP   all  --  any    any     10.0.0.0/8           anywhere
6        0     0 LOG_DROP   all  --  any    any     172.16.0.0/12        anywhere
7        4   445 LOG_DROP   all  --  any    any     192.168.0.0/16       anywhere
8        0     0 LOG_DROP   all  --  any    any     link-local/16        anywhere
9        0     0 ACCEPT     icmp --  any    any     anywhere             anywhere
          icmp echo-request limit: avg 1/sec burst 5
10       0     0 DROP       icmp --  any    any     anywhere             anywhere
          icmp echo-request
11       0     0 ACCEPT     all  --  any    any     anywhere             anywhere
          state RELATED,ESTABLISHED
12       0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere
          tcp dpt:ssh
13       0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere
          tcp dpt:http-alt
14       0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere
          tcp dpt:sunrpc
15       0     0 ACCEPT     udp  --  any    any     anywhere             anywhere
          udp dpt:sunrpc
16       0     0 LOG        tcp  --  any    any     anywhere             anywhere
          LOG level warning
17       0     0 LOG        udp  --  any    any     anywhere             anywhere
          LOG level warning
18       0     0 LOG        icmp --  any    any     anywhere             anywhere
          LOG level warning

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

```
1        0     0 DROP       tcp  --  any    any     anywhere             anywhere
          tcpflags: FIN,PSH,URG/FIN,PSH,URG
2        0     0 DROP       tcp  --  any    any     anywhere             anywhere
          tcpflags: FIN,SYN,RST,PSH,ACK,URG/NONE

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 ACCEPT     all  --  any    em1     anywhere             129.16.0.0/16
2       16  1100 ACCEPT     all  --  any    lo      anywhere             anywhere
3        0     0 LOG_DROP   all  --  any    any     10.0.0.0/8           anywhere
4        0     0 LOG_DROP   all  --  any    any     172.16.0.0/12        anywhere
5       73  4728 LOG_DROP   all  --  any    any     192.168.0.0/16       anywhere
6        0     0 LOG_DROP   all  --  any    any     link-local/16        anywhere
7        0     0 ACCEPT     all  --  any    em1     anywhere             anywhere
8        0     0 LOG        tcp  --  any    any     anywhere             anywhere
          LOG level warning
9        0     0 LOG        udp  --  any    any     anywhere             anywhere
          LOG level warning
10       0     0 LOG        icmp --  any    any     anywhere             anywhere
          LOG level warning

Chain CTH (1 references)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 ACCEPT     all  --  any    any     129.16.20.26         anywhere
          state RELATED,ESTABLISHED /* NFS server soleil */
2        0     0 RETURN     all  --  any    any     129.16.20.0/22       anywhere
          /* Dont look at CE */
3        0     0 ACCEPT     all  --  any    any     anywhere             anywhere
          state RELATED,ESTABLISHED /* Allow the rest to Chalmers */

Chain LOG_DROP (8 references)
num   pkts bytes target     prot opt in     out     source               destination
1       77  5173 LOG        all  --  any    any     anywhere             anywhere
          LOG level warning
2       77  5173 DROP       all  --  any    any     anywhere             anywhere
```

Listing 2: Final firewall configuration

4 Firewall Correctness

This chapter goes into detail of the set of rule and describes why they work and how they were verified. Making sure they work as intended.

4.1 Scan protection

Since the order matter when configuring rules in the firewall it is important to verify their intent. Even though all chains have a default policy of drop it is necessary to explicitly drop certain packets. Packets matching the pattern of common scan methods are explicitly dropped. This is stated on lines 62-66 in Appendix B. Blocking these packets before reaching any accept-rules ensures the intent of the blockage, even if they are targeting an open port in the firewall. Verifying the effectiveness of the rules was done by performing an XMAS- and NULL-scan against the host using the NMAP tool. The scan failed to detect any of our open ports thus the scan protection is working.

4.2 Loopback

The next rule is allowing all traffic on the loopback interface. Allowing us to reach local services without restriction. The rules are stated in lines 68-69 in Appendix B. The loopback rules were tested using a local web server running on port 80. Reaching it locally was no issue but trying from the outside failed. Thus telling us it is working as intended.

4.3 Ping flood

While we want to allow the outside world to ping our host we also want to limit the amount of ping requests, because otherwise it could be used to launch a denial of service attack. This was done by only accepting one ping request per second and dropping the rest. Verification was done by sending 5 ping requests per second to the host. This resulted in a limited number of replies. We could also verify that the rules were working by seeing how many packets matched the rules in the firewall. The ping rules are stated on lines 79-80 in Appendix B.

4.4 Stateful inspection

It is important than the firewall allows us to initiate connections to the outside of the firewall. It is also necessary that we receive replies on connections initiated by us, making the firewall stateful. The rules allowing this are stated on lines 82-82 in Appendix B. Testing the rules was done by accessing google.com through a web browser. Thus ensuring that we can establish a TCP connection to google.com.

4.5 Local services

Testing SSH and web-access through port 8080 was as simple as accessing the services from the outside. Rules for the local services are stated on lines 85-88 in Appendix B.

5 Discussion

The final firewall configuration worked as intended, blocking unwanted traffic while still allowing the user to communicate outwards. The order of the rules seems logical by explicitly blocking unwanted packets, allowing wanted and dropping the rest. Although sufficient it may not be enough for the future. New threats and type of attacks may be discovered. Maintenance will be required to ensure proper protection.

5.1 Further protection

There are a few things that we could implement to improve the firewall even further. One of these is limiting the damage in the case of an intrusion by limiting outgoing traffic. Malicious programs, like botnets, tend to have patterns and use specific ports in their network communication. Known malicious ports are a good idea to block. Some malicious programs might start to try and participate in a distributed denial of service attack (DDoS). Limiting the amount of outgoing connection attempts might mitigate the attack before it reaches its target. Cyber attacks like DDoS are illegal in some countries and legal measures may apply to the owner of an infected computer or network.

Although a good practice to implement protection against these sort of attacks, it may be hard for the average user to know of such patterns and malicious ports. Measures which blocks specific communication by pattern may be better implemented on a gateway or router level. Thereby protecting all computers within a network and allowing for easier management.

5.2 Lessons learned

One thing we learned from this lab is that the order matters when defining your rules. Some rules may overrule others so it is also important to test your firewall configuration. Your firewall requires verification, the rules might not work as intended. New rules may affect older rules so it is important to keep a logical order in your configuration. In our configuration we started by defining what should be explicitly dropped and then what we want to allow and finally a default drop on packets not matching any rules.

Our final firewall configuration may not be complete but it is a good start for a safer connection to the internet. Other purposes, like running secure services on a server, will require a different firewall configuration.

6 Conclusion

Setting up proper security is not an easy task. It is easy to make a statement and say that a system is secure. However a system can never be secure, the real question is HOW secure the system is. The conclusions that can be drawn from the laboratory work are that a firewall is a very good first step into making a system more secure. It is fairly easy to configure with a software such as `iptables`. The firewall can help in blocking unwanted incoming connections. It helps in restricting what kind of applications that are allowed, tightening up eventual security holes. As shown in the report it can also help against attacks such as ping floods and probing of ports using XMAS or NULL scans. Considering that it takes about two hours to construct a firewall that achieves this it seems very unwise to not do it.

A firewall is one essential tool in minimizing security threats in your system. As shown in the report it helps not only denying access to unwanted users but can also log these attempts. It helps the administrator in observing the eventual security threats pointed towards the system. It is the first perimeter of defense and in the task of achieving a more secure system we believe it is the first thing to implement.

References

[1] D. Hillis. “The Internet could crash. We need a Plan B.” In: 2013.

A Initial Firewall Configuration

The initial configuration script of the firewall is shown in Listing 3.

```
#!/bin/bash -
#

MY_NETWORK="129.16.21.0/24"

# Replace the ip address here with the ip address for your computer.
# You can use the program "/sbin/ifconfig", or "/sbin/ip addr show"
# to obtain the correct address.
MY_HOST="129.16.21.XX"

# Network devices
IN=eth0
OUT=eth0

# Path to iptables, "/sbin/iptables"
IPTABLES="sudo␣/sbin/iptables"


######################################################################
### NOTE: FOLLOWING ROLES MUST BE AT THE TOP OF THIS CONFIGURATION ###
###       AND THEY SHOULD NOT BE MODIFIED IN ANY WAY(!)            ###
###       CHANGING ANY OF THESE RULES MAY RESULT IN THAT YOUR      ###
###       MACHINE FREEZES (AS YOUR NFS CONNECTION IS LOST TO YOUR  ###
###       HOME DIRECTORY).                                         ###
######################################################################

# Flushing all chains and setting default rules
$IPTABLES -P INPUT ACCEPT
$IPTABLES -P FORWARD ACCEPT
$IPTABLES -P OUTPUT ACCEPT
$IPTABLES -F
$IPTABLES -F CTH
$IPTABLES -X CTH

# Make sure NFS works (allow traffic to Chalmers)
# If NFS connection is lost, your machine will hang for eternity
$IPTABLES -N CTH
$IPTABLES -A CTH -s 129.16.20.26 -m state --state ESTABLISHED,RELATED -m comment --
    comment "NFS␣server␣soleil" -j ACCEPT
$IPTABLES -A CTH -s 129.16.20.0/23 -m comment --comment "Dont␣look␣at␣CE" -j RETURN
$IPTABLES -A CTH -m state --state ESTABLISHED,RELATED -m comment --comment "Allow␣the␣
    rest␣to␣Chalmers" -j ACCEPT

$IPTABLES -A INPUT -i $IN -s 129.16.0.0/16 -m comment --comment "Fix␣NFS␣traffic" -j CTH
$IPTABLES -A OUTPUT -o $OUT -d 129.16.0.0/16 -j ACCEPT

$IPTABLES -Z

########################################
### WRITE YOUR OWN RULES FROM HERE... ###
```

```
49 ########################################


51
   # Kill malformed packets
53 # Block XMAS packets
   $IPTABLES -A INPUT -p tcp --tcp-flags FIN,PSH,URG FIN,PSH,URG -j DROP
55 $IPTABLES -A FORWARD -p tcp --tcp-flags FIN,PSH,URG FIN,PSH,URG -j DROP
   # Block NULL packets
57 $IPTABLES -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
   $IPTABLES -A FORWARD -p tcp --tcp-flags ALL NONE -j DROP
59


61 echo "Done!"
```

Listing 3: Initial firewall configuration script

B Final Configuration Script

The final configuration script of the firewall is shown in Listing 4.

```
#!/bin/bash -
#

MY_NETWORK="129.16.23.0/24"

# Replace the ip address here with the ip address for your computer.
# You can use the program "/sbin/ifconfig", or "/sbin/ip addr show"
# to obtain the correct address.
MY_HOST="129.16.23.133"

# Network devices
IN=em1
OUT=em1

# Path to iptables, "/sbin/iptables"
IPTABLES="sudo␣/sbin/iptables"


######################################################################
### NOTE: FOLLOWING ROLES MUST BE AT THE TOP OF THIS CONFIGURATION ###
###       AND THEY SHOULD NOT BE MODIFIED IN ANY WAY(!)            ###
###       CHANGING ANY OF THESE RULES MAY RESULT IN THAT YOUR      ###
###       MACHINE FREEZES (AS YOUR NFS CONNECTION IS LOST TO YOUR  ###
###       HOME DIRECTORY).                                         ###
######################################################################

# Flushing all chains and setting default rules
$IPTABLES -P INPUT ACCEPT
$IPTABLES -P FORWARD ACCEPT
$IPTABLES -P OUTPUT ACCEPT
$IPTABLES -F
$IPTABLES -F CTH
$IPTABLES -X CTH

# Make sure NFS works (allow traffic to Chalmers)
# If NFS connection is lost, your machine will hang for eternity
$IPTABLES -N CTH
$IPTABLES -A CTH -s 129.16.20.26 -m state --state ESTABLISHED,RELATED -m comment --
    comment "NFS␣server␣soleil" -j ACCEPT
$IPTABLES -A CTH -s 129.16.20.0/22 -m comment --comment "Dont␣look␣at␣CE" -j RETURN
$IPTABLES -A CTH -m state --state ESTABLISHED,RELATED -m comment --comment "Allow␣the␣
    rest␣to␣Chalmers" -j ACCEPT

$IPTABLES -A INPUT -i $IN -s 129.16.0.0/16 -m comment --comment "Fix␣NFS␣traffic" -j CTH
$IPTABLES -A OUTPUT -o $OUT -d 129.16.0.0/16 -j ACCEPT

$IPTABLES -Z

########################################
### WRITE YOUR OWN RULES FROM HERE... ###
```

```
#########################################
$IPTABLES -F LOG_DROP
$IPTABLES -X LOG_DROP

$IPTABLES -N LOG_DROP
$IPTABLES -A LOG_DROP -j LOG
$IPTABLES -A LOG_DROP -j DROP

$IPTABLES -P INPUT DROP
$IPTABLES -P FORWARD DROP
$IPTABLES -P OUTPUT DROP
# Kill malformed packets
# Block XMAS packets
$IPTABLES -A INPUT -p tcp --tcp-flags FIN,PSH,URG FIN,PSH,URG -j DROP
$IPTABLES -A FORWARD -p tcp --tcp-flags FIN,PSH,URG FIN,PSH,URG -j DROP
# Block NULL packets
$IPTABLES -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
$IPTABLES -A FORWARD -p tcp --tcp-flags ALL NONE -j DROP

$IPTABLES -A OUTPUT -o lo -j ACCEPT                     #ACCEPT all loopback traffic
$IPTABLES -A INPUT -i lo -j ACCEPT                              #ACCEPT all loopback
    traffic
$IPTABLES -A INPUT -s 10.0.0.0/8 -j LOG_DROP            #LOG and DROP private address
    traffic
$IPTABLES -A INPUT -s 172.16.0.0/12 -j LOG_DROP         #LOG and DROP private address
    traffic
$IPTABLES -A INPUT -s 192.168.0.0/16 -j LOG_DROP        #LOG and DROP private address
    traffic
$IPTABLES -A INPUT -s 169.254.0.0/16 -j LOG_DROP        #LOG and DROP private address
    traffic
$IPTABLES -A OUTPUT -s 10.0.0.0/8 -j LOG_DROP           #LOG and DROP private address
    traffic
$IPTABLES -A OUTPUT -s 172.16.0.0/12 -j LOG_DROP        #LOG and DROP private address
    traffic
$IPTABLES -A OUTPUT -s 192.168.0.0/16 -j LOG_DROP       #LOG and DROP private address
    traffic
$IPTABLES -A OUTPUT -s 169.254.0.0/16 -j LOG_DROP       #LOG and DROP private address
    traffic

$IPTABLES -A INPUT -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT #Ping
     flood block
$IPTABLES -A INPUT -p icmp --icmp-type echo-request -j DROP
               #Ping flood block

$IPTABLES -A OUTPUT -o $OUT -j ACCEPT                   #ACCEPT outgoing traffic
$IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT #Accept all
    established TCP connections

$IPTABLES -A INPUT -p tcp --dport ssh -j ACCEPT         #ACCEPT application ssh
$IPTABLES -A INPUT -p tcp --dport 8080 -j ACCEPT        #ACCEPT application web server
$IPTABLES -A INPUT -p tcp --dport 111 -j ACCEPT         #ACCEPT application rpc-bind
$IPTABLES -A INPUT -p udp --dport 111 -j ACCEPT         #ACCEPT application rpc-bind
```

```
90 $IPTABLES -A INPUT -p tcp -j LOG                                    #LOG all
       incoming TCP connections that did not match the other rules
   $IPTABLES -A INPUT -p udp -j LOG                                    #LOG all
       incoming UDP connections that did not match the other rules
92 $IPTABLES -A INPUT -p icmp -j LOG                                   #LOG all
       incoming ICMP, connections that did not match the other rules

94
   $IPTABLES -A OUTPUT -p tcp -j LOG                                   #LOG all
       outgoing TCP connections that did not match the other rules
96 $IPTABLES -A OUTPUT -p udp -j LOG                                   #LOG all
       outgoing UDP connections that did not match the other rules
   $IPTABLES -A OUTPUT -p icmp -j LOG                                  #LOG all
       outgoing ICMP, connections that did not match the other rules
98
   echo "Done!"
```

Listing 4: Final firewall configuration script

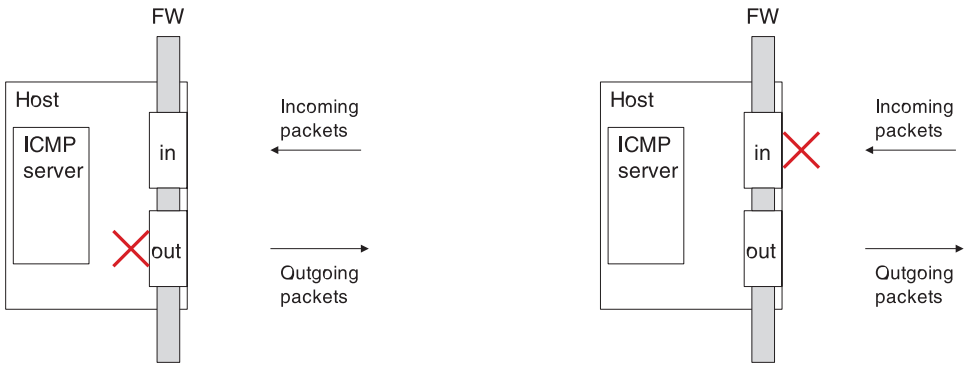
C Answers to Assignment Questions

Q1. After DROPping echo-reply packets on OUTPUT chain, what was the observed effect? Use Figure 1(a) to illustrate the path of the packets. Mark the path with arrows and use an X to mark the point where the packets are DROPped.

The host is receiving incoming echo requests and replies with and echo reply. However, outgoing echo replies are dropped when matching the newly created rule (as marked by the red X in figure 1(a)).

Q2. After DROPing echo-request packets on INPUT chain, what was the observed effect? How is this reaction different from the reaction achieved in Q1? Use Figure 1 to illustrate the path of the packets. Mark the path with arrows and use an X to mark the point where the packets are DROPped.

This time around incoming echo requests will be blocked going in to our host.



(a) Use this Figure to explain Q1

(b) Use this Figure to explain Q2

Figure 1: Figure to help you illustrate your thoughts regarding the packet flow in questions Q1 and Q2.

Q3. For each entry in the log, several information items are displayed. Some entries can be useful for creating new rules. Explain the items `IN`, `OUT`, `SRC`, `DST` and `PROTO` mean and why these might be useful.

IN and OUT denotes on what physical interface the packet entered and/or exited. SRC and DST represents the source and destination ip addresses respectively. PROTO is the protocol used on the network layer. This can be either TCP or UDP.

Q4. At this stage, with default policy set to DROP for all chains, would you consider the system secure? Would you consider it useful?

The system is indeed more secure with the default policy set to DROP, not allowing any traffic going either in or out. In the aspect of being useful, a local user would not be permitted to browse the web or even check their mail at this stage.

Q5. Assume instead that you used default policy ACCEPT, would you consider the system secure now? Would you consider it useful?

With a default policy set to ACCEPT the system would be less secure. Having no control over who has access or from where. Local users and processes would not have any issue connecting to the outside world making it convenient.

Q6. You just added some protection against flooding by limiting the number of packets the firewall will let through to 1 per second. Give two examples on how you can tell that you are protected!

Using the ping command we can try to send, for example, five ICMP echo-request packets per second. If the sequence number on the echo-reply packets are inconsistent we then know that some packets were dropped by the firewall. We can also check the current configuration to see how many packets matched the new rule with `sudo iptables -vL`. The output should list a number of packets