IK2206 – Internet Security and Privacy

Firewall & IP Tables

Group Assignment

Following persons were members of group C and authors of this report:

Name:	Christoph Moser	Name:	Thomas Galliker
Mail:	chmo@kth.se	Mail:	galliker@kth.se
P-Nr:	850923-T513	P-Nr:	860711-T773

Setup

•Output of ping when verifying connectivity.

Ping from C1 to C3:

```
root@iptables-C1:~# ping 10.2.0.2
PING 10.2.0.2 (10.2.0.2) 56(84) bytes of data.
64 bytes from 10.2.0.2: icmp_req=1 ttl=63 time=1.63 ms
64 bytes from 10.2.0.2: icmp_req=2 ttl=63 time=1.42 ms
64 bytes from 10.2.0.2: icmp_req=3 ttl=63 time=1.64 ms
```

Ping from C3 to C1:

```
student@iptables-C3:~$ ping 192.168.2.2
PING 192.168.2.2 (192.168.2.2) 56(84) bytes of data.
64 bytes from 192.168.2.2: icmp_req=1 ttl=63 time=1.68 ms
64 bytes from 192.168.2.2: icmp_req=2 ttl=63 time=1.52 ms
64 bytes from 192.168.2.2: icmp_req=3 ttl=63 time=1.87 ms
```

Nmap Enumeration

•How does nmap detect active hosts using:

•Link Layer
Nmap offers capabilities to issue ARP ping requests. As by the NMAP Reference Guide, ARP discovery is activated by default since layer-2 addresses are mostly used for further scan methods.

•Network Layer
Different layer-3 scanning options are offered: ICMP ping, IP protocol ping.

•Transport layer
Different layer-4 scanning options are offered: TCP-SYN ping, TCP-ACK ping, UDP ping.

Sources: [1], [2]

•What are the advantages and disadvantages of each type of scanning?

	Advantages	*Disadvantages*
Link Layer Scan	• Can possibly be used to determine the vendor and therefore the type of the end device.	• Scan only scan within the same subnet. • Result depends on network layer scan.
Network Layer Scan	• Gives information about the logical address of the end device.	• Can be blocked by simple firewalls.
Transport Layer Scan	• Gives information about what applications is possibly run on the end device.	• Can be blocked by stateful inspection firewalls.

Sources: [1], [3]

•Which parameters did you use to locate the server?

```
nmap -sP 10.2.0.0/16

Starting Nmap 5.21 ( http://nmap.org ) at 2011-12-08 12:52 CET
Nmap scan report for 10.2.0.1
Host is up (0.00082s latency).
MAC Address: 00:16:3E:3E:02:03 (Xensource)
Nmap scan report for 10.2.0.2
Host is up.
Nmap scan report for 10.2.130.40
Host is up (0.00087s latency).
MAC Address: 00:16:3E:3E:02:10 (Xensource)
```

•What is the address of the server?
The IP address is 10.2.130.40.

•How long did it take?
1331.98 seconds

•How many addresses did you scan?
We scanned the whole class-B subnet 10.2.0.0/16, means, 2^{16} = 65536 addresses.

Nmap scanning

•What command did you use for TCP discovery?

`nmap -sT 10.2.130.40`

•What command did you use for UDP discovery?

`nmap -sU 10.2.130.40`

•UDP discovery is much slower than TCP discovery. Why?

There are several reasons, why a UDP scan takes longer than a TCP scan: Open and filtered ports rarely send any response, leaving Nmap to time out. Awaiting a timeout may cost much time. Furthermore, the operating system of the target machine limits the number of “ICMP unreachable messages” to avoid flooding the network with useless packets.

TCP scan completed in 14.37 seconds.
UDP scan completed in 1095.42 seconds (~60 ports/sec)

Source: [3]

•List all open TCP services

22/tcp open ssh
53/tcp open domain
80/tcp open http

•List all open UDP services

53/udp open domain

•What is the difference between Open, Filtered, Unfiltered and Closed ports?

<u>Open:</u> An Open port accepts either TCP connection or UDP datagrams.

<u>Closed:</u> A close port is able to receive and response to Nmap packets, but there is no service bound to the port which is listening.

<u>Filtered:</u> Nmap is not able to detect if the port is open because NMAP packets are filtered, for example by a firewall or router-rules.

<u>Unfiltered:</u> Nmap is able to send and receive Nmap packets to and from a port, but Nmap is incapable of determining if the port is open or closed.

Sources:[3]

Nmap Service Identification

•What operating system does nmap detect?

```
nmap -O --osscan-guess 10.2.130.40
Linux 2.6.19 - 2.6.31 (96%)
(...)
```

Source: [4]

•How are the services identified?

```
nmap -sV 10.2.130.40

Nmap scan report for 10.2.130.40
Host is up (0.00039s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 5.5p1 Debian 4ubuntu4 (protocol 2.0)
53/tcp open  domain  ISC BIND 9.7.1-P2
80/tcp open  http    Apache httpd 2.2.16 ((Ubuntu))
MAC Address: 00:16:3E:3E:02:10 (Xensource)
Service Info: OS: Linux

Nmap done: 1 IP address (1 host up) scanned in 19.48 seconds
```

Source: [5]

•Are these sane guesses?

The guesses seem to be very accurate. If we compare the identified services, we can find out that all of them must be part of an Ubuntu Linux derivate.

•What other methods can be used to check the operating system and service implementations of an unknown server?

SNMP: The Simple Network Management Protocol offers a good way to get system information from remote systems. The TCP/IP MIB-2 (see RFC1213) can be used to gather information about basic networking settings, including the operating system of the target machine (MIB field `sysDescr`).

```
snmpget public 10.2.130.40 system.SysDescr.0
system.sysDescr.0 = Linux version 2.6.35-22-server
```

Source: [6]

Ping TTL: Different operating systems use different (default) values as Time To Live (TTL) in their TCP/IP configuration. There are lists with operating systems and their default TTL values on the internet. Each ping replies the TTL value of the remote system.

Example ping output where the target is a Linux/Unix system:

```
Reply from 10.2.130.40: bytes=32 time=3ms TTL=64
Reply from 10.2.130.40: bytes=32 time=3ms TTL=64
Reply from 10.2.130.40: bytes=32 time=3ms TTL=64
```

Example ping output where the target is a Windows system:

```
Reply from 10.2.130.40: bytes=32 time=3ms TTL=128
Reply from 10.2.130.40: bytes=32 time=3ms TTL=128
Reply from 10.2.130.40: bytes=32 time=3ms TTL=128
```

Note: This approach is, of course, just to make a rough guess.

Source: [7]

WBEM: Web Based Enterprise Management is a service that provides basic system management information. There are many professional system management products that are based upon WBEM. If security is not restricted, it could be possible to get system information from remote systems.

9 Basic IPTables

9.1 Block icmp pings

•Explain the order in which the rules are evaluated

Each packet that arrives at the firewall is compared to the configured firewall rules, starting at the first rule. The firewall continues this comparation process until the packet matches a rule. The order of the rules is therefore important.

Source: [8]

•Show your iptables rules (iptables -vL) where you drop ICMP echo packets.

```
root@iptables-C2:~# iptables -vL
Chain INPUT (policy ACCEPT 12 packets, 696 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
  101  8484 DROP       icmp --  any    any     anywhere             anywhere
icmp echo-request

Chain OUTPUT (policy ACCEPT 6 packets, 504 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

9.2 Reject icmp pings

Following rules were used to accept ICMP echo-requests from inside to outside and echo-replies back from outside to inside. All other ICMP traffic is rejected.

```
iptables -A FORWARD -p icmp --icmp-type echo-request -s 192.168.2.0/24 -d 10.2.0.0/16 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type echo-request -s 10.2.0.0/16 -d 192.168.2.0/24 -j REJECT

iptables -A FORWARD -p icmp --icmp-type echo-reply -s 192.168.2.0/24 -d 10.2.0.0/16 -j REJECT
iptables -A FORWARD -p icmp --icmp-type echo-reply -s 10.2.0.0/16 -d 192.168.2.0/24 -j ACCEPT
```

•List of ping logs showing everything works correctly

```
root@iptables-C2:~# iptables -vL
Chain INPUT (policy ACCEPT 2604 packets, 124K bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 1403 packets, 118K bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 REJECT     icmp --  any    any     192.168.2.0/24       10.2.0.0/16
         icmp echo-reply reject-with icmp-port-unreachable
  101  8484 REJECT     icmp --  any    any     10.2.0.0/16          192.168.2.0/24
      icmp echo-request reject-with icmp-port-unreachable
   54  4536 ACCEPT     icmp --  any    any     10.2.0.0/16          192.168.2.0/24
      icmp echo-reply
  232 19488 ACCEPT     icmp --  any    any     192.168.2.0/24       10.2.0.0/16
         icmp echo-request
    0     0 DROP       all  --  any    any     anywhere             anywhere

Chain OUTPUT (policy ACCEPT 2431 packets, 125K bytes)
 pkts bytes target     prot opt in     out     source               destination
```

•Can you ping from the external host to the internal interface on the firewall?
Yes, I can.

•Why/why not?
The firewall has 3 different policies that can be used to place rules:

- "INPUT": Packet is going to be locally delivered. (N.B.: It does not have anything to do with processes having a socket open. Local delivery is controlled by the "local-delivery" routing table: `ip route show table local`.)
- "FORWARD": All packets that have been routed and were not for local delivery will traverse this chain.
- "OUTPUT": Packets sent from the machine itself will be visiting this chain.

In the steps above, we just cared about the FORWARD policy. We didn't create any INPUT/OUTPUT policies. This means that everyone can still send ICMP packets to the interfaces of the firewall (=INPUT policy) and the firewall itself can send ICMP packets to other participants (=OUTPUT policy).

→ Important hint: Each policy should have a "drop/reject all packets" instruction as last rule! Otherwise, packets might slip through the mesh of policies.

•Can this have any security implications?
The outside interface of the firewall is highly exposed. If there is a vulnerability in the operating system of the firewall, there is a certain risk of attacks. At least the INPUT policy should be configured more restrictive.

•What is the difference between rejecting and dropping blocked traffic?
Rejecting responses with a failure message whereas dropping does not take further actions but destroying the received packet. From the sender perspective, dropping looks as if there was no such end point available.

•What are the advantages of rejecting resp. dropping?

- The only advantage (from the perspective of a sender) of using "REJECT" instead of "DROP" is that it gets to know whether the recipient is online or whether we use an invalid IP address. "DROP" makes it appear as if there is no such IP online.
- REJECT generates additional overhead to the network. This can be misused to flood the network. Rejecting traffic lets the user's computer respond much more quickly, which makes the server seem more responsive. A typical example is DNS servers that are down without sending rejects. It makes the resolvers on clients to hang for up to few minutes.
- Services with limited intelligence may try to resend dropped packets again and again. This can also burden the network.
- DROP is considered as more secure since attackers have to wait for a timeout to exceed while probing ports.

Derived from: [9]

9.3 Logging

Following commands were used to implement the LOGREJECT policy:

```
iptables -A LOGREJECT -j LOG --log-prefix "Ping rejected by Firewall: " --log-level 7
iptables -A LOGREJECT -j REJECT

iptables -A FORWARD -p icmp --icmp-type echo-request -s 192.168.2.0/24 -d 10.2.0.0/16 -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type echo-request -s 10.2.0.0/16 -d 192.168.2.0/24 -j LOGREJECT

iptables -A FORWARD -p icmp --icmp-type echo-reply -s 192.168.2.0/24 -d 10.2.0.0/16 -j LOGREJECT
iptables -A FORWARD -p icmp --icmp-type echo-reply -s 10.2.0.0/16 -d 192.168.2.0/24 -j ACCEPT
```

•A sample from the system log showing what you have logged

```
Dec  8 17:42:59 iptables kernel: [22269.633034] Ping dropped by FirewallIN=eth1 OUT=eth0 SRC=10.2.0.2
DST=192.168.2.2 LEN=84 TOS=0x00 PREC=0x00 TTL=63 ID=0 DF PROTO=ICMP TYPE=8 CODE=0 ID=982 SEQ=8
Dec  8 17:43:00 iptables kernel: [22270.634549] Ping dropped by FirewallIN=eth1 OUT=eth0 SRC=10.2.0.2
DST=192.168.2.2 LEN=84 TOS=0x00 PREC=0x00 TTL=63 ID=0 DF PROTO=ICMP TYPE=8 CODE=0 ID=982 SEQ=9
Dec  8 17:43:01 iptables kernel: [22271.635947] Ping dropped by FirewallIN=eth1 OUT=eth0 SRC=10.2.0.2
DST=192.168.2.2 LEN=84 TOS=0x00 PREC=0x00 TTL=63 ID=0 DF PROTO=ICMP TYPE=8 CODE=0 ID=982 SEQ=10
Dec  8 17:43:02 iptables kernel: [22272.637363] Ping dropped by FirewallIN=eth1 OUT=eth0 SRC=10.2.0.2
DST=192.168.2.2 LEN=84 TOS=0x00 PREC=0x00 TTL=63 ID=0 DF PROTO=ICMP TYPE=8 CODE=0 ID=982 SEQ=11
Dec  8 17:43:03 iptables kernel: [22273.639637] Ping dropped by FirewallIN=eth1 OUT=eth0 SRC=10.2.0.2
DST=192.168.2.2 LEN=84 TOS=0x00 PREC=0x00 TTL=63 ID=0 DF PROTO=ICMP TYPE=8 CODE=0 ID=982 SEQ=12
```

•List your complete set of rules (iptables -vL) at this point

```
root@iptables-C2:~# iptables -vL
Chain INPUT (policy ACCEPT 2666 packets, 126K bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 1403 packets, 118K bytes)
 pkts bytes target     prot opt in     out     source               destination
   54  4536 ACCEPT     icmp --  any    any     10.2.0.0/16          192.168.2.0/24
      icmp echo-reply
  232 19488 ACCEPT     icmp --  any    any     192.168.2.0/24       10.2.0.0/16
         icmp echo-request
   13  1092 LOGREJECT  icmp --  any    any     10.2.0.0/16          192.168.2.0/24
      icmp echo-request
    0     0 LOGREJECT  icmp --  any    any     192.168.2.0/24       10.2.0.0/16
         icmp echo-reply
    0     0 REJECT     all  --  any    any     anywhere             anywhere
            reject-with icmp-port-unreachable

Chain OUTPUT (policy ACCEPT 2467 packets, 129K bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain LOGREJECT (2 references)
 pkts bytes target     prot opt in     out     source               destination
   13  1092 LOG        all  --  any    any     anywhere             anywhere
            LOG level info prefix `Ping dropped by Firewall'
   13  1092 REJECT     all  --  any    any     anywhere             anywhere
            reject-with icmp-port-unreachable
```

Derived from: [10]

10 Building a firewall

10.1 Network permissions

```
iptables -I INPUT  -s 192.168.2.0/24 -j ACCEPT
iptables -I OUTPUT -d 192.168.2.0/24 -j ACCEPT
```

10.2 Permitting a service

•What kind of security advantage does a setup with a SSH terminal server offer?

SSH (Secure Shell) offers a secure remote shell. Transmitted session data is encrypted. Telnet, on the contrary, sends session data (incl. username/password) unencrypted to the remote system.

Source: [11]

•What kind of security disadvantage does a setup with a SSH terminal server introduce?

- Port forwarding can also introduce security problems. The SSH server doesn't allow detailed configuration of what forwarding is allowed from what client to what server etc.
- When a user is authenticated by password, the client's RSA identity is not verified (against ssh_known_hosts).

Source: [12]

•List the rules you used to setup the firewall as a terminal server for ssh.

```
iptables -I INPUT 1 -p tcp --dport 22 -j ACCEPT
iptables -I OUTPUT 1 -p tcp --sport 22 -j ACCEPT
```

10.3 Stateful Filtering

```
iptables -I FORWARD 1 -s 192.168.2.0/24 -d 10.2.0.0/16 -p tcp -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT

iptables -I FORWARD 1 -s 10.2.0.0/16 -d 192.168.2.0/24 -p tcp -m state --state RELATED,ESTABLISHED -j ACCEPT
```

10.4 FTP Forwarding

```
iptables -I FORWARD 1 -s 10.2.0.0/16 -d 192.168.2.2 -p tcp --dport 21 -m state --state NEW,ESTABLISHED -j ACCEPT

iptables -I FORWARD 2 -s 10.2.0.0/16 -d 192.168.2.2 -p tcp --dport 20 -m state --state ESTABLISHED -j ACCEPT

iptables -I FORWARD 3 -s 192.168.2.2 -d 10.2.0.0/16 -p tcp --sport 21 -m state --state ESTABLISHED -j ACCEPT
```

Basic FTP commands can be found here: [13]

10.5 Blocking ports

```
iptables -I FORWARD 5 -s 192.168.2.0/24 -p tcp -m multiport --dport 139,445
iptables -I FORWARD 6 -s 192.168.2.0/24 -p udp -m multiport --dport 137,138
```

Your Rule Set

•How did you verify that the firewall works as intended?

Requirements for new firewall rules should never be implemented before defining positive and negative test cases. Iptables (-vL) allows to see which rules have processed how many packets. Test cases for firewall rules (e.g. establishing an ftp connection and transmitting data) indicate whether a certain rule was activated or default rule (e.g. “drop all”) was used.

•List your final set of firewall rules

```
iptables -vL
Chain INPUT (policy ACCEPT 2674 packets, 126K bytes)
pkts bytes target     prot opt in     out     source               destination
552 54995 ACCEPT     tcp  --  any    any     anywhere             anywhere            tcp
dpt:ssh
181 19147 ACCEPT     all  --  any    any     192.168.2.0/24       anywhere
141  4772 DROP       all  --  any    any     anywhere             anywhere

Chain FORWARD (policy ACCEPT 1403 packets, 118K bytes)
pkts bytes target     prot opt in     out     source               destination
126  7301 ACCEPT     tcp  --  any    any     10.2.0.0/16          192.168.2.2         tcp
dpt:ftp state NEW,ESTABLISHED
41  2244 ACCEPT     tcp  --  any    any     10.2.0.0/16          192.168.2.2         tcp
dpt:ftp-data state ESTABLISHED
85  7285 ACCEPT     tcp  --  any    any     192.168.2.2          10.2.0.0/16         tcp
spt:ftp state ESTABLISHED
68  5022 ACCEPT     tcp  --  any    any     192.168.2.2          10.2.0.0/16         tcp
spt:ftp-data state RELATED,ESTABLISHED
0     0            tcp  --  any    any     192.168.2.0/24       anywhere
multiport dports netbios-ssn,microsoft-ds
0     0            udp  --  any    any     192.168.2.0/24       anywhere
multiport dports netbios-ns,netbios-dgm
0     0 ACCEPT     tcp  --  any    any     10.2.0.0/16          192.168.2.0/24      state
RELATED,ESTABLISHED
0     0 ACCEPT     tcp  --  any    any     192.168.2.0/24       10.2.0.0/16         state
NEW,RELATED,ESTABLISHED
17   964 DROP       all  --  any    any     anywhere             anywhere

Chain OUTPUT (policy ACCEPT 2467 packets, 129K bytes)
pkts bytes target     prot opt in     out     source               destination
315 42104 ACCEPT     tcp  --  any    any     anywhere             anywhere            tcp
spt:ssh
320 30005 ACCEPT     all  --  any    any     anywhere             192.168.2.0/24
25  1500 DROP       all  --  any    any     anywhere             anywhere
```

Feedback

•References

[1] Andrew J. Bennieston, NMAP - A Stealth Port Scanner, http://nmap.org/bennieston-tutorial/

[2] Nmap Reference Guide, Chapter 15, Host Discovery, http://nmap.org/book/man-host-discovery.html

[3] Nmap Reference Guide, Chapter 15, Port Scanning Techniques, http://nmap.org/book/man-port-scanning-techniques.html

[4] Nmap Reference Guide, Chapter 8, Remote OS Detection, http://nmap.org/book/osdetect-usage.html

[5] Nmap Reference Guide, Chapter 15, Service and Version Detection, http://nmap.org/book/man-version-detection.html

[6] K. McCloghrie, M. Rose, RFC-1213, Management Information Base for Network Management of TCP/IP-based internets: MIB-II, http://tools.ietf.org/html/rfc1213

[7] Noah Davids, Initial TTL values for different operating systems, http://noahdavids.org/self_published/TTL_values.html

[8] Guy Shipard, an article about iptable rules, http://gr8idea.info/os/tutorials/security/iptables6.html

[9] BASH Cures Cancer blog, "netcat, iptables, and why you should drop packets instead of rejecting them", http://bashcurescancer.com/netcat-iptables-and-why-you-should-drop-packets-instead-of-rejecting-them.html

[10] Ubuntu Help, IptablesHowTo, https://help.ubuntu.com/community/IptablesHowTo#More_detailed_Logging

[11] Wikipedia article about “Secure Shell”, http://en.wikipedia.org/wiki/Secure_Shell

[12] Hemant Shah, “The Secure Shell”, http://uniforumchicago.org/slides/ssh/SSH.pdf

[13] Dave Lozinski, “The Basics of FTP”, http://www.davelozinski.com/tutorials/ftp/index.php?1323369980079

•Suggested improvements to the lab system

Iptables is fine to show the basic purpose of a firewall but it would be fairly awkward to run a complex firewall of a big company using iptables. Why not using Checkpoint or Cisco firewalls in this lab?

•Suggested improvements to the lab instructions

It could be helpful if you provide some ftp commands that can be used to check the firewall rules.

•Time estimation

11h/pers.