CSC-NETLAB

Packet filtering with Iptables

Group Nr	
Name1	
Name2	
Name3	
Date	
Instructor's Signature	

Table of Contents

1 Goals

The goal of this lab is to introduce you to the concepts of packet filtering, firewalls and setting up iptables in Linux. You will learn to use the packet filtering interface and a tool to explore firewalls and systems on the Internet.

The lab is also a refresher of your basic linux and networking skills. Make sure you feel you have a firm understanding of everything in this lab. You will need it in the future.

Before you begin this lab, consult the ip-tables tutorial [1] (additional help and tutorials are found at [2]) and the documentation for nmap [3]. The lab is much easier if you use these as a reference.

Any part, concept or question from this lab may be used in the exam.

2 Introduction

The laboratory system consists of a virtualized system where each group has access to 3 virtual hosts. These will be running Linux (Ubuntu Server 10.10 to be exact). You will receive the root password for these computers.

Some of the tasks in the lab can be done at the same time. If you get stuck consider if you can continue with another part.

3 Preparation questions

The following questions should be answered before you begin with the laboration. All questions can be answered with the references [1] and [3]. You should familiarize yourself with these documents.

3.1 Nmap

In the lab network there are essentially three methods at different layers in the OSI-model that can be used to discover which hosts that are online with Nmap. Which are they, what are their limitations? How do you get nmap to do a scan (what flags)?

1:_____
_____

2:_____
_____

3:_____
_____

3.2 Iptables chains

Rules in iptables are always parts of a chain. Chains can either be user created or one of the built-in chains. In this laboration we will only use three chains: INPUT, FORWARD and OUTPUT.

Explain which packets will pass through each of these chains:

INPUT:_____
_____
_____

FORWARD:_____
_____
_____

OUTPUT:_____
_____
_____

3.3 Operators

To change a chain you need to use an operator. Explain the following operators, their short form and what they do:

Append:_____
_____
Insert:_____
_____
Delete:_____
_____
Replace:_____
_____
List:_____
_____
Flush:_____
_____

3.4 Filters

Filters are used to choose which packets match a rule. Each filter is used to create matches. Matches are listed in section 10 of [1]. Some matches requires modules to be loaded with the -m <module> to be available. Explain the following matches and how they can be used:

-p:_____
-s:_____
-d:_____
-i:_____
-o:_____
--sport:_____
--dport:_____

3.5 Jump targets

Jump targets are used to decide what to do with a packet once it has matched a rule. Targets are listed in section 11 of [1]. Explain the following targets:

ACCEPT:_____
DROP:_____
REJECT:_____
LOG:_____

Milestone: Report your progress to a lab assistant. _____

4 Getting started

User accounts for the xen account and your virtualized hosts are handed out at the start of the lab. You will be assigned a set of hosts. The groups are numbered, so: A=0, B=1, C=2

etc. You will need this numbering to assign IP addresses.

group # (X):_____
Number:_____
Login(xen): student-YOUR_GROUP_LETTER
password(xen):_____
Login(Virtual machines): student
password(virtual machines): m0rris

The IP addresses for your network is listed in the network topology map (found in Appendix A), where X is the number of your group. Please note the virtual hosts will not have access to the general Internet (unlike your workstation)

5 Connecting to the virtual hosts

Login to the virtualization host, xen.netlab.csc.kth.se via ssh using the "student-#" user with the password supplied to you.

Example:
silver:~ pehrs$ ssh student-A@xen.netlab.csc.kth.se
You will end up in the virt-shell, letting you administrate your virtual machines.

Login to the three virtual machine iptables-#.1 (internal), iptables-#.2 (firewall), and iptables-#.3 (external). See appendix A for the network topology. You can login by first choosing the instance (enter the instance name), booting them (if they are not running) and then use the console command like this:

```
Welcome to the NetLab environment!
Last login: Fri Nov  5 19:54:54 2010 from silver.ssvl.kth.se
Netlab Virt-Shell
virt-shell> list
You may control the following Xen instances:

        iptables-A.1
        iptables-A.2
        iptables-A.3

(Use 'control' to take control of a particular instance.)
virt-shell> control iptables-A.1
Controlling: iptables-A.1
virt-shell[iptables-A.1]> boot
Booting instance: iptables-A.1
Domain iptables-A.1 created from /etc/libvirt/qemu/iptables-A.1

Use 'console' to see the bootup messages.
virt-shell[iptables-A.1]> console

Running console for iptables-A.1 - exit with Ctrl+]
(You might need to press return a couple of times to see activity.)

Connected to domain iptables-A.1
Escape character is ^]

Ubuntu 10.10 iptables ttyS0
```

iptables login:

In the virt-shell there are a few additional commands that can be useful: shutdown lets you restart your vm. If your vm gets messed up to such a

degree that it's no longer usable you can try reimage. It will wipe the image and replace it with a clean one, destroying all work you have done.

Hints when working with virtual serial consoles:

- Use 3 terminals for the 3 different virtual hosts.
- Do not enable multiple consoles to the same virtual host (X.1, X.2 or X.3). This will mess up your consoles!
- For line-wraps to work correctly the size of your terminal window has to match the virtual console. You can either make sure your terminal window has the default size (24*80) or update the virtual console window size (using **`# stty columns X rows Y`**). **`# stty -a`** prints current settings and lets you get the size of a window not connected on a serial console.
- If your virtual console seems strange, try the “clear” command. It will try to reset the console.

6 Getting root

The first step is to get root access on the host. Use the **`sudo`** command:

```
student@iptables:~$ sudo bash
root@iptables:~#
```

Here the **#** means you have root access on the host. While it's a bad habit (normally you should use **`sudo`** instead of a root shell) almost all commands in this lab requires root access, so this will save you from typing **`sudo`** constantly.

To ensure you are on the right machine, set the **`hostname`** as soon as you have logged in, and then logout/login again.

```
root@iptables:~# hostname iptables-A1
root@iptables:~# exit
student@iptables:~$ logout

Ubuntu 10.10 iptables-A1 ttyS0

iptables-A1 login: student
Password:
...
student@iptables-A1:~$
```

7 Setting up interfaces

First of all you need to configure your network interfaces. You do this by manually assigning them an IP address and netmask from the map of your network. The map can be found in the end of the instructions. Remember your group number and insert it into the map where you see x (for example C.1 has ip 192.168.2.2)

Under Linux the interface configuration is done with the ifconfig tool. Make sure the configuration is applied on the correct interface (eth0, eth1) in each host.

Example:
```
# ifconfig eth0 10.18.0.33 netmask 255.255.0.0
```

Check the configuration with **`ifconfig`** without any arguments.

8 Setting up routing

At this point it should be possible to ping between directly connected hosts (verify this!). To ensure the data can be routed through a host you have to setup forwarding.

First verify so packet forwarding is enabled on the firewall host. This is done by checking so the file **/proc/sys/net/ipv4/ip_forward** contains "**1**". If it's not so use **echo 1 > /proc/sys/net/ipv4/ip_forward** to enable forwarding.

Then use the command **route add default gw <address>** to give both the end hosts a default route to the central host in your network. The address has to be an address of a directly connected interface on the central host.

Check the routing table with the **route** command without arguments. Verify so you can ping between the two hosts at the end of your network.

Milestone: Report your progress to a lab assistant. _____

9 Detecting server capabilities with Nmap

Nmap is a free open source utility commonly used for network exploration and security auditing. Nmap can determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics.
To view the nmap help, run the command:
```
# man nmap
```

9.1 Enumeration

Use **`nmap`** to enumerate your outside network from the external host. Try to locate the IP address of the server that is connected to the network. Note: The scan can take a lot of time, so test with a smaller network and make sure you can scan the whole subnet in a reasonable time.

Tip:
Since this will take some time, pipe the output from **`nmap`** to a file and put the process in the background so you can continue your work. Use the **>** operator to pipe the output to a file and the **`&`** operator to background the process.

```
# nmap [flags and operators to nmap] > [output_file] &
```

While waiting for the scan to complete you can continue with the lab from step 10.

Which parameters did you use to locate the server?

_____

What is the address of the server?

_____

How long time did it take?

_____

How many addresses did you scan?

_____

If you get an error message while scanning: Make sure you are root. If you still can't find the server: Verify your routing and netmasks!

9.2 Service discovery

Discover which services are running on the server using **nmap** and complete the table listed on the next page. Scan for both TCP and UDP and compile a list.

What command did you use for TCP discovery?

_____

What command did you use for UDP discovery?

_____

UDP discovery is much slower than TCP discovery. Why?

_____

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

_____

Type	Port	Service
TCP	22	SSH Daemon

9.3 OS discovery

Nmap can also be used to check which operating system and services a host is running. Use OS detection to check what OS is used on the server and use service detection to check the versions of the services on the server.

What operating system does nmap detect?

_____

How are the services identified?

_____

Are these sane guesses?

_____

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

_____

Milestone: Report your progress to a lab assistant. _____

10 IPTables

Please notice that all the work in iptables will be done at the central host (x.2) in your network, the firewall.

To view the iptables help, run the command:
```
# man iptables
```

It may also be a good idea to open up a web browser and load [1] as a reference.

To list the current state, run the command:
```
# iptables -vL
```

10.1 Block icmp pings

Ping the internal host from the external one. What happens? Now execute the following line on the firewall:

```
# iptables -A FORWARD -p icmp --icmp-type echo-request -j DROP
```

Ping the host again. What happens? Why?
Has the output from **`iptables -vL`** changed?

To remove the rule from the chain, first list the rules again:

```
# iptables --line-numbers -L FORWARD
```

To find the line number of the rule you just added, and then remove it with:

```
# iptables -D FORWARD <line number>
```

10.2 Reject icmp pings

IPTABLES rules will only match if all the conditions in the rule are true. Create a new rule which REJECTS all ICMP echo-requests from the external network to your internal network.

Verify so you can **ping** from the internal host to the external one.

Verify so the external host can not **ping** the internal one.

Verify so the firewall can **ping** both hosts.

Can you **ping** from the external host to the internal interface on the firewall?

_____

Why/why not?

_____

Can this have any security implications?

_____

What is the difference between rejecting and dropping blocked traffic?

_____

What are the advantages of blocking and what are the advantages of dropping?

_____

10.3 Logging and limits

One of the important tasks for a firewall is to log rejected packets, making it easier to trace attacks on the network. A rule can be created with the jump target LOG to save information to the system log. Iptables logs directly to the syslog system.

Create a rule that logs all dropped **ping** messages. Make sure the string "Ping dropped by <your name>:" is written in the log message. Check the firewall log so the traffic is saved. This can easily be done with

```
# tail -f /var/log/messages
```

The module limit can be used to limit how often a rule can be triggered. Use the limit module to make sure no more than 5 pings each minute are saved to the system log. Documentation of the limit module can be found in [1].

Milestone: Report your progress to a lab assistant. _____

11 Building a simple firewall

Each chain has a default policy, which details what do to with a packet that does not match any rules. Set all chains to the policy DROP and remove all old rules.

Make sure SSHD is running on all hosts by executing

```
# service ssh restart
```

Verify so you can not send any data through or to the firewall. Use **nmap** and **ping**.

11.1 Network permissions

Now it's time to start allowing some carefully chosen traffic through the firewall.

Create a rule which allows all traffic originating from the internal network that arrives to the internal interface (eth0) to reach the firewall host.

Create a rule which allows all traffic originating from the firewall to reach the internal network.

Make sure you now can reach the internal network from the firewall, and the firewall from the internal network.

11.2 Permitting a service

SSH (Secure shell) is a common service for remote management of firewalls. Create a rule which allows SSH traffic directly to the firewall from the external network. SSH runs on port 22 and uses only TCP. Verify so you can connect to the firewall with SSH from the other hosts.

Verify so you can not connect directly from your external host to the internal host with ssh.

Verify so you can connect to the internal host if you first connect to the firewall and then connect from the firewall to the internal host.

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

_____

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

11.3 Stateful filtering

In most cases we want to allow hosts on the internal network to connect to the Internet. However, we do not wish hosts on the Internet to be able to connect to hosts inside the firewall. For some protocols, such as TCP this can be done STATELESS, due to the three way handshake needed to create a connection. By blocking the initial packets we can prevent connections.

However, stateless filtering breaks a large number of protocols, for example UDP based protocols can not easily be allowed through in only one direction. Even if UDP is stateless we need a stateful firewall to properly handle it. Some protocols such as FTP also breaks if connections from the Internet are completely denied.

Examine the module **state**. Use this module to create a match that allows the hosts on the inside of the firewall to establish connection to the outside. Allow all data that belongs to these connections through the firewall. Keep blocking all other connection attempts from the outside. Documentation of the **state** module can be found in [1]

11.4 Opening a port

Frequently you want a computer on the outside of the network to have access to a single service on the inside. Make sure that external computers can reach the echo service (port 7), both on UDP and TCP from the outside.

11.5 Blocking a port

Sometimes you do not want your internal users to be able to connect to the internet on a specific port at all. One commonly blocked port is 135 (*Microsoft's RPC implementation*) which is used for windows file sharing. Make sure your firewall blocks all traffic on port 135 (both TCP and UDP)

11.6 Verifying your setup

Now you only have to verify your setup. Should do this in a systematic manner, making sure all rules works as correctly. Use the counters in iptables -vL to make sure the packets match the rules you expect.

To test if a port is open you can open one with netcat and try to connect with telnet.
On the reciving host run:

```
# nc -l <port>
```

On the sending host run:

```
# telnet <ip> <port>
```

At this point you should have the following rules active, in this order of priority:

- Port 135 is blocked.

- Connections on port SSH are allowed to the firewall host from the outside
- Connections on port echo are allowed to the internal host from the outside.
- Connections directly to/from the firewall are allowed from the internal networks
- Connections from the inside are allowed out
- Connections from the outside are blocked

How do you verify that the firewall works as intended?

Milestone: Report your progress to a lab assistant. _____

12 Build your own firewall

Consider your home network, the services you offer and the service you require from the Internet. Which protocols do you actually use? Which can you block?

- Make a list of the TCP and UDP ports you need to open for incoming traffic.
- Make a list of the TCP and UDP ports you need to open for outgoing traffic.
- Make a list of the types of ICMP packets you need to send and receive.

A list of the protocols with an assigned IP number can be found at [4].

Do you need any additional network or transport protocols such as CLNP, IPX/SPX, AppleTalk, DCCP or SCTP? How do you enable them?

Milestone: Report your progress to a lab assistant. _____

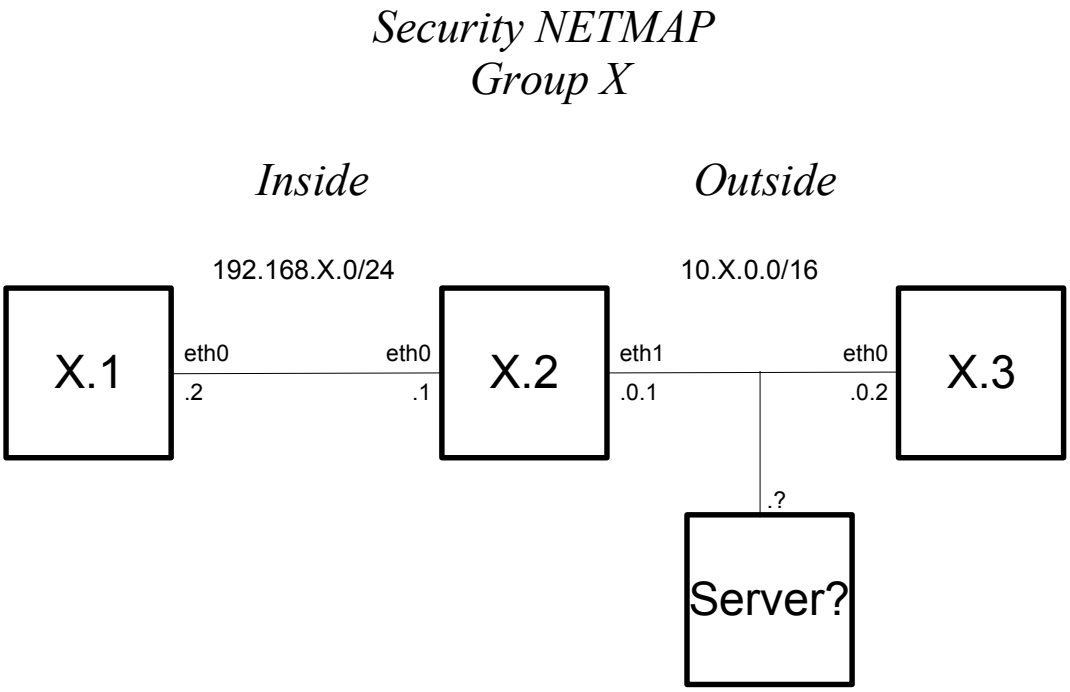
13 Cleanup

Remove all files that you have created (if any) and remove all firewall rules. Set the default policy to accept. Log out before you close the consoles.

14 References

[1] http://www.frozentux.net/documents/iptables-tutorial/
[2] http://www.netfilter.org/documentation/index.html
[3] http://insecure.org/nmap/man
[4] http://www.networksorcery.com/enp/topic/ipsuite.htm
[5] http://slacksite.com/other/ftp.html

Appendix A: LAB network map



KTHNOC security_netmap.odp rev 1.3