DAT151 – OBLIG8

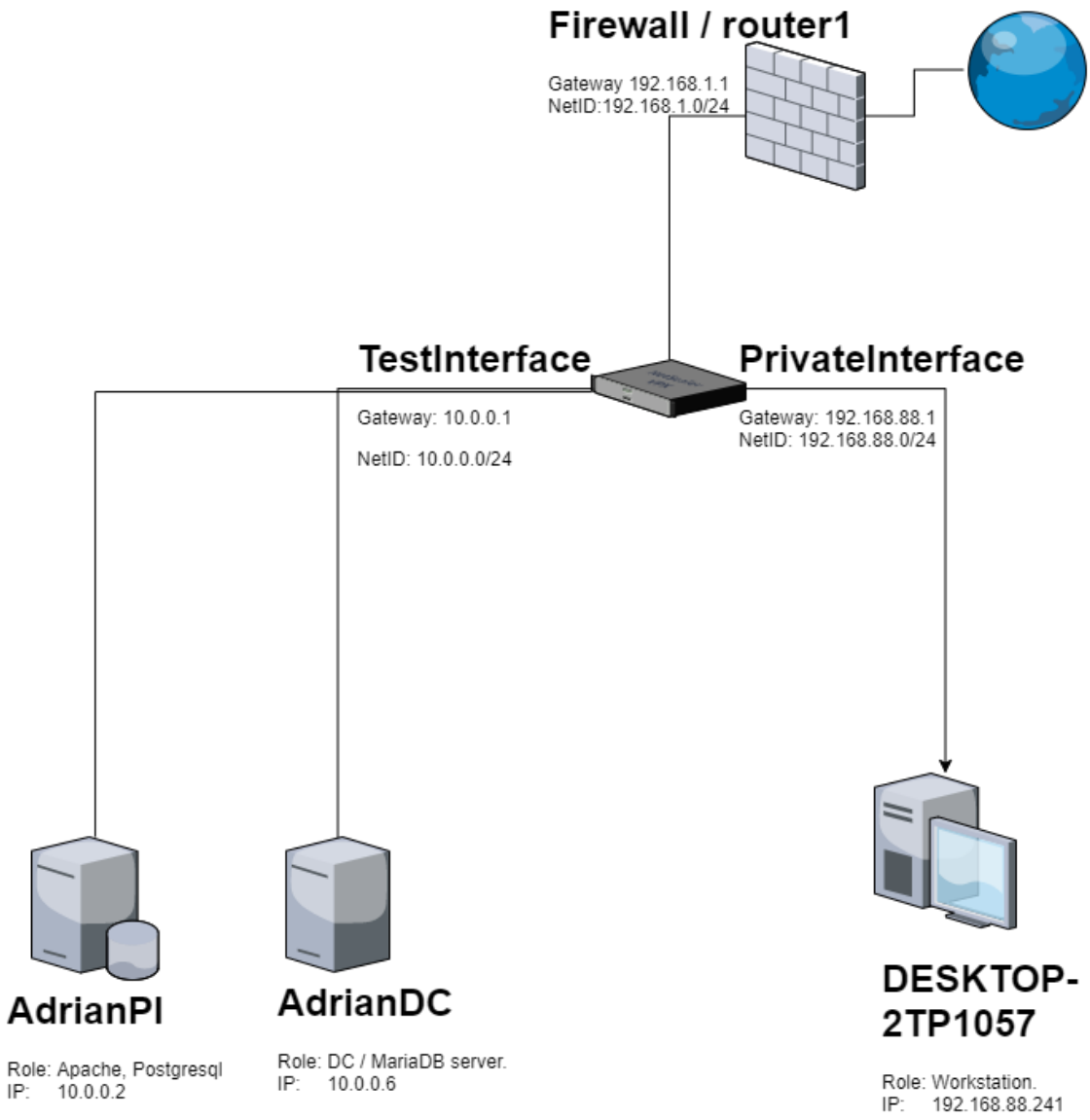
Security and virtualization

Adrian Mortensen & Kamil Sosna

Innhold

Task1: Firewall

Setup explanation



The desktop / workstation is where most of the work will be done from. AdrianPI has a mounted display so this will be the one we work on the most in terms of security since its easier to fix mistakes.

a) Blocking everything.

Rule:

```
pi@adrianpi:~ $ sudo iptables -A INPUT -j DROP
```

Small explanation; -A is what direction. I chose to do input since this will block all connections to the client. This will also stop it from receiving any packet from the internet. Meaning even if it asks for instance a website the answer will be blocked. Using -j DROP makes the rule silently drop the package not giving back a response at all. This could make a potential attacker think that they have the wrong IP address or that there is some other network issue.

Testing:

```
C:\Users\adria>ping 10.0.0.2


Pinging 10.0.0.2 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 10.0.0.2:
      Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

b) Blocking all SSH except Workstation

Rule:

```
iptables -A INPUT -p tcp -s 192.168.88.241 --dport 22 -j ACCEPT
iptables -A INPUT -p tcp -m tcp --dport 22 -j DROP
```

Small explanation; -p is what type of protocol used (here TCP) -s is source address or mask, --dport is destination port and -j is Jump to target. (Here accept, letting the connection through)

The 2nd rule is pretty similar to the previous one just that don't specify a source address and drops all incoming on the port.

Testing:

```
adrian@DESKTOP-2TP1O57:~$ ssh pi@10.0.0.2

pi@10.0.0.2s password:
Linux adrianpi.adrian.local 4.14.79-v7+ #1159 SMP Sun Nov 4 17:50:20 GMT
2018 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Feb 28 00:46:45 2020 from 192.168.88.241
pi@adrianpi:~ $


pi@AdrianDC:~ $ ssh pi@10.0.0.2
ssh: connect to host 10.0.0.2 port 22: Connection timed out
```

c) Making sure b) persists through restart.

This can be done multiple ways. One is to just add these rules to crontab as root.

But one can also make a small script for ease of edit that is ran by crontab so that you don't fill up your crontab page with iptables rules.

Script:

```
#!/bin/bash
iptables -A INPUT -p tcp -s 192.168.88.241 --dport 22 -j ACCEPT
iptables -A INPUT -p tcp -m tcp --dport 22 -j DROP
```

Making a folder on /FirewallRules (Not ideal location but this is temporary)

```
pi@adrianpi:/FirewallRules $ sudo chmod 755 Firewall.sh
pi@adrianpi:/FirewallRules $ sudo crontab -e
crontab: installing new crontab
```

In the sudo crontab:

```
@reboot sudo /FirewallRules/Firewall.sh
```

Testing:

pi@adrianpi:/FirewallRules $ sudo shutdown -r now

After restart:

```
Last login: Fri Feb 28 01:30:26 2020 from 192.168.88.241
pi@adrianpi:sudo iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A INPUT -s 192.168.88.241/32 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j DROP
pi@adrianpi:~ $
```

(Here we can see the rules being there and that I did connect through SSH. Still testing through AdrianDC:

```
pi@AdrianDC:~ $ ssh pi@10.0.0.2
ssh: connect to host 10.0.0.2 port 22: Connection timed out
```

Task2: SELinux

Task done on a CentOs 8 virtual machine.

1) Checking if SELinux is enabled or not

```
[admo@EXAMPLE ~]$ getenforce
Enforcing
```

```
[admo@EXAMPLE ~]$ sestatus

SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      31
```

You can turn it off by going to the configuration file.

```
[admo@EXAMPLE ~]$ sudo nano /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes $
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Changing from “enforcing” to disabled would turn it off.

However here it is on.

2) Running semanage login

```
[admo@EXAMPLE ~]$ sudo semanage login -l

Login Name           SELinux User         MLS/MCS Range
Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
```

semanage login controls the mapping between a Linux User and the SELinux User. It can be used to turn on confined users. For example you could define that a particular user or group of users will login to a system as the user_u user. You can also see the SELinux users in this list (default, root)

while with the option "-l" it lists the objects.

3) Context

a) Create a file setest

```
[admo@EXAMPLE ~]$ touch /tmp/setest
```

b) Check the context of setest

```
[admo@EXAMPLE ~]$ ls -Z /tmp/setest
unconfined_u:object_r:user_tmp_t:s0 /tmp/setest
```

Output just gives the default context that is given to a new file. It is not configured yet.

c) Change the context of setest to user_home_t

```
[admo@EXAMPLE ~]$ chcon -t user_home_t /tmp/setest
```

d) Show the context of setest.

```
[admo@EXAMPLE ~]$ ls -Z /tmp/setest
unconfined_u:object_r:user_home_t:s0 /tmp/setest
```

e) Show the context of the current user

```
[admo@EXAMPLE ~]$ id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

f) How to check if httpd has privilege to read the file /etc/passwd

First have a look at the httpd context

```
[admo@EXAMPLE ~]$ ps axZ | grep httpd
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 14436 pts/2 R+   0:0
```

Here we can see the type is unconfined_t. While /etc/passwd has:

```
[admo@EXAMPLE ~]$ ls -Z /etc/passwd
system_u:object_r:passwd_file_t:s0 /etc/passwd
```

For httpd to have access to passwd it would have to be a similar type to passwd_file_t and it is not.

Task3: Virtualisation

KVM is known as Kernel based Virtual Machine because when we install KVM package then KVM module is loaded into the current kernel and turns our Linux machine into a hypervisor. Install a non-graphical Virtual machine using KVM kickstart file located at

https://eple.hib.no/dat151/kvm_centos8.ks

Use the following address to download the Centos8

http://centos.uib.no/8-stream/BaseOS/x86_64/

```
[admo@s1 ~]$ sudo virt-install -n CentOsO8 -f /vm/CentOsVirtO8.img,size=10
--os-variant rhel8.0 -l http://centos.uib.no/8-stream/BaseOS/x86_64/os -r
1024 --nographics --extra-args
'ks=https://eple.hib.no/dat151/kvm_centos8.ks,console=ttyS0,115200n8
serial'
WARNING  KVM acceleration not available, using 'qemu'

Starting install...
Retrieving file vmlinuz...
|  7.8 MB  00:00:01
Retrieving file initrd.img...
|   59 MB  00:00:11
Allocating 'CentOsVirtO8.img'
|   10 GB  00:00:00
Connected to domain CentOsO8
```