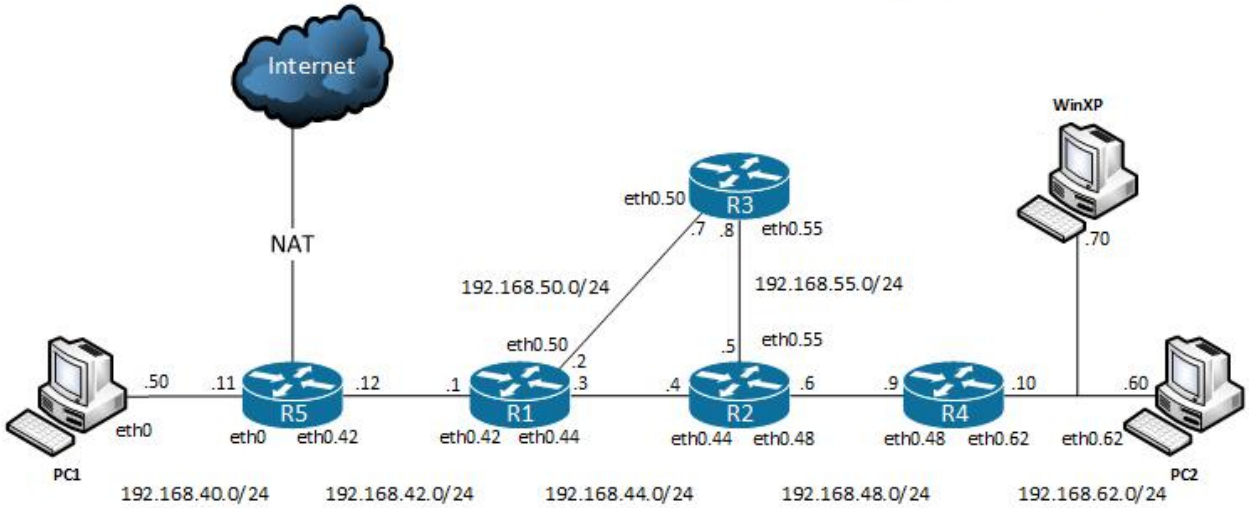
Generation and Analysing Network Attacks using Scapy

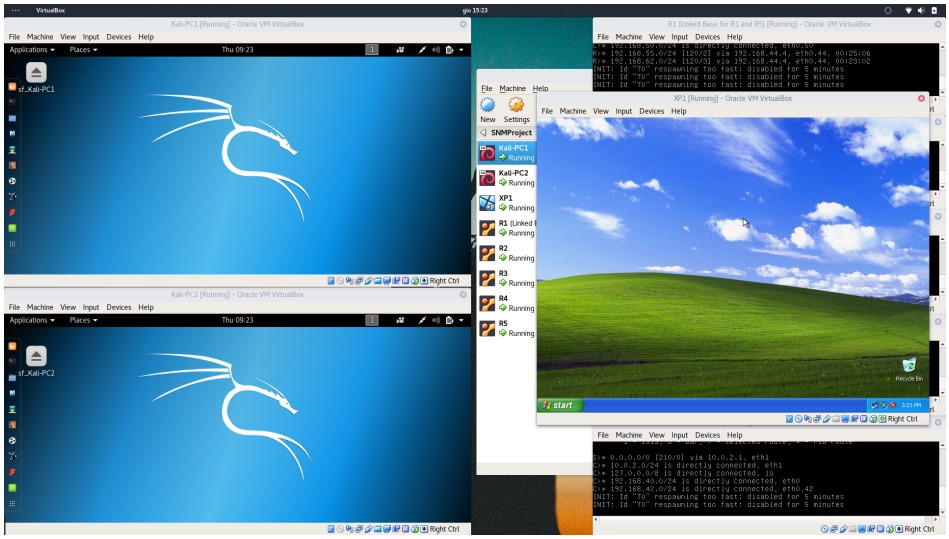
Project of the Secure Network Management course by DECAMP

Pietro Prandini - Stefano Romanello @ UniPD

December 28, 2018

1 The configuration used





1.1 Devices Configuration

R5

The Router5 is a clone of the Router1. The network of this router is composed by two enabled adapters:

- Adapter 1: Internal Network (Name: intnet);
- Adapter 2: NAT Network (Name: NatNetwork).

After the start of the machine it is setted with this commands:

```
# Configuring the router 5 (R5)
## Basic configuration
configure
load /live/image/R1/lab16
commit

## Setting the new ethernet eth0 address
delete interfaces ethernet eth0 address 192.168.40.1/24
set interfaces ethernet eth0 address 192.168.40.11/24
commit

## Setting the new ethernet eth0.42 address
delete interfaces ethernet eth0 vif 44
delete interfaces ethernet eth0 vif 50
set interfaces ethernet eth0 vif 42 address 192.168.42.12/24
commit

## Enablig RIP
set protocols rip interface eth0.42
set protocols rip interface eth0
set protocols rip network 192.168.40.0/24
set protocols rip network 192.168.42.0/24
set protocols rip redistribute connected
set protocols rip timers timeout 35
commit

## Configuring NAT
set interfaces ethernet eth1 address dhcp
commit

edit nat source rule 10
        set translation address masquerade
        set source address 192.168.32.0/19
        set outbound-interface eth1
commit
exit

## NAT routing
set protocols rip redistribute static
set protocols rip network 0.0.0.0/0
commit

exit
```

R1

```
# Configuring the router 1 (R1)
## Basic configuration
configure
load /live/image/R1/lab16
commit

## Considering the new router R5
delete interfaces ethernet eth0 address 192.168.40.1/24
commit
set interfaces ethernet eth0 vif 42 address 192.168.42.1/24
commit

## Enablig the RIP protocol
```

```
set protocols rip interface eth0.42
set protocols rip interface eth0.44
set protocols rip interface eth0.50
commit
```

R2

```
# Configuring the router 2 (R2)
## Basic configuration
configure
load /live/image/R2/lab16_rip
commit
exit
```

R3

```
# Configuring the router 3 (R3)
## Basic configuration
configure
load /live/image/R3/lab16_rip
commit
exit
```

R4

```
# Configuring the router 4 (R4)
## Basic configuration
configure
load /live/image/R4/lab16_rip
commit
exit
```

Kali-PC1

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
        address 192.168.40.50
        netmask 255.255.255.0
        gateway 192.168.40.11
```

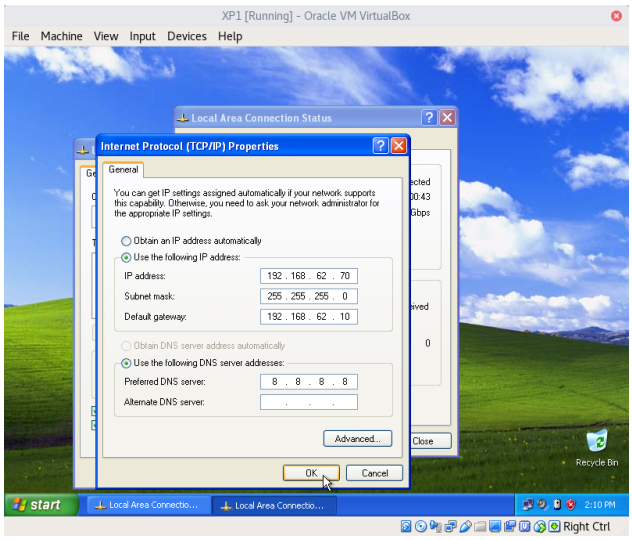
```
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
```

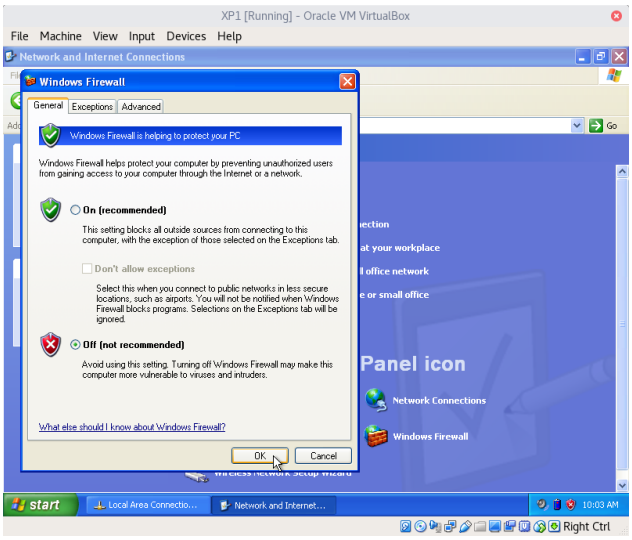
Kali-PC2

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet manual
up ifconfig eth0 up

auto eth0.62
iface eth0.62 inet static
        address 192.168.62.60
        netmask 255.255.255.0
        gateway 192.168.62.10
        vLAN-raw-device eth0
```

```
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
```

XP1





1.2 Test to the configuration

Bash version of a test.

```
#!/usr/bin/env bash

# Availability of each device

echo "
-------------------------------------------------------------------------------
ping to 192.168.40.11 (R5)"
ping -c 3 192.168.40.11

echo "
-------------------------------------------------------------------------------
ping to 192.168.42.12 (R5)"
ping -c 3 192.168.42.1
```

```
echo "
-----------------------------------------------------------------------------
ping to 192.168.42.1 (R1)"
ping -c 3 192.168.42.12

echo "
-----------------------------------------------------------------------------
ping to 192.168.50.2 (R1)"
ping -c 3 192.168.50.2

echo "
-----------------------------------------------------------------------------
ping to 192.168.44.3 (R1)"
ping -c 3 192.168.44.3

echo "
-----------------------------------------------------------------------------
ping to 192.168.44.4 (R2)"
ping -c 3 192.168.44.4

echo "
-----------------------------------------------------------------------------
ping to 192.168.55.5 (R2)"
ping -c 3 192.168.55.5

echo "
-----------------------------------------------------------------------------
ping to 192.168.48.6 (R2)"
ping -c 3 192.168.48.6

echo "
-----------------------------------------------------------------------------
ping to 192.168.50.7 (R3)"
ping -c 3 192.168.50.7

echo "
-----------------------------------------------------------------------------
ping to 192.168.55.8 (R3)"
ping -c 3 192.168.55.8

echo "
-----------------------------------------------------------------------------
ping to 192.168.48.9 (R4)"
ping -c 3 192.168.48.9

echo "
-----------------------------------------------------------------------------
ping to 192.168.62.10 (R4)"
ping -c 3 192.168.62.10

echo "
-----------------------------------------------------------------------------
ping to 192.168.40.50 (PC1)"
ping -c 3 192.168.40.50

echo "
-----------------------------------------------------------------------------
ping to 192.168.62.60 (PC2)"
ping -c 3 192.168.62.60
```

```
echo "
-------------------------------------------------------------------------------
ping to 192.168.62.70 (XP1)"
ping -c 3 192.168.62.70

# Testing the NAT
echo "
-------------------------------------------------------------------------------
ping to www.google.com (NAT)"
ping -c 3 www.google.com
```

Now it's presented a scapy program used to test if the network was working properly before the attacks.

```
#! /usr/bin/env python
from scapy.all import *

def check_availability(target):
    ans,unans = sr(IP(dst=target)/ICMP())
    if ans:
        print(target + ' is reachable, summary:')
        ans.summary()
        return ans,unans
    else:
        print(target + ' is not reachable, summary:')
        unans.summary()
        return ans,unans

# Availability of each device
target = "192.168.40.50"
check_availability(target)

target = "192.168.40.11"
check_availability(target)

target = "192.168.42.1"
check_availability(target)

target = "192.168.42.12"
check_availability(target)

target = "192.168.44.3"
check_availability(target)

target = "192.168.44.4"
check_availability(target)

target = "192.168.48.6"
check_availability(target)

target = "192.168.48.9"
check_availability(target)

target = "192.168.50.2"
check_availability(target)

target = "192.168.50.7"
check_availability(target)

target = "192.168.55.8"
check_availability(target)

target = "192.168.55.5"
```

```
check_availability(target)

target = "192.168.62.10"
check_availability(target)

target = "192.168.62.60"
check_availability(target)

target = "192.168.62.70"
check_availability(target)

# Testing the NAT
target = "www.google.com"
check_availability(target)
```

2 Reconnaissance Attacks

2.1 IP Spoofing

2.2 Introduction

2.2.1 SCAPY program

```
#! /usr/bin/env python
from scapy.all import *
```

2.2.2 Attacker's messages

2.2.3 Attack's result

2.2.4 How to protect the network

2.3 No Flags Set

2.4 Introduction

2.4.1 SCAPY program

```
#! /usr/bin/env python
from scapy.all import *
```

2.4.2 Attacker's messages

2.4.3 Attack's result

2.4.4 How to protect the network

3 DoS Attacks

3.1 ICMP Redirect

3.2 Introduction

3.2.1 SCAPY program

```
#! /usr/bin/env python
from scapy.all import *
```

3.2.2 Attacker's messages

3.2.3 Attack's result

3.2.4 How to protect the network

3.3 Ping of Death

3.4 Introduction

3.4.1 SCAPY program

```
#! /usr/bin/env python
from scapy.all import *
```

3.4.2 Attacker's messages

3.4.3 Attack's result

3.4.4 How to protect the network