CYBERSECURITY DEPARTMENT CCCY 323

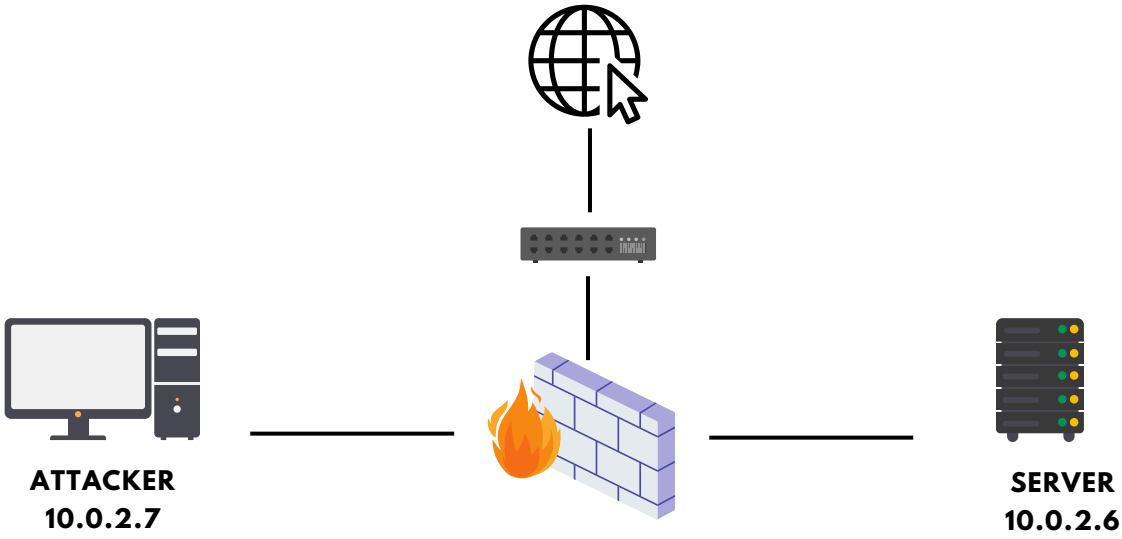
NETWORK SECURITY PROJECT

PRESENTED BY:
RIMAN BANDAR 2006588
ARWA YUSEF 2006599
KHULOD JABER 2006926
HADEEL ALSULMI 2006616

PART 1: NETWORK SETUP

TASK 1: SERVER SETTING

1. NETWORK SETUP OF THE PROJECT



2. SCREENSHOTS OF YOUR VM CONFIGURATIONS.

ATTACKER

Preview

General

Name: attacker
Operating System: Ubuntu (64-bit)

System

Base Memory: 2048 MB
Processors: 6
Boot Order: Floppy,Optical,Hard Disk
Acceleration: Nested Paging,KVM Paravirtualization

Display

Video Memory: 128 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage

Controller: IDE
IDE Secondary Device 0: [Optical Drive] Empty
Controller: SATA
SATA Port 0: attacker.vdi (Normal,25.00 GB)

Audio

Host Driver: Default
Controller: ICH AC97

Network

Adapter 1: Intel PRO/1000 MT Desktop (Bridged Adapter,en0: Wi-Fi)

USB

USB Controller: OHCI,EHCI
Device Filters: 0 (0 active)

Shared folders

None

Description

None

SERVER

Preview

General

Name: server
Operating System: Ubuntu (64-bit)

System

Base Memory: 2048 MB
Processors: 6
Boot Order: Floppy,Optical,Hard Disk
Acceleration: Nested Paging,KVM Paravirtualization

Display

Video Memory: 128 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage

Controller: IDE
IDE Secondary Device 0: [Optical Drive] Empty
Controller: SATA
SATA Port 0: s.vdi (Normal,25.00 GB)

Audio

Host Driver: Default
Controller: ICH AC97

Network

Adapter 1: Intel PRO/1000 MT Desktop (Bridged Adapter,en0: Wi-Fi)

USB

USB Controller: OHCI,EHCI
Device Filters: 0 (0 active)

Shared folders

None

Description

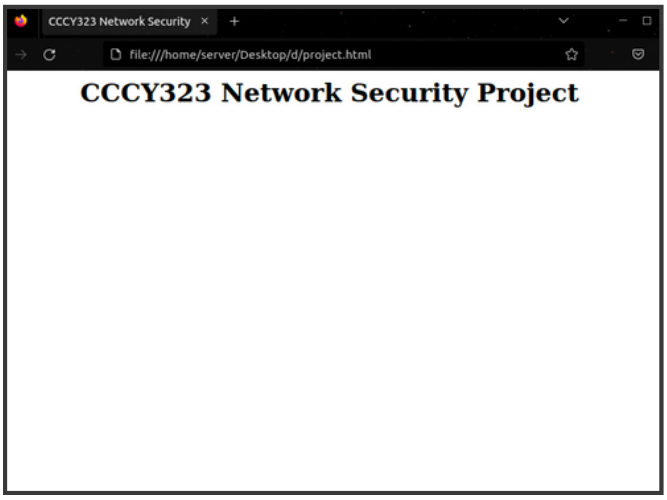
None

3. SCREENSHOTS OF YOUR WEB PAGE.

➤ XAMPP INSTALLATION AND SETUP

```
server@server:~$ cd Downloads
server@server:~/Downloads$ ls
xampp-linux-x64-8.2.0-0-installer.run
server@server:~/Downloads$ sudo ./xampp-linux-x64-8.2.0-0-installer.run
```

➤ WEB PAGE



TASK 2: ATTACKER VM

4.SCREENSHOTS OF THE NMAP TOOL INSTALLATION COMMANDS.

```
attacker@attacker:~$ sudo apt-get install nmap
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  liblinear4 libssh2-1 lua-lpeg nmap-common
Suggested packages:
  liblinear-tools liblinear-dev ncat ndiff zenmap
The following NEW packages will be installed:
  liblinear4 libssh2-1 lua-lpeg nmap nmap-common
```

```
attacker@attacker:~$ nmap --version
Nmap version 7.92 ( https://nmap.org )
Platform: x86_64-pc-linux-gnu
Compiled with: liblua-5.3.6 openssl-3.0.5 libssh2-1.10.0 libz-1.2.11 libpcre-8.3
9 libpcap-1.10.1 nmap-libdnet-1.12 ipv6
Compiled without:
Available nsock engines: epoll poll select
```

PART 2: PERFORMING PASSIVE ATTACK.

TASK 1:

PERFORM NETWORK SCANNING ATTACK FROM THE ATTACKER MACHINE TO THE SERVER VM.

PERFORM TCP CONNECT SCAN

```
attacker@attacker:~$ sudo nmap -sT 10.0.2.6
Starting Nmap 7.93 ( https://nmap.org ) at 2023-02-07 13:05 +03
Nmap scan report for 10.0.2.6
Host is up (0.0024s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT     STATE SERVICE
21/tcp   open  ftp
80/tcp   open  http
443/tcp  open  https
3306/tcp open  mysql
MAC Address: 08:00:27:2E:EE:83 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.58 seconds
```

-sT

TCP Connect scan uses the concept of a full three-way handshake to discover whether a given port is open, filtered, or closed according to the response it receives. Nmap sends a TCP request packet to each and every port specified and determines the status of the port by the response it receives.

PERFORM STEALTH SCAN

```
attacker@attacker:~$ sudo nmap -sS 10.0.2.6
Starting Nmap 7.93 ( https://nmap.org ) at 2023-02-07 13:06 +03
Nmap scan report for 10.0.2.6
Host is up (0.00086s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
80/tcp   open  http
443/tcp  open  https
3306/tcp open  mysql
MAC Address: 08:00:27:2E:EE:83 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.56 seconds
```

-sS

SYN scans are often called “Half-open” or “Stealth” scans. SYN scan works the same way as TCP Connect scan with closed and filtered ports i.e receives a RST packet for closed port and no response for filtered ports. The only difference is in the way they handle the open ports. SYN scan sends a response packet to the server with its RESET FLAG set(but not ACK which is usually the default in the actual three-way handshake) after receiving SYN/ACK from the target server. This is to avoid the server from continuously making requests to establish a connection and thereby reduce the scan time.

PERFORM A SCAN THAT ENABLES OS DETECTION, VERSION DETECTION, SCRIPT SCANNING, AND TRACEROUTE.

```
attacker@attacker:~$ sudo nmap -O -sV -sC --traceroute 10.0.2.6
Starting Nmap 7.93 ( https://nmap.org ) at 2023-02-07 13:15 +03
Nmap scan report for 10.0.2.6
Host is up (0.0014s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE  VERSION
21/tcp   open  ftp      ProFTPD
80/tcp   open  http     Apache httpd 2.4.54 ((Unix) OpenSSL/1.1.1s PHP/8.2.0 mod
_perl/2.0.12 Perl/v5.34.1)
|_http-server-header: Apache/2.4.54 (Unix) OpenSSL/1.1.1s PHP/8.2.0 mod_perl/2.0
.12 Perl/v5.34.1
| http-title: Welcome to XAMPP
|_Requested resource was http://10.0.2.6/dashboard/
443/tcp  open  ssl/http Apache httpd 2.4.54 ((Unix) OpenSSL/1.1.1s PHP/8.2.0 mod
_perl/2.0.12 Perl/v5.34.1)
| http-title: Welcome to XAMPP
|_Requested resource was https://10.0.2.6/dashboard/
| ssl-cert: Subject: commonName=localhost/organizationName=Apache Friends/stateO
rProvinceName=Berlin/countryName=DE
| Not valid before: 2004-10-01T09:10:30
|_Not valid after:  2010-09-30T09:10:30
|_http-server-header: Apache/2.4.54 (Unix) OpenSSL/1.1.1s PHP/8.2.0 mod_perl/2.0
.12 Perl/v5.34.1
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_  http/1.1
3306/tcp open  mysql    MariaDB (unauthorized)
MAC Address: 08:00:27:2E:EE:83 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   1.35 ms 10.0.2.6

OS and Service detection performed. Please report any incorrect results at https
://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.18 seconds
```

we used option -A for enabling the OS detection, script scanning, version detection and traceroute with the server IP address.

PART 3: WIRESHARK

INSTALL WIRESHARK TOOL ON THE SERVER VM AND USE IT TO CAPTURE:

- **THE TCP CONNECT SCAN.**

1. ADD THE TWO SCREENSHOTS OF THE WIRESHARK-CAPTURED TRAFFIC TO YOUR SUBMITTED REPORT.

tcp.port == 21

No.	Source	Destination	Protocol	Length	Info
85	10.0.2.7	10.0.2.6	TCP	74	50626 → 21 [SYN] Seq=0 Win=64240 Len=0 MSS=
91	10.0.2.6	10.0.2.7	TCP	74	21 → 50626 [SYN, ACK] Seq=0 Ack=1 Win=65160
109	10.0.2.7	10.0.2.6	TCP	66	50626 → 21 [ACK] Seq=1 Ack=1 Win=64256 Len=
142	10.0.2.7	10.0.2.6	TCP	66	50626 → 21 [RST, ACK] Seq=1 Ack=1 Win=64256

tcp.port == 80

No.	Source	Destination	Protocol	Length	Info
65	10.0.2.7	10.0.2.6	TCP	74	44418 → 80 [SYN] Seq=0 Win=64240 Le
67	10.0.2.6	10.0.2.7	TCP	74	80 → 44418 [SYN, ACK] Seq=0 Ack=1 W
72	10.0.2.7	10.0.2.6	TCP	66	44418 → 80 [ACK] Seq=1 Ack=1 Win=64
73	10.0.2.7	10.0.2.6	TCP	66	44418 → 80 [RST, ACK] Seq=1 Ack=1 W

tcp.port == 443

No.	Source	Destination	Protocol	Length	Info
98	10.0.2.7	10.0.2.6	TCP	74	37154 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=
106	10.0.2.6	10.0.2.7	TCP	74	443 → 37154 [SYN, ACK] Seq=0 Ack=1 Win=65160
143	10.0.2.7	10.0.2.6	TCP	66	37154 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=
144	10.0.2.7	10.0.2.6	TCP	66	37154 → 443 [RST, ACK] Seq=1 Ack=1 Win=64256

tcp.port == 3306

No.	Source	Destination	Protocol	Length	Info
71	10.0.2.7	10.0.2.6	TCP	74	54378 → 3306 [SYN] Seq=0 Win=64240 Len=0 MSS=1
74	10.0.2.6	10.0.2.7	TCP	74	3306 → 54378 [SYN, ACK] Seq=0 Ack=1 Win=65160
77	10.0.2.7	10.0.2.6	TCP	66	54378 → 3306 [ACK] Seq=1 Ack=1 Win=64256 Len=0
108	10.0.2.7	10.0.2.6	TCP	66	54378 → 3306 [RST, ACK] Seq=1 Ack=1 Win=64256

2. BRIEFLY EXPLAIN EACH OF THEM.
   - Source sent SYN packet to the destination
   - Destination sent SYN, ACK to source
   - Source sent ACK packet to the destination
   - Source again sent RST, ACK to destination

- **THE STEALTH SCAN.**

1. ADD THE TWO SCREENSHOTS OF THE WIRESHARK-CAPTURED TRAFFIC TO YOUR SUBMITTED REPORT.

tcp.port == 21

No.	Source	Destination	Protocol	Length	Info
90	10.0.2.7	10.0.2.6	TCP	60	48313 → 21 [SYN] Seq=0 Win=1024 Len=0
96	10.0.2.6	10.0.2.7	TCP	58	21 → 48313 [SYN, ACK] Seq=0 Ack=1 Wi
101	10.0.2.7	10.0.2.6	TCP	60	48313 → 21 [RST] Seq=1 Win=0 Len=0

tcp.port == 80

No.	Source	Destination	Protocol	Length	Info
89	10.0.2.7	10.0.2.6	TCP	60	48313 → 80 [SYN] Seq=0 Win=1024 Len=
95	10.0.2.6	10.0.2.7	TCP	58	80 → 48313 [SYN, ACK] Seq=0 Ack=1 Wi
100	10.0.2.7	10.0.2.6	TCP	60	48313 → 80 [RST] Seq=1 Win=0 Len=0

tcp.port == 443

No.	Source	Destination	Protocol	Length	Info
79	10.0.2.7	10.0.2.6	TCP	60	48313 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
85	10.0.2.6	10.0.2.7	TCP	58	443 → 48313 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=146
93	10.0.2.7	10.0.2.6	TCP	60	48313 → 443 [RST] Seq=1 Win=0 Len=0

tcp.port == 3306

No.	Source	Destination	Protocol	Length	Info
48	10.0.2.7	10.0.2.6	TCP	60	48313 → 3306 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
51	10.0.2.6	10.0.2.7	TCP	58	3306 → 48313 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
58	10.0.2.7	10.0.2.6	TCP	60	48313 → 3306 [RST] Seq=1 Win=0 Len=0

2. BRIEFLY EXPLAIN EACH OF THEM.

- SOURCE SENT SYN PACKETS TO THE DESTINATION
- DESTINATION SENT SYN, ACK PACKETS TO THE SOURCE
- SOURCE SENT RST PACKETS TO THE DESTINATION

PART 4: FIREWALL

TASK 1

AFTER YOU SUCCESSFULLY COMPLETE PART 2 AND 3, WRITE THE FOLLOWING IPTABLES ON THE SERVER TO BLOCK THE FOLLOWING TRAFFIC TYPES ORIGINATED FROM THE ATTACKER TO THE SERVER:

1. HTTP CONNECTION REQUEST FROM THE ATTACKER TO THE SERVER.
2. SSH CONNECTION REQUEST FROM THE ATTACKER TO THE SERVER.
3. FTP AND TELNET REQUESTS (USE SINGLE RULE TO BLOCK THESE MULTIPLE PORTS).

1. SCREENSHOT THE IPTABLES BLOCK COMMANDS WITH A BRIEF EXPLANATION FOR EACH COMMAND.

```
server@server:~$ sudo iptables -A INPUT -s 10.0.2.7 -p tcp --dport 80 -j DROP
server@server:~$ sudo iptables -A INPUT -s 10.0.2.7 -p tcp --dport 22 -j DROP
server@server:~$ sudo iptables -A INPUT -s 10.0.2.7 -p tcp --match multiport --dport 20,21,23 -j DROP
```

TASK 2

CONFIGURE THE IPTABLES TO LOG DROPPED PACKETS (ENABLE LOGGING IN IPTABLES) AND THEN SHOW THE LOG MESSAGES.

2.SCREENSHOT THE IPTABLES LOG ENABLING COMMAND WITH A BRIEF EXPLANATION

```
server@server:~$ sudo iptables -N LOGGING
server@server:~$ sudo iptables -A INPUT -j LOGGING
server@server:~$ sudo iptables -A LOGGING -m limit --limit 2/min -j LOG --log-prefix "IPTa
bles-Dropped: " --log-level 4
server@server:~$ sudo iptables -A LOGGING -j DROP
```

add a new rule that logs all incoming traffic.
define the level of LOG generated by iptables use --log-level followed by the level number.
add prefixe

3.SCREENSHOT OF THE LOG MESSAGES

```
Feb 11 14:36:04 server kernel: [12674.636943] ipt-dn: IN=enp0s3 OUT= MAC=08:00:27:2e:e
:83:08:00:27:be:a9:a6:08:00 SRC=10.0.2.7 DST=10.0.2.6 LEN=576 TOS=0x00 PREC=0x00 TTL=2
5 ID=161 PROTO=UDP SPT=23 DPT=68 LEN=556
```

CONFIGURE THE IPTABLES TO LOG DROPPED PACKETS (ENABLE LOGGING IN IPTABLES) AND THEN SHOW THE LOG MESSAGES.

1. SCREENSHOT THE CONFIGURED ALERT SNORT COMMAND FROM THE SERVER VM AND BRIEFLY EXPLAIN IT.

-install snort

```
server@server:~$ sudo apt install snort
[sudo] password for server:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libdaq2 libdumbnet1 libluajit-5.1-2 libluajit-5.1-common libnetfilter-queue1 oinkmaster
  snort-common snort-common-libraries snort-rules-default
Suggested packages:
  snort-doc
The following NEW packages will be installed:
  libdaq2 libdumbnet1 libluajit-5.1-2 libluajit-5.1-common libnetfilter-queue1 oinkmaster snort
  snort-common snort-common-libraries snort-rules-default
0 upgraded, 10 newly installed, 0 to remove and 170 not upgraded.
Need to get 2,391 kB of archives.
After this operation, 10.7 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

-open the snort configuration file in gedit text editor:

```
server@server:~$ sudo gedit /etc/snort/snort.conf
```

-change the IP address part to match the server VM IP

```
#
# Note to Debian users: this value is overriden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 10.0.2.6/24

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
```

-run "sudo snort -T -i eth0 -c /etc/snort/snort.conf"

```
Snort successfully validated the configuration!
Snort exiting
```

- apply the following rule in snort local rule file.

ALERT TCP 10.0.2.7 ANY -> 10.0.2.6 22 (MSG: "NMAP TCP SCAN";SID:10000005; REV:2;)

```
Open    *local.rules    Save
/etc/snort/rules
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ---------------
# LOCAL RULES
# ---------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert tcp 10.0.2.7 any -> 10.0.2.6 22 (msg: "NMAP TCP Scan";sid:10000005; rev:2; )
```

2.SCREENSHOT THE TCP CONNECT SCAN ON PORT 22 COMMAND FROM THE ATTACKER VM.

```
attacker@attacker:~$ sudo nmap -sT -p22 10.0.2.6
Starting Nmap 7.93 ( https://nmap.org ) at 2023-02-11 20:17 +03
Nmap scan report for 10.0.2.6
Host is up (0.0011s latency).

PORT   STATE    SERVICE
22/tcp filtered ssh
MAC Address: 08:00:27:2E:EE:83 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.37 seconds
```

3.SCREENSHOT THE SNORT RESPONSE AT THE SERVER VM.

SNORT IS CAPTURING ALL INCOMING TRAFFIC HERE WE CAN OBSERVE THAT IT IS GENERATING AN ALERT FOR NMAP TCP SCAN.

```
server@server: ~
server@server:~$ sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i enp0s3
02/11-12:17:42.520864  [**] [1:10000005:2] NMAP TCP Scan [**] [Priority: 0] {TCP} 10.0.2.7:56376 -> 10.0.2.6:22
02/11-12:17:42.622413  [**] [1:10000005:2] NMAP TCP Scan [**] [Priority: 0] {TCP} 10.0.2.7:56380 -> 10.0.2.6:22
```

CONCLUSION

Finally, let us review what we did. We deployed and configured two virtual machines, the first acting as the server and the second as the attacker.
We initiated a passive assault from the attacker computer by configuring tcp and sleath scanning to learn about the server's open ports, operating system, version detection, script scanning, and traceroute.
We utilized the Wireshark tool from the server machine to capture the traffic, analyse it, and ensure that no problems occurred throughout the exchange.
We discovered that the server was not secured from such scanning efforts, therefore we started implementing firewalls by blocking the attacker machine ports of "HTTP, SSH, FTP, AND TELNET ". Then activate logging so that any packet drops are recorded. We employed snort to gather TCP scans of SSH connections on port 22 and also applied roles to activate alarms if intrusion detection systems were discovered.