CFC130124
Penetration Testing Project - Vulners
Shaun Sng (S17)
Jun 2024

Table of Contents

1. Introduction

Summary This project entails drafting a script to automate scanning a network/host for vulnerabilities and should achieve the following functions:

a. Incorporate and validate user input e.g. choice of ‘Basic’ or ‘Full Scan’, specifying output directory and custom password list.
b. Check for weak passwords in login services - SSH, RDP, FTP and Telnet.
c. Basic scan checks for open ports/services and weak passwords. Full scan additionally runs the Nmap vulners script and Searchsploit.
d. Allow user to search results.
e. Save results in the Zip file.

Aim - Project requires students to demonstrate competence in tools for network scanning/enumeration, while further developing familiarity in scripting.

2. Methodologies

2.1 Initialisation and Processing User Input

Prior to the ‘main’ script itself, several ‘fixed’ variables are declared to facilitate subsequent scripting - e.g. for password/user lists, output directory, text formatting code and strings/messages.

```
# Various files required for this script saved as FIXED variables.
# If user does not specfiy a passwordlist, script will look for below options in their machine.
# >> User may wish to change list to suit your context and time/rigour for the search.

# These lists are possible options chosen as relatively short lists (100+) with a  fair chance of existing on user system.
# However, they still take up some time for weakpassword check, in particular for telnet.
# As such they will be kept in 'cold storage' - for user's consideration.

# PASS_DEF='/usr/share/seclists/Passwords/darkweb2017-top100.txt'
# USERS_DEF='/usr/share/seclists/Usernames/top-usernames-shortlist.txt'

 # For project submission and to facilitate quick script processing, will retain these 'fake lists' - intended to return a 'Not found' result.
 PASS_DEF='/usr/share/seclists/Passwords/TEMPFAKE'
 USERS_DEF='/usr/share/seclists/Usernames/TEMPFAKE'

# If above lists can't be found, script wll generate user and password files/lists using below arrays.
# >> Arrays have been kept very short for testing/efficiency. Feel free to edit/add if needed.

 PASSW=("user" "msfadmin" "123456" "password" "qwerty")
 USERS=("user" "msfadmin" "root")

# Default output directory if user does not specify.
# REMINDER -- change to $pwd in final submission.

 # OUTPUT_DIR='/home/kali/pentest/proj/scan_1306'
 OUTPUT_DIR=$(pwd)
```

```
# Text formatting options as variables for easier use.
GREEN='\e[32m'
CYAN='\e[36m'
RED='\e[31m'
YELLOW='\e[33m'
CLEAR='\e[0m'
BOLD='\e[1m'
LINED='\e[4m'
BLUE='\e[34m'
MAGENTA='\e[35m'
```

The first stage of the script prompts the user for required input such as an IP address, and selection of basic or full scan.

```
## Main body. Title/banner and welcome message.
echo -e "[-] Vulnerability scan for CFC-130124 PT project. By optimus_s17 2024. "
echo -e "[⚙] Welcome, $HUMAN. Script is initialising..."
echo

# Ask user for IP address. Call check_ip function to validate.
echo -e "[?] Please enter an IP address for scanning (e.g. ${GREEN}192.168.130.24)${CLEAR}: "
read ip_addr
echo

# If invalid input, this script will exit. At this initial stage of the script, it is deemed fair for poor input to force a break/exit.
# As the script progresses, we avoid doing an exit/break due to invalid input and use defaults where possible.

if check_ip $ip_addr; then
    :
    echo -e "[+] $HUMAN has entered a valid IP ${CYAN}$ip_addr${CLEAR}. Script will scan this address. "
else
    echo -e "[x] $HUMAN has entered an invalid IP ${RED}$ip_addr${CLEAR}. This script will exit."
    exit
fi

echo
```

```
┌──(kali㉿kali)-[~/pentest/proj/scan_final]
└─$ ./vulners_shaun_final.sh
[-] Vulnerability scan for CFC-130124 PT project. By: Shaun Sng (S17). Trainer: Kar Wei
[⚙] Welcome, kali. Script is initialising...

[?] Please enter an IP address for scanning (e.g. 192.168.130.24):
192.168.133.132

[+] kali has entered a valid IP 192.168.133.132. Script will scan this address.

[?] Please input the directory you wish to save scan results. (e.g. /home/kali/pentest/proj )
n

[x] Directory could not be found, or kali has entered invalid input. Results will be saved to the current working directory.

[?] Please enter '1' for 'Basic' scan or '2' for 'Full' scan.
1

Preparing to run 1 scan per user selection.
[?] Does kali wish to specify your own password list in this scan? Enter 'Y' or 'Yes' if so. All other input will be taken as 'No' [Y/N]
```

Inputs are validated. If not done for IP address, subsequent commands would fail/break the script. For other options such as type of scan, invalid inputs would just default the choice to ‘basic scan’ instead of stopping the script.

```
53  ## Functions
54
55  # Check if IP address input by user is 'valid'.  i.e. if  it matches ipv4 format. Namely, four sets of 3 digits between 0 - 255, separated with a period.
56
57  check_ip() {
58      local ip=$1
59      local stat=1
60      ip_grep=$(echo $ip | grep -Eo '^(([1-9]?[0-9]|1[0-9][0-9]|2([0-4][0-9]|5[0-5]))\.){3}([1-9]?[0-9]|1[0-9][0-9]|2([0-4][0-9]|5[0-5]))$')
61
62      if [[ "$ip_grep" == "$ip" ]]; then
63          stat=0
64      fi
65      return $stat
66  }
67
68  # Check/parse user input for choice of basic or full scan. Input validation to 'fix' results as only two options - basic or full in the variable $scan_type.
69
70  check_scan() {
71      local type=$(echo $1 | tr '[:upper:]' '[:lower:]')
72      if [[ $type == 'basic' || $type == '1' ]]; then
73          echo "Preparing to run $type scan per user selection."
74          scan_type='basic'
75      elif [[ $type == 'full' || $type == '2' ]]; then
76          scan_type='full'
77      else
78          echo "User has input an invalid scan type. Defaulting to basic scan."
79          scan_type='basic'
80      fi
81  }
```

2.2 TCP and UDP Port Scans

The foundation of this script is a port scan - using Nmap for TCP and Masscan for UDP. Open UDP ports detected by Masscan are processed once more through Nmap to obtain service versions.

For ‘Basic scan’ only the default 1000 TCP ports will be scanned. For ‘Full scan’ all TCP ports will be scanned. This is partly to speed up the basic scan, while further differentiating between the two levels of scan.

```
204 ## Functions to do the core scanning work.
205 # Port scans with masscan and Nmap
206
207 port_scan(){
208     # UDP scan with masscan. Speed/rate bumped up to 10k, and scan all ports.
209
210     echo -e "${BOLD}$UDP1${CLEAR} Starting..."
211     sudo masscan $ip_addr -pU:1-65535 --rate=10000 -oL $out_dir/1_massudp
212
213     # Process the output from masscan to derive the list of open UDP ports.
214     # For Nmap to scan these specific UDP ports later.
215
216     udp_ports=$(cat $out_dir/1_massudp | awk '{print$3}'| grep '[0-9]' | paste -s -d, /dev/stdin)
217     echo
218
219     # TCP scan with nmap. We incorporate one optional $1 argument
220     # Toggles between scanning default 1k ports for basic scan, and all ports for full scan.
221
222     echo -e "${BOLD}$TCP${CLEAR} Scanning $tcp_ports ports.$TAKETIME"
223     nmap -oA $out_dir/2_nmap_tcp -sV -T5 $ip_addr $1
224     echo
225     echo -e "${BOLD}$UDP2${CLEAR} Starting..."
226     sudo nmap -oA $out_dir/3_nmap_udp -sU -sV $ip_addr -p $udp_ports
227
228 }
```

```
[1] UDP port scan - all ports. Starting ...
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:
Starting masscan 1.3.2 (http://bit.ly/14GZzcT) at 2024-06-13 09:38:55 GMT
Initiating SYN Stealth Scan
Scanning 1 hosts [65535 ports/host]
                                                                    d=0
[2] TCP scan with service version. Scanning 65,535 ports.This may take 2 - 3 minutes ...
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-13 05:40 EDT
Stats: 0:00:08 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 40.00% done; ETC: 05:40 (0:00:09 remaining)
Stats: 0:00:13 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 80.00% done; ETC: 05:40 (0:00:03 remaining)
Nmap scan report for msf (192.168.133.132)
Host is up (0.0046s latency).
Not shown: 65505 closed tcp ports (conn-refused)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
53/tcp    open  domain      ISC BIND 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec        netkit-rsh rexecd
```

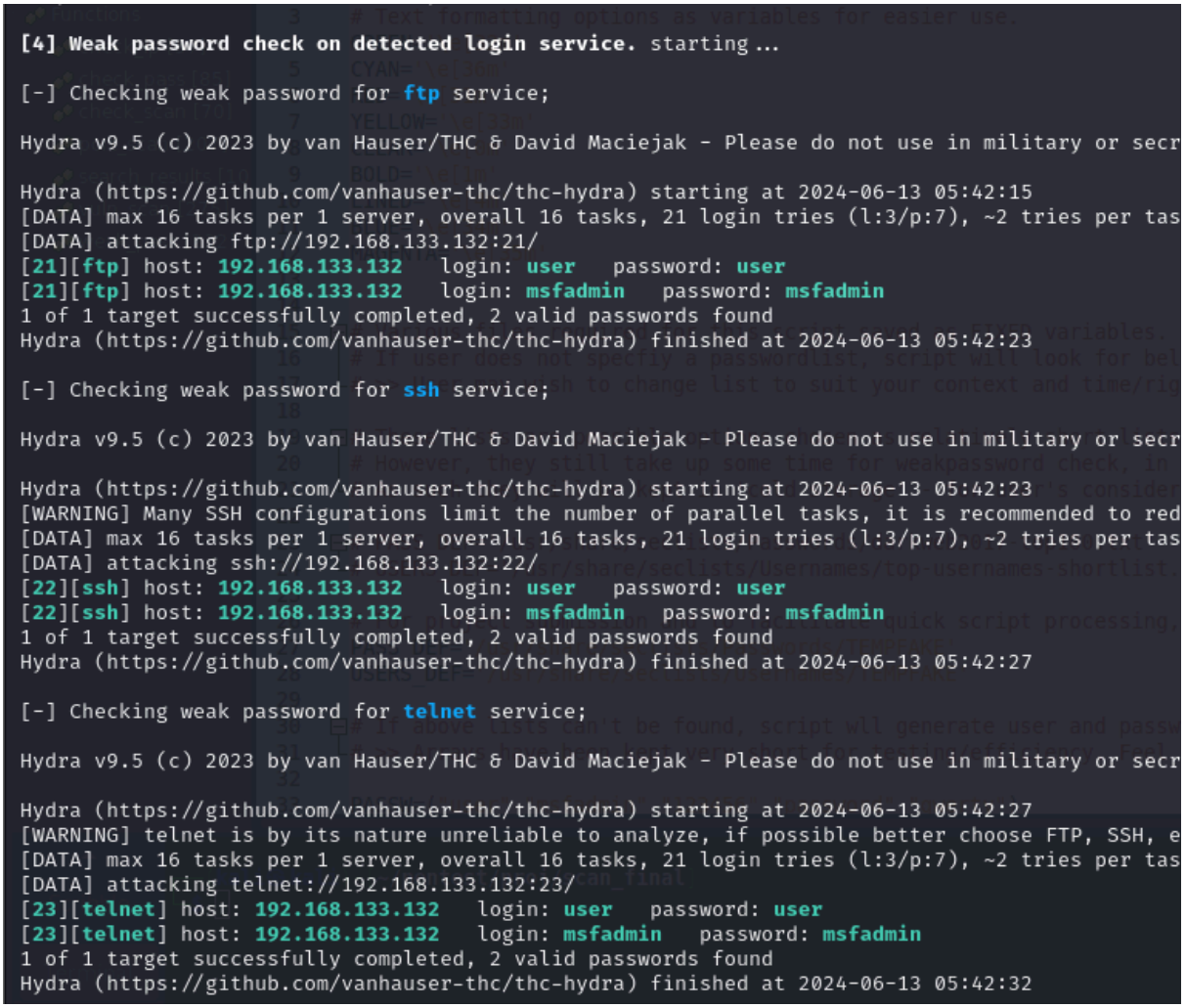
```
[1] UDP port scan - all ports. Starting ...
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:
Starting masscan 1.3.2 (http://bit.ly/14GZzcT) at 2024-06-13 09:38:55 GMT
Initiating SYN Stealth Scan
Scanning 1 hosts [65535 ports/host]
                                                                    d=0
[2] TCP scan with service version. Scanning 65,535 ports.This may take 2 - 3 minutes ...
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-13 05:40 EDT
Stats: 0:00:08 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 40.00% done; ETC: 05:40 (0:00:09 remaining)
Stats: 0:00:13 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 80.00% done; ETC: 05:40 (0:00:03 remaining)
```

2.3 Credential Check - Weak Password check

The output of both TCP and UDP Nmap scans are checked to determine if any login services are running and saved to the variable *'check_services'*.

If none are detected (by conditional [-z "$check services"]), the user is informed and the weak password check is skipped. For each detected service, *hydra* is run with password and user lists to check for any weak passwords.

```
230  # Function to check for weak passwords.
231
232  weak_pass(){
233      userlist=""
234
235      # Check if default password list exist/can be found. Generate own list if not.
236      if test -f $USERS_DEF; then
237          userlist=$USERS_DEF
238      else
239          :> $out_dir/users.lst
240          for user in ${USERS[@]};
241          do
242              echo $user >> $out_dir/users.lst
243          done
244          userlist=$out_dir/users.lst
245      fi
246
247      # Process earlier nmap results to check if open ports detected on the four login services - ftp, ssh, rdp, telnet.
248
249      check_services=$(cat $out_dir/2_nmap_tcp.nmap $out_dir/3_nmap_udp.nmap | grep open | grep -Eo "ftp|ssh|rdp|telnet"
250      echo
251
252      # For any login service detected from earlier scan, run weak password check for each. If none, inform user and skip
253
254      echo -e "${BOLD}$WEAK${CLEAR} starting... "
255      echo
256      # This conditional check dependent whether check_services variable is empty.
257      if [ -z "$check_services" ]; then
258          echo "[x] No running login services (FTP, SSH, Telnet or RDP) detected. Skipping weak password checks."
259          echo
260      else
261          # Core/key code - for each of the detected login services saved in variable $check_services, run hydra with the
262          for item in $check_services;
263          do
264              echo -e "[-] Checking weak password for ${CYAN}${BOLD}$item${CLEAR} service;"
265              echo
266              # Some brief performance optimisation was attempted by reducing -w to 10, instead of default.
267              hydra -L $userlist -P $pass_list -e ns -w 10 -o $out_dir/4_hydra.out -I $ip_addr $item
268              echo
269          done
270      fi
271  }
```


[4] Weak password check on detected login service. starting ...
[-] Checking weak password for ftp service;
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secr
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-13 05:42:15
[DATA] max 16 tasks per 1 server, overall 16 tasks, 21 login tries (l:3/p:7), ~2 tries per tas
[DATA] attacking ftp://192.168.133.132:21/
[21][ftp] host: 192.168.133.132 login: user password: user
[21][ftp] host: 192.168.133.132 login: msfadmin password: msfadmin
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-13 05:42:23
[-] Checking weak password for ssh service;
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secr
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-13 05:42:23
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to red
[DATA] max 16 tasks per 1 server, overall 16 tasks, 21 login tries (l:3/p:7), ~2 tries per tas
[DATA] attacking ssh://192.168.133.132:22/
[22][ssh] host: 192.168.133.132 login: user password: user
[22][ssh] host: 192.168.133.132 login: msfadmin password: msfadmin
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-13 05:42:27
[-] Checking weak password for telnet service;
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secr
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-13 05:42:27
[WARNING] telnet is by its nature unreliable to analyze, if possible better choose FTP, SSH, e
[DATA] max 16 tasks per 1 server, overall 16 tasks, 21 login tries (l:3/p:7), ~2 tries per tas
[DATA] attacking telnet://192.168.133.132:23/
[23][telnet] host: 192.168.133.132 login: user password: user
[23][telnet] host: 192.168.133.132 login: msfadmin password: msfadmin
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-13 05:42:32

2.4 Nmap NSE Vulners and Searchsploit

The ‘Full Scan’ option, includes running Nmap vulners script and Searchsploit to check for potential vulnerabilities.

For Searchsploit, we run using the .XML output files from earlier Nmap scans. More ‘manual’ manipulation was explored instead of using XML mode but was not determined to provide significant marginal value.

```
# Function to conduct the Nmap NSE scan and searchsploit search for Full scan option.
vuln_scan(){

    echo -e "${BOLD}$NSE Starting.${CLEAR} $TAKETIME"

    # Key Nmap command - run the vulners script and save to basic output format.
    # >> T5 used to maximise performance. User may wish to adjust for your context if needed.

    nmap -oN $out_dir/5_nmap_vulners.out -T5 -sV --script vulners $ip_addr -p-

    # Key searchsploit commands - Use in built option to take in nmap xml files as input.
    # Hence we use two commands as earlier nmap searches were done separately for TCP and UDP - producing two separate XML files.
    # Combine both files in one '6_sploit_all.out' file for later search.

    ## Using metasploitable as test machine retrieves a large number of results. Noisy/difficult to know where to pay attention.

    searchsploit --disable-colour -x --nmap $out_dir/2_nmap_tcp.xml > $out_dir/6_sploit_tcp.lst
    searchsploit --disable-colour -x --nmap $out_dir/3_nmap_udp.xml > $out_dir/6_sploit_udp.lst
    cat $out_dir/6_sploit_tcp.lst $out_dir/6_sploit_udp.lst > $out_dir/6_sploit_all.out
}
```

```
[5] Nmmap vulnerabiity scan - vulners script. Starting. This may take 2 - 3 minutes...
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-13 05:42 EDT
Nmap scan report for msf (192.168.133.132)
Host is up (0.0095s latency).
Not shown: 65505 closed tcp ports (conn-refused)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
| vulners:
|   cpe:/a:vsftpd:vsftpd:2.3.4:
|       PRION:CVE-2011-2523     10.0    https://vulners.com/prion/PRION:CVE-2011-2523
|       EDB-ID:49757    9.8     https://vulners.com/exploitdb/EDB-ID:49757      *EXPLOIT*
|_      1337DAY-ID-36095        9.8     https://vulners.com/zdt/1337DAY-ID-36095        *EXPLOIT*
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| vulners:
|   cpe:/a:openbsd:openssh:4.7p1:
|       SSV:78173       7.8     https://vulners.com/seebug/SSV:78173    *EXPLOIT*
|       SSV:69983       7.8     https://vulners.com/seebug/SSV:69983    *EXPLOIT*
|       PACKETSTORM:98796       7.8     https://vulners.com/packetstorm/PACKETSTORM:98796       *EXPLOIT*
|       PACKETSTORM:94556       7.8     https://vulners.com/packetstorm/PACKETSTORM:94556       *EXPLOIT*
|       PACKETSTORM:101052      7.8     https://vulners.com/packetstorm/PACKETSTORM:101052      *EXPLOIT*
|       EXPLOITPACK:71D51B69AA2D3A74753D7A921EE79985    7.8     https://vulners.com/exploitpack/EXPLOITPACK:71D51B69AA2D3A74753D7A921EE79985    *EXPLOIT*
|       EXPLOITPACK:67F6569F63A082199721C069C852BBD7    7.8     https://vulners.com/exploitpack/EXPLOITPACK:67F6569F63A082199721C069C852BBD7    *EXPLOIT*
|       EDB-ID:24450    7.8     https://vulners.com/exploitdb/EDB-ID:24450      *EXPLOIT*
|       EDB-ID:15215    7.8     https://vulners.com/exploitdb/EDB-ID:15215      *EXPLOIT*
|       SECURITYVULNS:VULN:8166 7.5     https://vulners.com/securityvulns/SECURITYVULNS:VULN:8166
```

```
[i] SearchSploit's XML mode (without verbose enabled).   To enable: searchsploit -v --xml...
[i] Reading: '/home/kali/pentest/proj/scan_final/2_nmap_tcp.xml'

[-] Skipping term: ftp   (Term is too general. Please re-search manually: /usr/bin/searchsploit -t --disable-colour ftp)

[i] /usr/bin/searchsploit -t --disable-colour vsftpd
[i] /usr/bin/searchsploit -t --disable-colour vsftpd 2.3.4
[-] Skipping term: ssh   (Term is too general. Please re-search manually: /usr/bin/searchsploit -t --disable-colour ssh)

[i] /usr/bin/searchsploit -t --disable-colour openssh
[i] /usr/bin/searchsploit -t --disable-colour openssh 4.7p1 debian 8ubuntu1
[i] /usr/bin/searchsploit -t --disable-colour telnet
[i] /usr/bin/searchsploit -t --disable-colour linux telnetd
[i] /usr/bin/searchsploit -t --disable-colour smtp
[i] /usr/bin/searchsploit -t --disable-colour postfix smtpd
[i] /usr/bin/searchsploit -t --disable-colour domain
[-] Skipping output: domain   (Too many results, 100+. You'll need to force a search: /usr/bin/searchsploit -t --disable-colour domain)

[i] /usr/bin/searchsploit -t --disable-colour isc bind
[i] /usr/bin/searchsploit -t --disable-colour isc bind 9.4.2
[-] Skipping term: http   (Term is too general. Please re-search manually: /usr/bin/searchsploit -t --disable-colour http)
```

2.5 Reporting Results

After the ‘main’ scan is complete, further processing of earlier results is done to simplify and highlight key pieces of information - such as the number of open ports (saved in variables ‘open_tcp’ and ‘open_udp’).

```
326  # Reporting Stage. Prepare to summarise findings for user.
327  echo
328  echo
329  echo -e "${BOLD}${GREEN}[-] Scan complete.${CLEAR} ${BOLD}Organising results...${CLEAR}"
330  sleep 2
331  echo
332
333  # Quick processing of earlier results to count the number of open ports.
334  open_tcp=$(cat $out_dir/2_nmap_tcp.nmap | grep open | wc -l)
335  open_udp=$(cat $out_dir/3_nmap_udp.nmap | grep open | wc -l)
336
337
338  # At each stage, we prepare the user what info they will be seeing.
339  echo "[-] Results for port scans from stages:"
340  echo --$UDP1
341  echo --$TCP
342  echo --$UDP2
343  echo
344
345  # Output the key messages of how many open ports were detected.
346  echo -e "[-] ${RED}${BOLD}$open_tcp${CLEAR} open ports on TCP were detected. Service and versions detected below:"
347  echo
348
349
350  # Extract/recap the main table of open ports & services from earlier nmap scans.
351  cat $out_dir/2_nmap_tcp.nmap |  grep --color=never ^PORT && cat $out_dir/2_nmap_tcp.nmap | grep --color=never open
352  echo
353  echo -e "[-] ${RED}${BOLD}$open_udp${CLEAR} open ports on UDP were detected. Service and versions detected below:"
354  echo
355  cat $out_dir/3_nmap_udp.nmap |  grep --color=never ^PORT && cat $out_dir/3_nmap_udp.nmap | grep --color=never open
356  echo
357
358  echo "[!] User should consider closing ports for any unrequired service."
359  echo
360  sleep 5
```

```
[-] Scan complete. Organising results...

[-] Results for port scans from stages:
--[1] UDP port scan - all ports.
--[2] TCP scan with service version.
--[3] UDP scan with service version - open ports only.

[-] 30 open ports on TCP were detected. Service and versions detected below:

PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
53/tcp    open  domain      ISC BIND 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec        netkit-rsh rexecd
513/tcp   open  login       OpenBSD or Solaris rlogind
514/tcp   open  tcpwrapped
1099/tcp  open  java-rmi    GNU Classpath grmiregistry
1524/tcp  open  bindshell   Metasploitable root shell
2049/tcp  open  nfs         2-4 (RPC #100003)
2121/tcp  open  ftp         ProFTPD 1.3.1
3306/tcp  open  mysql       MySQL (blocked - too many connection errors)
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc         VNC (protocol 3.3)
6000/tcp  open  X11         (access denied)
6667/tcp  open  irc         UnrealIRCd (Admin email admin@Metasploitable.LAN)
6697/tcp  open  irc         UnrealIRCd
8009/tcp  open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb         Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
47595/tcp open  java-rmi    GNU Classpath grmiregistry
52999/tcp open  mountd      1-3 (RPC #100005)
53825/tcp open  status      1 (RPC #100024)
59445/tcp open  nlockmgr    1-4 (RPC #100021)

[-] 2 open ports on UDP were detected. Service and versions detected below:

PORT    STATE SERVICE    VERSION
53/udp  open  domain     ISC BIND 9.4.2
137/udp open  netbios-ns Microsoft Windows netbios-ns (workgroup: WORKGROUP)
```

Similarly, weak password results are processed/extracted to summarise and output to user which credentials were found.

```
364   echo $WEAK stage results:
365
366   # Checking and doing and appropriate output if no password checks was done.
367
368   if [ -z "$check_services" ]; then
369       echo "[-] No weak password checks for login services were done. No results to report."
370       # continue
371   else
372       echo "[-] Weak passwords (and corresponding matching user login IDs) were detected on the following ports & services:"
373       echo
374       # Extract and summarise just the key data from earlier hydra scan -i.e. which services/ports, user login and passwords were detected.
375       grep login $out_dir/4_hydra.out | sort | uniq | awk '{print$1,$4,$5,$6,$7}'
376       echo
377       echo "[!] User should consider strengthening passwords or removing these login accounts."
378
379   fi
```

```
[!] User should consider closing ports for any unrequired service.

[4] Weak password check on detected login service. stage results:
[-] Weak passwords (and corresponding matching user login IDs) were detected on the following ports & services:

[21][ftp] login: msfadmin password: msfadmin
[21][ftp] login: user password: user
[22][ssh] login: msfadmin password: msfadmin
[22][ssh] login: user password: user
[23][telnet] login: msfadmin password: msfadmin
[23][telnet] login: user password: user

[!] User should consider strengthening passwords or removing these login accounts.
```

More involved processing was required to determine how to output the results from the Nmap vulners and Searchsploit results. Using metasploitable as the test machine resulted in a long list of detected vulnerabilities and information overload.

The topline number of total vulnerabilities detected by the NMap vulners script is output to the user (*line 410: 'Total Matches'.*) The user is cautioned that this includes all reports and 'duplicates.' The affected services is extracted and output to user.

The vulners script/Exploit DB includes reports from multiple sources/vendors that may refer to the same CVE. To simplify the results, the script focuses only on the 'vanilla' CVEs detected. Summarising this figure (*line 417: 'CVEs'*) and outputting the top 10 with URL links for user's closer attention. Note that this figure is deliberately coloured red for closer attention, compared to earlier text in yellow for 'total matches'.

```
384  ## Processing results from Nmap NSE scan.
385
386  # Nmap NSE detected many results. The total/complete results from ExploitDB include many difference sources/reporters/vendors - many reports refer to the same CVE.
387  # As such you can say there is "double counting/duplicates". Still relevant to get a sense, but very noisy.
388
389  # We process the earlier nmap results to get first both the 'total vulnerabilities' and 'CVE' only. Output to text files for easier manipulation.
390  # The latter serves to avoid double counting to get a more manageable list/sense of the extent of vulnerability.
391
392  grep CVE $out_dir/5_nmap_vulners.out | awk '{print$2}' | sed 's/^[^:]*://' | sort | uniq | wc -l > $out_dir/cve_trim.lst
393  grep cve $out_dir/5_nmap_vulners.out | awk '{print$2,$3,$4}' | sort | uniq > $out_dir/cve_only.out
394
395
396  # Save numbers of total vulnerabilities and CVE only to variables for subsequent reporting to console/user.
397  num_all=$(cat $out_dir/5_nmap_vulners.out | grep -E "cve|CVE|SSV|OSV|PRION|packetstorm|securityvulns|1337DAY|github|EDB-ID|SAINT|EXPLOIT" | wc -l)
398  num_cve=$(grep cve $out_dir/5_nmap_vulners.out | awk '{print$2,$3,$4}' | sort | uniq | wc -l)
399
400  # Save /extract the total vulnerabilities and CVE results themselves subsequent reporting to console/user.
401  vul_table=$(grep ^'| vulners' -B 1 $out_dir/5_nmap_vulners.out | grep --color=never open)
402  vul_services=$(grep ^'| vulners' -B 1 $out_dir/5_nmap_vulners.out | grep --color=never open | awk '{print$3}' | sort | uniq)
403
404  grep ^'| vulners' -B 1 $out_dir/5_nmap_vulners.out | grep --color=never open | awk '{print$3}' | sort | uniq > $out_dir/6_vul_service.lst
405  num_services=$(cat $out_dir/6_vul_service.lst| wc -l)
406
407  ## Reporting Nmap NSE results to user.
408  echo $NSE stage results:
409  echo
410  echo -e "[-] Total matches: ${YELLOW}$num_all potential vulnerabilities${CLEAR} were detected. This counts ${LINED}all reports across multiple vendors and info sources${CLEAR}, and includes 'duplicate' ent
411  echo -e "[-] Affected services: Detected vulnerabilities were from the ${YELLOW}$num_services services${CLEAR} recapped in table below:"
412  echo "$vul_table"
413  echo
414  echo
415
416  # Output to the top 10 CVEs list sorted by CVSS score. This was one approach to filter/extract the most key info for user.
417  echo -e "[-] CVEs: To avoid 'double counting,' we focus on the CVEs detected. The scan detected ${RED}${BOLD}$num_cve CVEs${CLEAR}. The top 10 potentially most severe based on CVSS score are shown below:"
418  echo
419  grep cve $out_dir/5_nmap_vulners.out | awk '{print$2,$3,$4}' | sort | uniq | sort -k 2 -r -n | head -n 10
420  echo
421  echo "[>] For more details and full results, see log file $out_dir/5_nmap_vulners.nmap"
```

```
[5] Nmmap vulnerabiity scan - vulners script. stage results:

[-] Total matches: 476 potential vulnerabilities were detected. This counts all reports across multiple vendors and info sources, and includes 'duplicate' entries referring to the same CVE.
[-] Affected services: Detected vulnerabilities were from the 6 services recapped in table below:
21/tcp    open  ftp         vsftpd 2.3.4
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
53/tcp    open  domain      ISC BIND 9.4.2
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
2121/tcp  open  ftp         ProFTPD 1.3.1
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1

[-] CVEs: To avoid 'double counting,' we focus on the CVEs detected. The scan detected 177 CVEs. The top 10 potentially most severe based on CVSS score are shown below:

CVE-2008-0122 10.0 https://vulners.com/cve/CVE-2008-0122
CVE-2022-45141 9.8 https://vulners.com/cve/CVE-2022-45141
CVE-2019-12815 9.8 https://vulners.com/cve/CVE-2019-12815
CVE-2017-7679 9.8 https://vulners.com/cve/CVE-2017-7679
CVE-2017-3167 9.8 https://vulners.com/cve/CVE-2017-3167
CVE-2017-9788 9.1 https://vulners.com/cve/CVE-2017-9788
CVE-2011-4130 9.0 https://vulners.com/cve/CVE-2011-4130
CVE-2022-32744 8.8 https://vulners.com/cve/CVE-2022-32744
CVE-2022-2031 8.8 https://vulners.com/cve/CVE-2022-2031
CVE-2022-0336 8.8 https://vulners.com/cve/CVE-2022-0336

[>] For more details and full results, see log file /home/kali/pentest/proj/scan_final/5_nmap_vulners.nmap
```

Searchsploit results are comprehensive and potentially overwhelming, and would benefit from refinement/priorisation for more targeted attention.

The services detected in the earlier Nmap vulners scan was identified as an appropriate avenue to facilitate this refinement. Both Nmap and Searchploit detecting vulnerabilities reduces the chances of a false positive.

The detected services was earlier output as a text file *'vul_service.lst'* and used to filter the earlier searchsploit results (Line 435) - constituting a 'priority' list for attention.

```
425  ## Reporting Searchsploit results
426  echo $SPLOIT stage results:
427
428  # Processing searchsploit results. Saving numbers for reporting.
429  num_sploit=$(grep -v 'No Results' $out_dir/6_sploit_all.out | grep ^[A-Za-z] | wc -l)
430  num_filter=$(grep -i -f $out_dir/6_vul_service.lst $out_dir/6_sploit_all.out | wc -l)
431
432  # As searchsploit yielded a large number of noisy results. We opted triangulate and filter using the Nmap NSE scan.
433  # The detected services from Nmap NSE are used as search/filter to filter down the searchsploit results.
434
435  grep -i -f $out_dir/6_vul_service.lst $out_dir/6_sploit_all.out > $out_dir/6_sploit_priority.out
436
437  echo
438  echo -e "[-] Total matches: ${YELLOW}$num_sploit potential vulnerabilities${CLEAR} were detected in the full searchsploit results - see $out_dir/6_sploit_all.out."
439
440  echo -e "[!] Priority: Filtering with the services detected in the Nmap NSE scan obtains a higher priority list of ${RED}$num_filter potential vulnerabilities${CLEAR}:"
441  echo
442
443  # Outputting the filter sploit to screen.
444  cat $out_dir/6_sploit_priority.out
445  # grep -i -f $out_dir/6_vul_service.lst $out_dir/6_sploit_all.out
446  echo
447  echo -e "[-] Filtered/prioritised searchsploit results in $out_dir/6_sploit_priority.out."
448  echo
449  echo "[-] End of main scan."
```

```
[6] Searchsploit potential vulnerabilities. stage results:

[-] Total matches: 403 potential vulnerabilities were detected in the full searchsploit results - see /home/kali/pentest/proj/scan_final/6_sploit_all.out.
[!] Priority: Filtering with the services detected in the Nmap NSE scan obtains a higher priority list of 102 potential vulnerabilities:

vsftpd 2.0.5 - 'CWD' (Authenticated) Remote Memory Consumption                                  | linux/dos/5814.pl
vsftpd 2.0.5 - 'deny_file' Option Remote Denial of Service (1)                                  | windows/dos/31818.sh
vsftpd 2.0.5 - 'deny_file' Option Remote Denial of Service (2)                                  | windows/dos/31819.pl
vsftpd 2.3.2 - Denial of Service                                                                | linux/dos/16270.c
vsftpd 2.3.4 - Backdoor Command Execution                                                       | unix/remote/49757.py
vsftpd 2.3.4 - Backdoor Command Execution (Metasploit)                                          | unix/remote/17491.rb
vsftpd 3.0.3 - Remote Denial of Service                                                         | multiple/remote/49719.py
vsftpd 2.3.4 - Backdoor Command Execution                                                       | unix/remote/49757.py
vsftpd 2.3.4 - Backdoor Command Execution (Metasploit)                                          | unix/remote/17491.rb
Debian OpenSSH - (Authenticated) Remote SELinux Privilege Escalation                            | linux/remote/6094.txt
Dropbear / OpenSSH Server - 'MAX_UNAUTH_CLIENTS' Denial of Service                              | multiple/dos/1572.pl
FreeBSD OpenSSH 3.5p1 - Remote Command Execution                                                | freebsd/remote/17462.txt
glibc-2.2 / openssh-2.3.0p1 / glibc 2.1.9x - File Read                                          | linux/local/258.sh
Novell Netware 6.5 - OpenSSH Remote Stack Overflow                                              | novell/dos/14866.txt
OpenSSH 1.2 - '.scp' File Create/Overwrite                                                      | linux/remote/20253.sh
OpenSSH 2.3 < 7.7 - Username Enumeration                                                        | linux/remote/45233.py
OpenSSH 2.3 < 7.7 - Username Enumeration (PoC)                                                  | linux/remote/45210.py
OpenSSH 2.x/3.0.1/3.0.2 - Channel Code Off-by-One                                               | unix/remote/21314.txt
```

2.6 Search and Logging

Per project requirement, user is provided an option to search through results.

```
452  ## Ask user if they want to search within results, and call keyword search function if so.
453
454  echo -e "[?] Would $HUMAN like to search within scan results now? Enter 'Y' for Yes or 'N' for No. [Y/N]"
455  read searchnow
456  choice=$(echo $searchnow | tr '[:upper:]' '[:lower:]')
457  echo $choice
458  if [[ $choice == 'y' || $choice == 'yes' ]]; then
459      search_results
460  elif [[ $choice == 'n' || $choice == 'no' ]]; then
461      echo "[-] Acknowledged. Skipping search."
462  else
463      echo -e "[x] $HUMAN has entered an invalid option. Skipping search."
464  fi
465
466  echo
467
```

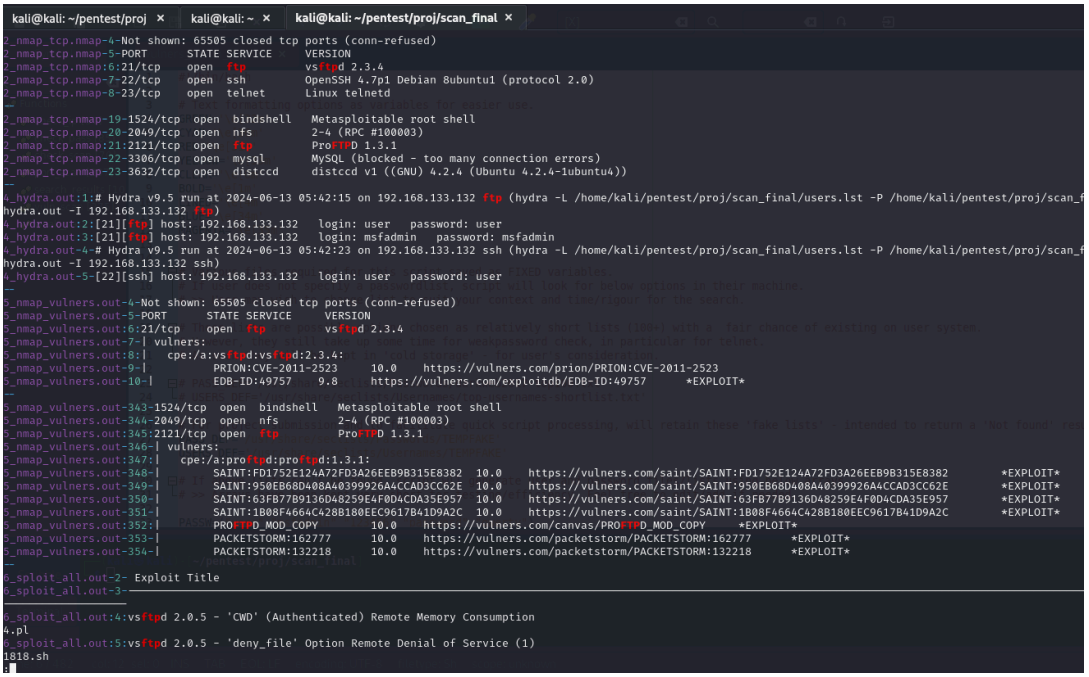
The search_results function uses *grep* for keyword match across several files output from earlier results. This comprises the TCP and UDP *.nmap output files (default from the Nmap -oA flag), and logged results from weak password checks, vulners script and searchsploit, earlier saved as files with an *"*.out"* extension to distinguish from other 'working' files.

```
105  search_results() {
106      echo -e "[-] Please enter a keyword to search. Results will open in a 'less' window -- Use Page Up/Page Down keys to scroll and ${MAGENTA}'q'${CLEAR} to exit."
107      echo -e "[?] If you wish to stop search and exit, enter ${MAGENTA}'n'${CLEAR} or ${MAGENTA}'q'${CLEAR}."
108
109      read keyword
110      choice=$(echo $keyword | tr '[:upper:]' '[:lower:]')
111      if  [[ $choice == 'n' || $choice == 'no' || $choice == 'q' || $choice == 'quit' ]]; then
112          echo "[-] Exiting search..."
113          echo
114      else
115          echo -e "[-] Searching for keyword ${BOLD}$keyword${CLEAR} within results:"
116          echo
117          echo
118          echo
119
120          # By first cd to the output dir, only the filenames itself will be output in the grep search - instead of full path, which results in very cluttered output.
121          cd $out_dir
122          # Grepping/searching for keyword specified by user in the key output files.
123          # -Hn flag to include filename and line number since searching across multiple file
124
125          # First line to append all search to file for archiving.
126          grep --color=always -Hn -i -A 2 -B 2 $keyword *.nmap *.out >> search_$date_time.lst
127
128          # Second line to pipe to less for immediate output to screen/user.
129          grep --color=always -Hn -i -A 2 -B 2 $keyword *.nmap *.out | less -R
130          echo
131          search_results
132          echo
133      fi
134  }
```

```
[-] End of main scan.
[?] Would kali like to search within scan results now? Enter 'Y' for Yes or 'N' for No. [Y/N]
Y
y
[-] Please enter a keyword to search. Results will open in a 'less' window -- Use Page Up/Page Down keys to scroll and 'q' to exit.
[?] If you wish to stop search and exit, enter 'n' or 'q'.
ftp
[-] Searching for keyword ftp within results:
```

```
[-] Please enter a keyword to search. Results will open in a 'less' window -- Use Page Up/Page Down keys to scroll and 'q' to exit.
[?] If you wish to stop search and exit, enter 'n' or 'q'.
ssh
[-] Searching for keyword ssh within results:
```

Keyword searches are done with grep case insensitive search and less - the latter to facilitate pagination. Below screenshots for an 'ftp' search. The filename and line number are indicated on the left column for later closer investigation.



If the user exits or chooses not to do a search, the script exits and zips up all the files generated.

```
467
468 # Housekeeping - zipping up scan and search output.
469 # Saving datetime as variable to use as as suffix in zipfilename. Can differentiate between multiple runs/zips, avoid overwriting.
470 date_time=$(date +"%F_%H:%M")
471
472 # Create new 'zipped' child directory to store zipped package.
473 # "Raw" files produced in the script retained in the parent output directory. If not needed can use '-m' flag.
474 # Retained for user to do immediate investigation instead of having to unzip.
475
476 mkdir -p $out_dir/zipped
477 zip -j $out_dir/zipped/"scan_s17_$date_time.zip" $out_dir/* -x \*.zip \*.sh
478 echo
479 echo "[-] Key results are saved in $out_dir/zipped/scan_s17_$date_time.zip"
480 echo "[-] This script will now exit. Have a nice day."
```

```
q
[-] Exiting search ...


  adding: 1_massudp (deflated 33%)
  adding: 2_nmap_tcp.gnmap (deflated 50%)
  adding: 2_nmap_tcp.nmap (deflated 50%)
  adding: 2_nmap_tcp.xml (deflated 77%)
  adding: 3_nmap_udp.gnmap (deflated 34%)
  adding: 3_nmap_udp.nmap (deflated 33%)
  adding: 3_nmap_udp.xml (deflated 58%)
  adding: 4_hydra.out (deflated 79%)
  adding: 5_nmap_vulners.out (deflated 83%)
  adding: 6_sploit_all.out (deflated 89%)
  adding: 6_sploit_priority.out (deflated 88%)
  adding: 6_sploit_tcp.lst (deflated 89%)
  adding: 6_sploit_udp.lst (deflated 88%)
  adding: 6_vul_service.lst (stored 0%)
  adding: cve_only.out (deflated 87%)
  adding: cve_trim.lst (stored 0%)
  adding: passwords.lst (stored 0%)
  adding: search_.lst (deflated 92%)
  adding: users.lst (stored 0%)

[-] Key results are saved in /home/kali/pentest/proj/scan_final/zipped/scan_s17_2024-06-13_05:53.zip
[-] This script will now exit. Have a nice day.
```

3. Discussion

3.1 Script design and limitations

This section briefly outlines some choices and limitations in the script

Input validation Firstly, the current Input validation approach is not the most elegant nor robust. The IP address check is just based on a text matching of IPv4 octets. A potential improvement may be pinging the address to check that the host is up. Additionally, the script is currently designed to check a single host only.

```
53  ## Functions
54
55  # Check if IP address input by user is 'valid'.  i.e. if  it matches ipv4 format. Namely, four sets of 3 digits between 0 - 255, separated with a period.
56
57  check_ip() {
58      local ip=$1
59      local stat=1
60      ip_grep=$(echo $ip | grep -Eo '^(([1-9]?[0-9]|1[0-9][0-9]|2([0-4][0-9]|5[0-5]))\.){3}([1-9]?[0-9]|1[0-9][0-9]|2([0-4][0-9]|5[0-5]))$')
61
62      if [[ "$ip_grep" == "$ip" ]]; then
63          stat=0
64      fi
65      return $stat
66  }
67
```

Performance vs Noise For the purpose of this project speed of port scans is bumped up (masscan: --rate=10k, nmap: T5). In practice this may be too noisy.

```
204 ## Functions to do the core scanning work.
205 # Port scans with masscan and Nmap
206
207 port_scan(){
208     # UDP scan with masscan. Speed/rate bumped up to 10k, and scan all ports.
209
210     echo -e "${BOLD}$UDP1${CLEAR} Starting..."
211     sudo masscan $ip_addr -pU:1-65535 --rate=10000 -oL $out_dir/1_massudp
212
213     # Process the output from masscan to derive the list of open UDP ports.
214     # For Nmap to scan these specific UDP ports later.
215
216     udp_ports=$(cat $out_dir/1_massudp | awk '{print$3}'| grep '[0-9]' | paste -s -d, /dev/stdin)
217     echo
218
219     # TCP scan with nmap. We incorporate one optional $1 argument
220     # Toggles between scanning default 1k ports for basic scan, and all ports for full scan.
221
222     echo -e "${BOLD}$TCP${CLEAR} Scanning $tcp_ports ports.$TAKETIME"
223     nmap -oA $out_dir/2_nmap_tcp -sV -T5 $ip_addr $1
224     echo
225     echo -e "${BOLD}$UDP2${CLEAR} Starting..."
226     sudo nmap -oA $out_dir/3_nmap_udp -sU -sV $ip_addr -p $udp_ports
227
228 }
```

The weak password checks can be done in relatively short time, using the in-built but very short password (*Line 33: 'PASSW' - 5 passwords only*) and user list (*Line 34: 'USERS' - 3 IDs only*). This may not be sufficiently rigorous for a proper check.

```
18
19  # These lists are possible options chosen as relatively short lists (100+) with a  fair chance of existing on user system.
20  # However, they still take up some time for weakpassword check, in particular for telnet.
21  # As such they will be kept in 'cold storage' - for user's consideration.
22
23  # PASS_DEF='/usr/share/seclists/Passwords/darkweb2017-top100.txt'
24  # USERS_DEF='/usr/share/seclists/Usernames/top-usernames-shortlist.txt'
25
26  # For project submission and to facilitate quick script processing, will retain these 'fake lists' - intended to return a 'Not found' result.
27  PASS_DEF='/usr/share/seclists/Passwords/TEMPFAKE'
28  USERS_DEF='/usr/share/seclists/Usernames/TEMPFAKE'
29
30  # If above lists can't be found, script wll generate user and password files/lists using below arrays.
31  # >> Arrays have been kept very short for testing/efficiency. Feel free to edit/add if needed.
32
33  PASSW=("user" "msfadmin" "123456" "password" "qwerty")
34  USERS=("user" "msfadmin" "root")
35
```

```
260     else
261         # Core/key code - for each of the detected login services saved in variable $check_services, run hydra with the
262         for item in $check_services;
263         do
264             echo -e "[-] Checking weak password for ${CYAN}${BOLD}$item${CLEAR} service;"
265             echo
266             # Some brief performance optimisation was attempted by reducing -w to 10, instead of default.
267             hydra -L $userlist -P $pass_list -e ns -w 10 -o $out_dir/4_hydra.out -I $ip_addr $item
268             echo
269         done
```

The script has also defined two preferred but still relatively short password and user lists in the Seclists set (Line 23 and 24). Testing these for ftp and ssh still allowed for reasonable time for hydra checks but resulted in an unrealistically long wait for telnet. Further refinement needed for more rigorous weak password check.

```
[-] Checking weak password for telnet service;

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organiz

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-13 04:19:20
[WARNING] telnet is by its nature unreliable to analyze, if possible better choose FTP, SSH, etc. if available
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1717 login tries (l:17/p:101), ~108 tries per task
[DATA] attacking telnet://192.168.133.132:23/
[STATUS] 288.00 tries/min, 288 tries in 00:01h, 1429 to do in 00:05h, 16 active
[STATUS] 146.00 tries/min, 438 tries in 00:03h, 1279 to do in 00:09h, 16 active
[STATUS] 84.14 tries/min, 589 tries in 00:07h, 1128 to do in 00:14h, 16 active
[STATUS] 58.25 tries/min, 699 tries in 00:12h, 1018 to do in 00:18h, 16 active
[STATUS] 47.41 tries/min, 806 tries in 00:17h, 911 to do in 00:20h, 16 active
[STATUS] 41.32 tries/min, 909 tries in 00:22h, 808 to do in 00:20h, 16 active
[STATUS] 37.67 tries/min, 1017 tries in 00:27h, 700 to do in 00:19h, 16 active
[STATUS] 35.22 tries/min, 1127 tries in 00:32h, 590 to do in 00:17h, 16 active
[STATUS] 33.46 tries/min, 1238 tries in 00:37h, 479 to do in 00:15h, 16 active
[STATUS] 31.96 tries/min, 1343 tries in 00:42h, 374 to do in 00:12h, 16 active
[STATUS] 30.82 tries/min, 1450 tries in 00:47h, 267 to do in 00:09h, 16 active
[STATUS] 29.93 tries/min, 1558 tries in 00:52h, 159 to do in 00:06h, 16 active
[STATUS] 29.19 tries/min, 1666 tries in 00:57h, 51 to do in 00:02h, 16 active
[STATUS] 29.04 tries/min, 1687 tries in 00:58h, 30 to do in 00:02h, 16 active
[STATUS] 28.89 tries/min, 1707 tries in 00:59h, 10 to do in 00:01h, 16 active
[STATUS] 28.58 tries/min, 1717 tries in 01:00h, 1 to do in 00:01h, 15 active
```

Search can be improved The current keyword search uses a grep and less combination to search through multiple files. The output to screen may not be the most intuitive and does not facilitate immediate searching of the whole file. A more developed solution could be by passing the keyword to less (or other commands/tools) to open and search through the various output files themselves.

4. Conclusion & Recommendations

This project is a useful exercise to recapitulate the use of basic network scanning tools. From a script development perspective, apart from the ‘core code’ that provides the functionality, aspects like formatting and timing of output are also important to highlight the most pertinent information to the user.

Some additional observations include:

- Text manipulation is a critical skill (perhaps underappreciated) for processing of results and output to the user.
- Navigating the nuances/differences between bash and other coding/programming languages was a challenge:
    - the intricacies in calling a variable with just $var (or “$var”)
    - string vs number comparisons using “test” command vs bracket notations
    - Technically, no multi dimensions arrays in bash
    - Bash variables technically all ‘global’

5. References

Websites/references that provided guidance on this project include those listed below.

Validating IP

1. https://tecadmin.net/shell-script-validate-ipv4-addresses/
2. https://www.baeldung.com/linux/ip-address-test-valid
3. https://ioflood.com/blog/bash-boolean/

Formatting

4. https://stackoverflow.com/questions/5947742/how-to-change-the-output-color-of-echo-in-linux
5. https://misc.flogisoft.com/bash/tip_colors_and_formatting

Bash Array/Command

6. https://kodekloud.com/blog/bash-scripts-loop-through-array-values/
7. https://stackoverflow.com/questions/61821979/bash-multiline-command-to-variable-results-in-only-the-first-line

Zipping Files

8. https://unix.stackexchange.com/questions/57013/zip-all-files-in-directory
9. https://askubuntu.com/questions/261079/how-can-i-create-zip-file-with-the-date-in-its-name