Project

Penetration testing

Israel Gatterer

Objective

Creating a bash script that maps network devices for ports, services, and vulnerabilities.

Penetration testing project

echo -e "\033[0;36m……….. \033[0m" - This is a Bash command that uses the echo command to print a message to the console with colored text. The **‘-e’** option enables the interpretation of escape sequences, and **\033[0;36m** sets the text color to denim-blue.

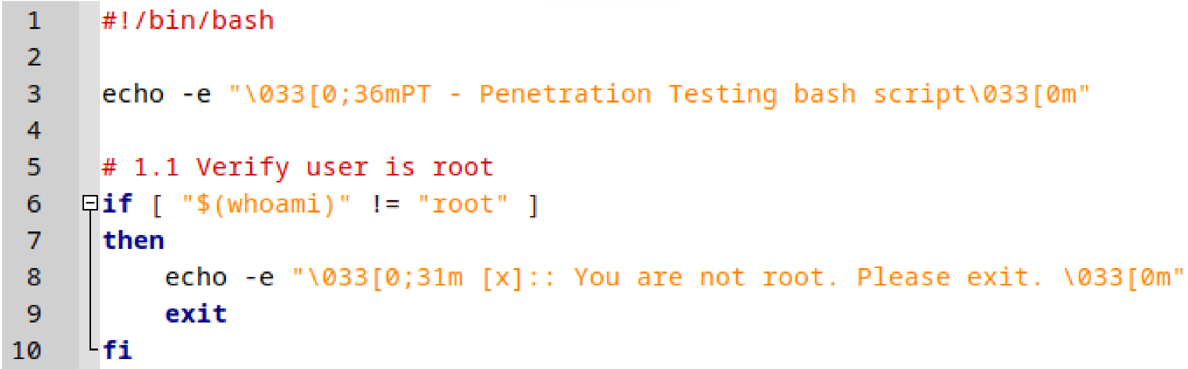
- Ignored by the shell.

The **if** statement is checking whether the output of the whoami command (which prints the current user's username) is equal to the string "root".

If the user is not root, **then** the script prints an error message in red color using the echo command and the escape sequence **"\033[0;31m"** to change the color. The message says " [x]:: You are not root. Please exit. "

The **exit** command is used to terminate the script if the user is not root.

In summary, lines 6-10 used to ensure that the script is only run as the root user, and it will exit if the user is not root.



```
1   #!/bin/bash
2
3   echo -e "\033[0;36mPT - Penetration Testing bash script\033[0m"
4
5   # 1.1 Verify user is root
6   if [ "$(whoami)" != "root" ]
7   then
8       echo -e "\033[0;31m [x]:: You are not root. Please exit. \033[0m"
9       exit
10  fi
```

function RNG - In order to map the network devices and open ports.
This function is used to identify the local network range by parsing the output of the ip add command and storing the result in the LANNET variable.

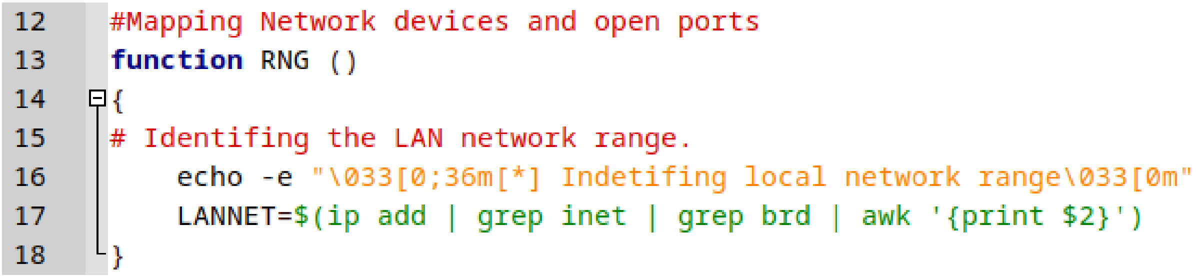
Starts with echo command - Is used to print a message in cyan color to indicate that the script is identifying the local network range.

The **ip add** command is used to display the IP addresses assigned to network interfaces. The output of ip add is piped to the **grep** command to filter only lines containing the word "inet".

The output of the first grep command is piped to the second grep command to filter only lines containing the word "brd".

The output of the second grep command is piped to the **awk** command to print the second field (which contains the IP address and subnet mask) separated by a space.

The output of the awk command is assigned to the variable **LANNET**.



```
12  #Mapping Network devices and open ports
13  function RNG ()
14  {
15  # Identifing the LAN network range.
16      echo -e "\033[0;36m[*] Indetifing local network range\033[0m"
17      LANNET=$(ip add | grep inet | grep brd | awk '{print $2}')
18  }
```

function SCAN

This function is used to scan the LAN for live hosts and vulnerabilities using the nmap tool. The results are stored in a file named "hosts", and each IP address found in the file is scanned for vulnerabilities using a port scan.

Line 25- The first echo command prints a message in cyan color to indicate that the script is scanning the LAN for live hosts.

Line 26- The second echo command prints a message in cyan color to indicate that the results will be kept in a file named "hosts".

Line 27- The **nmap** command is used to scan the LAN for live hosts by using the **-sn** option to ping all IP addresses in the specified network range without performing port scans.
The output of nmap is piped to the **grep** command to filter only lines containing the word "for".
The output of the grep command is piped to the awk command to print the last field (which contains the IP address) separated by spaces.

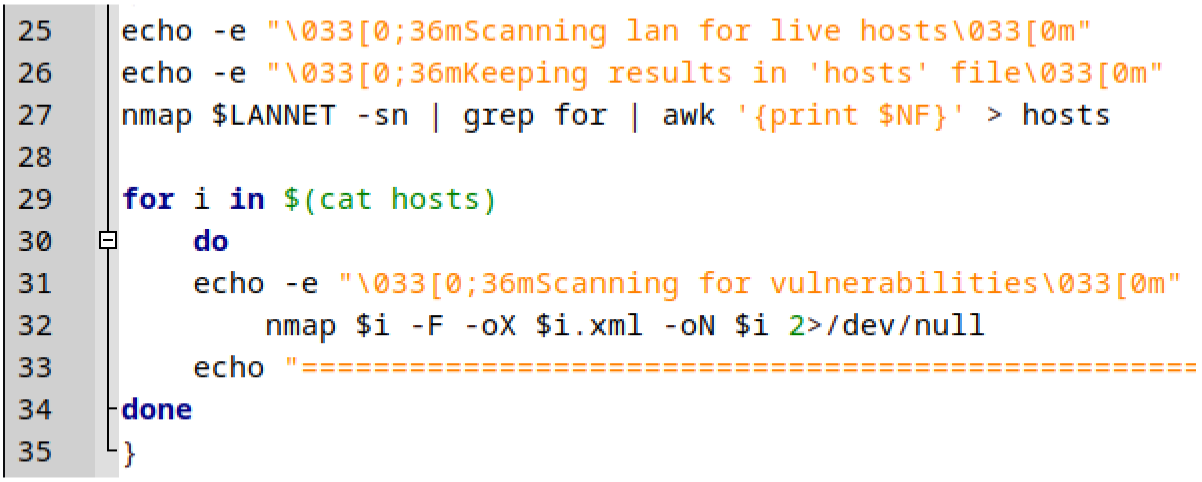
The output of the awk command is redirected (‘**>**’) to a file named "hosts".

for in ; do ; done - for loop

Lines 29-35 - A for loop is used to iterate over each IP address in the "hosts" file.
Within the loop, the echo command prints a message in cyan color to indicate that the script is scanning for vulnerabilities.

The **nmap** command is used to perform a port scan on the specified IP address (**$i**) using the **-F** option to scan only the most common ports, and the -oX and -oN options to output the results in XML and human-readable formats, respectively.

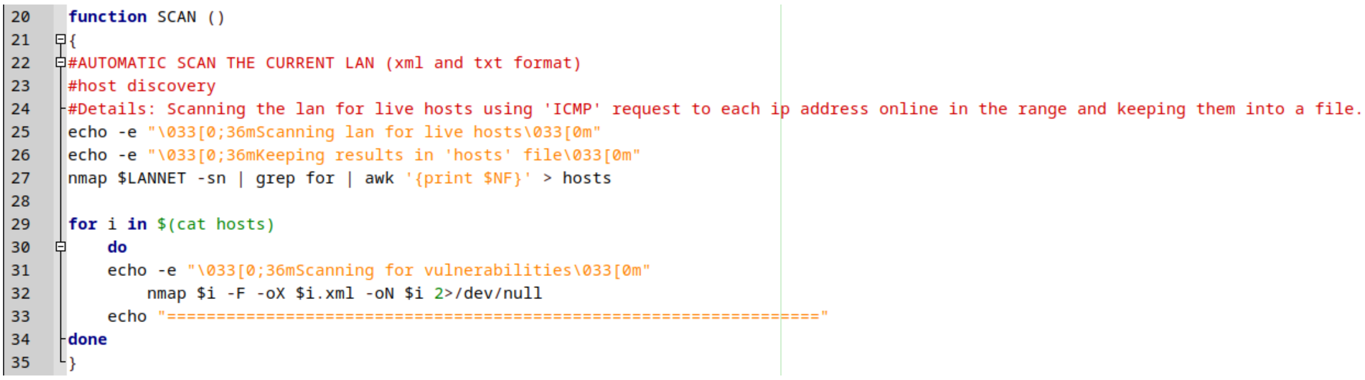
Any error messages generated by the nmap command are redirected to the **/dev/null** device.



```
echo -e "\033[0;36mScanning lan for live hosts\033[0m"
echo -e "\033[0;36mKeeping results in 'hosts' file\033[0m"
nmap $LANNET -sn | grep for | awk '{print $NF}' > hosts

for i in $(cat hosts)
    do
    echo -e "\033[0;36mScanning for vulnerabilities\033[0m"
        nmap $i -F -oX $i.xml -oN $i 2>/dev/null
    echo "=================================================
done
}
```

(functio’s zoom out)



```
function SCAN ()
{
#AUTOMATIC SCAN THE CURRENT LAN (xml and txt format)
#host discovery
#Details: Scanning the lan for live hosts using 'ICMP' request to each ip address online in the range and keeping them into a file.
echo -e "\033[0;36mScanning lan for live hosts\033[0m"
echo -e "\033[0;36mKeeping results in 'hosts' file\033[0m"
nmap $LANNET -sn | grep for | awk '{print $NF}' > hosts

for i in $(cat hosts)
    do
    echo -e "\033[0;36mScanning for vulnerabilities\033[0m"
        nmap $i -F -oX $i.xml -oN $i 2>/dev/null
    echo "==================================================================="
done
}
```

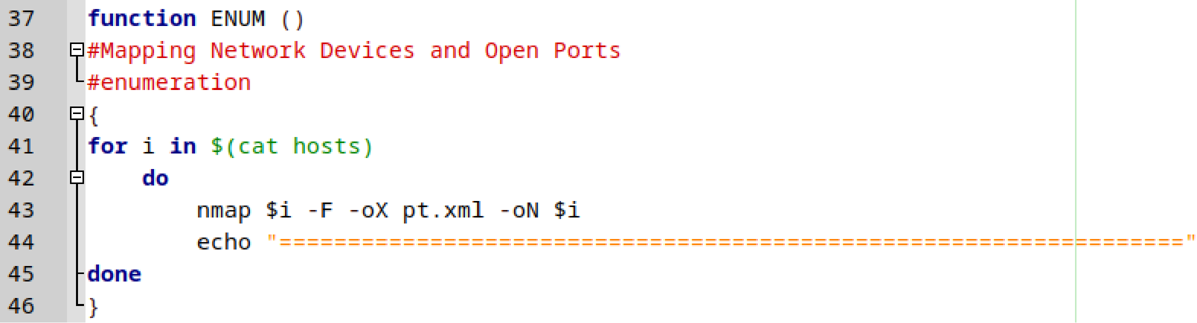
function ENUM

This function is used to perform a port scan on each IP address found in the "hosts" file using the nmap tool. The results are stored in files named after the corresponding IP addresses, and each file contains the output of the port scan in both XML and human-readable formats.

The for loop is used to iterate over each IP address in the "hosts" file.

Within the loop, the nmap command is used to perform a port scan on the specified IP address ($i) using the -F option to scan only the most common ports, and the -oX and -oN options to output the results in XML and human-readable formats, respectively.

The output of nmap is redirected to a file named after the IP address being scanned.



```
function ENUM ()
#Mapping Network Devices and Open Ports
#enumeration
{
for i in $(cat hosts)
    do
        nmap $i -F -oX pt.xml -oN $i
        echo "================================================================="
done
}
```

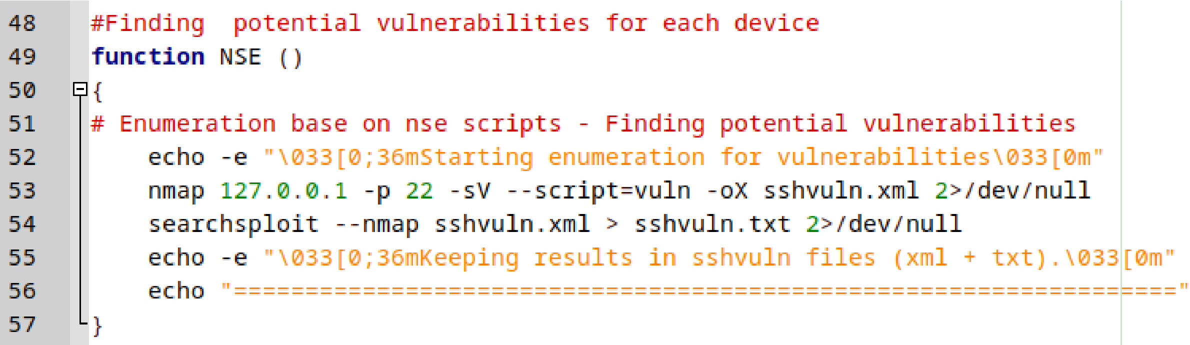
function NSE

The **nmap** command is used to perform a version scan on the local machine's SSH service running on port 22 (-p 22 -sV). The **--script=vuln** option tells nmap to run vulnerability detection scripts, and the **-oX** option specifies the output format as **XML**.

Any error messages generated by the nmap command are redirected to the **/dev/null** device.

The **searchsploit** command is used to search for exploits in the searchsploit database that correspond to the vulnerabilities detected by nmap in the previous step. The **--nmap** option tells searchsploit to use the XML output generated by nmap as input, and the output is redirected to a file named "sshvuln.txt".

The echo command prints a message in cyan color to indicate that the results are being stored in files named "sshvuln.xml" and "sshvuln.txt".



```
#Finding  potential vulnerabilities for each device
function NSE ()
{
# Enumeration base on nse scripts - Finding potential vulnerabilities
    echo -e "\033[0;36mStarting enumeration for vulnerabilities\033[0m"
    nmap 127.0.0.1 -p 22 -sV --script=vuln -oX sshvuln.xml 2>/dev/null
    searchsploit --nmap sshvuln.xml > sshvuln.txt 2>/dev/null
    echo -e "\033[0;36mKeeping results in sshvuln files (xml + txt).\033[0m"
    echo "================================================================="
}
```

function HYDRA

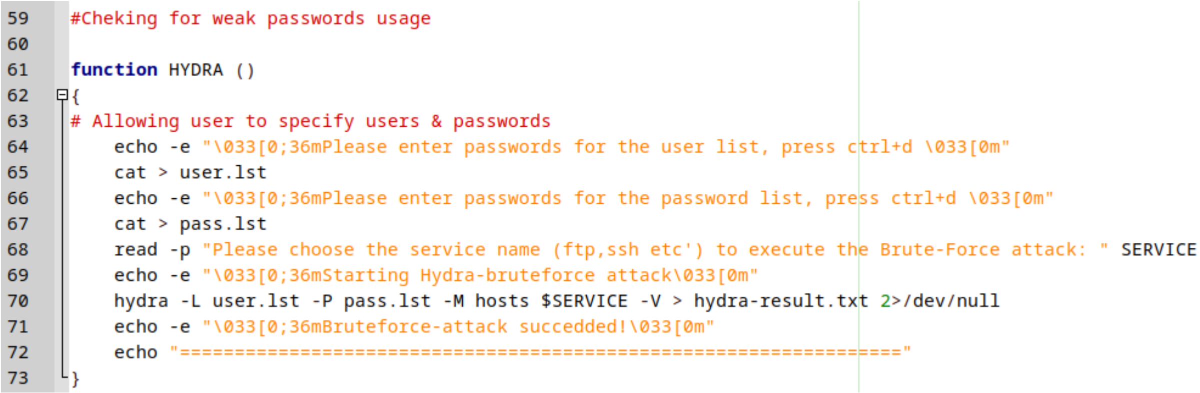
This function uses the Hydra tool for performing a brute-force attack against a specified service, such as FTP or SSH. The function prompts the user to input a list of usernames and passwords, and then executes the attack using the Hydra command. The results of the attack are saved to a file called "hydra-result.txt".

This code snippet executes a brute-force attack using the Hydra tool with the previously specified user and password lists (user.lst and pass.lst, respectively) and the service name ($SERVICE).

-L user.lst: Specifies the file containing a list of usernames to use in the brute-force attack
-P pass.lst: Specifies the file containing a list of passwords to use in the brute-force attack

-M hosts: Specifies that the attack should be performed against the hosts file (who containes the results from the vulnerabilities nmap scanning in the LAN).

$SERVICE: Specifies the name of the service to attack by the attacker/user.



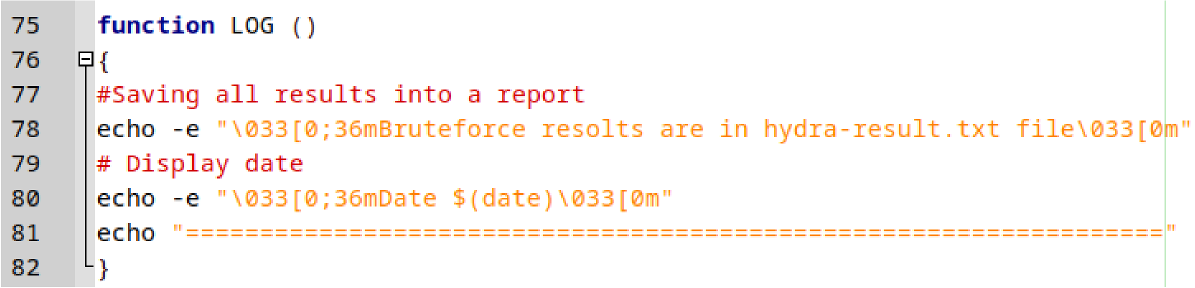
```
#Cheking for weak passwords usage

function HYDRA ()
{
# Allowing user to specify users & passwords
    echo -e "\033[0;36mPlease enter passwords for the user list, press ctrl+d \033[0m"
    cat > user.lst
    echo -e "\033[0;36mPlease enter passwords for the password list, press ctrl+d \033[0m"
    cat > pass.lst
    read -p "Please choose the service name (ftp,ssh etc') to execute the Brute-Force attack: " SERVICE
    echo -e "\033[0;36mStarting Hydra-bruteforce attack\033[0m"
    hydra -L user.lst -P pass.lst -M hosts $SERVICE -V > hydra-result.txt 2>/dev/null
    echo -e "\033[0;36mBruteforce-attack succedded!\033[0m"
    echo "==================================================================="
}
```

function LOG

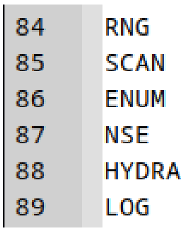
Using ANSI escape codes the message indicates that the results of the brute force attack are saved in a file named hydra-result.txt.

The second statement also uses the echo command to print the current date and time to the console in cyan color using ANSI escape codes.



```
function LOG ()
{
#Saving all results into a report
echo -e "\033[0;36mBruteforce resolts are in hydra-result.txt file\033[0m"
# Display date
echo -e "\033[0;36mDate $(date)\033[0m"
echo "=================================================================="
}
```

At the end of the script, the functions are called by typing their names (RNG, SCAN etc’). This will execute the code inside the “name”() function and print the resolts to the console.



```
RNG
SCAN
ENUM
NSE
HYDRA
LOG
```