Name: reut abergel

SID:s8

Unit: TMagen7736.38

Program code: ZX301

penetration testing project

Table of Contents:

1. INTRODUCTION

This guide explains the mechanics of my automated Penetration Testing script. The tool is designed to take an IP range, perform a full or basic scan, find vulnerabilities, and perform credential brute-forcing. While the script runs, it makes a file for the user to read with all the info that was gathered(for each ip).

1.2 Who is this guide for?

1. **For students** learning how to chain security tools like Nmap, SearchSploit, and Hydra.
2. **to organizations** that want to be aware of their own vulnerabilities.

1.3 What you will need

1. **Kali Linux:** The script relies on pre-installed security tools.
2. **Root Privileges:** Required for Nmap "Stealth" scans (-sS) and OS fingerprinting (-O).
3. **Tools Installed:**

Nmap: For network discovery and NSE scripting.

Searchsploit: To correlate service versions with known exploits.

Hydra: For automated credential testing.

1.4 How to Run

1. **Execution:** Run the script with root permissions: `sudo ./pentest_proj.sh`
2. **Setup:** The script will ask for:

Output Directory: Where all logs, XMLs, and reports will be saved.

Target Range: A valid IP range (example: 192.168.1.0/24).

2. Purpose of the Tool

The script automates the Tedious parts of the penetration test:

1. **Validation:** It verifies the IP range is valid before wasting time on a scan.
2. **Vulnerability Mapping:** Automatically converts Nmap XML output into a Searchsploit query to find relevant CVEs.
3. **Credential Auditing with Hydra:** It identifies open login services (SSH, FTP, RDP) and automatically launches brute-force attacks using internal or custom wordlists.

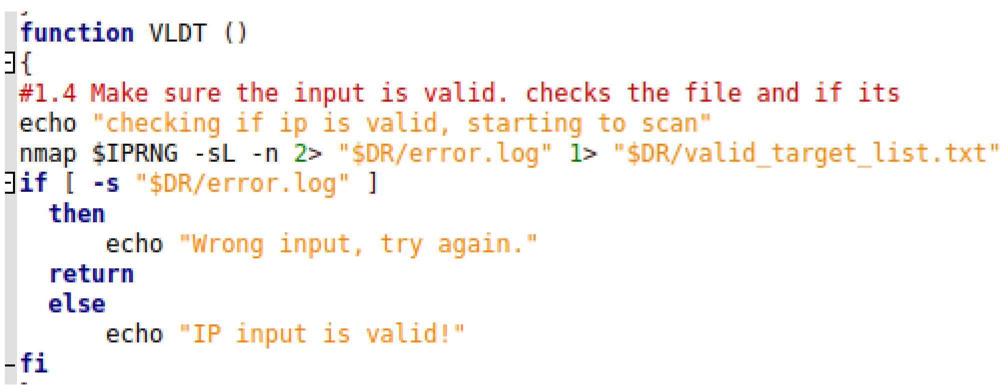
3. What I learned from building the script

- **for loops:** i learnd more about for loops and how to use them inside functions and with complex statements inside them
- **Cleaning the Output:** I used grep -E with color coding to ensure that out of thousands of lines of Nmap data, the user only sees the Important Findings, like open ports and vulnerabilities.
- **Safety Checks for Empty Lines:** Throughout all the script i use if statements to validate the input/output
- **case.** I learned how to use case statements in all sorts of ways to make it easy to use with commands
- **export**: I learned to use variable exporting to keep the script organized as it moves between different security tools.
- **Automate with Hydra and Nmap**: I learned how to use those tools better inside a script, insidea case in if statements

4. Analysis of complex commands

Function: VLDT (Validation)

This function goes through all the ip range you gave it and ensures the target is reachable and valid before the heavy scanning begins.
It uses Nmap's "List Scan" (-sL), which doesn't send packets to the target, making it a fast way to verify if the syntax of the IP range is correct.



```
function VLDT ()
{
#1.4 Make sure the input is valid. checks the file and if its
echo "checking if ip is valid, starting to scan"
nmap $IPRNG -sL -n 2> "$DR/error.log" 1> "$DR/valid_target_list.txt"
if [ -s "$DR/error.log" ]
  then
      echo "Wrong input, try again."
  return
  else
      echo "IP input is valid!"
fi
```

Function: FRSTSCN ()

Creates a dedicated "**checkip_results**" subdirectory within the main workspace to organize individual host data.

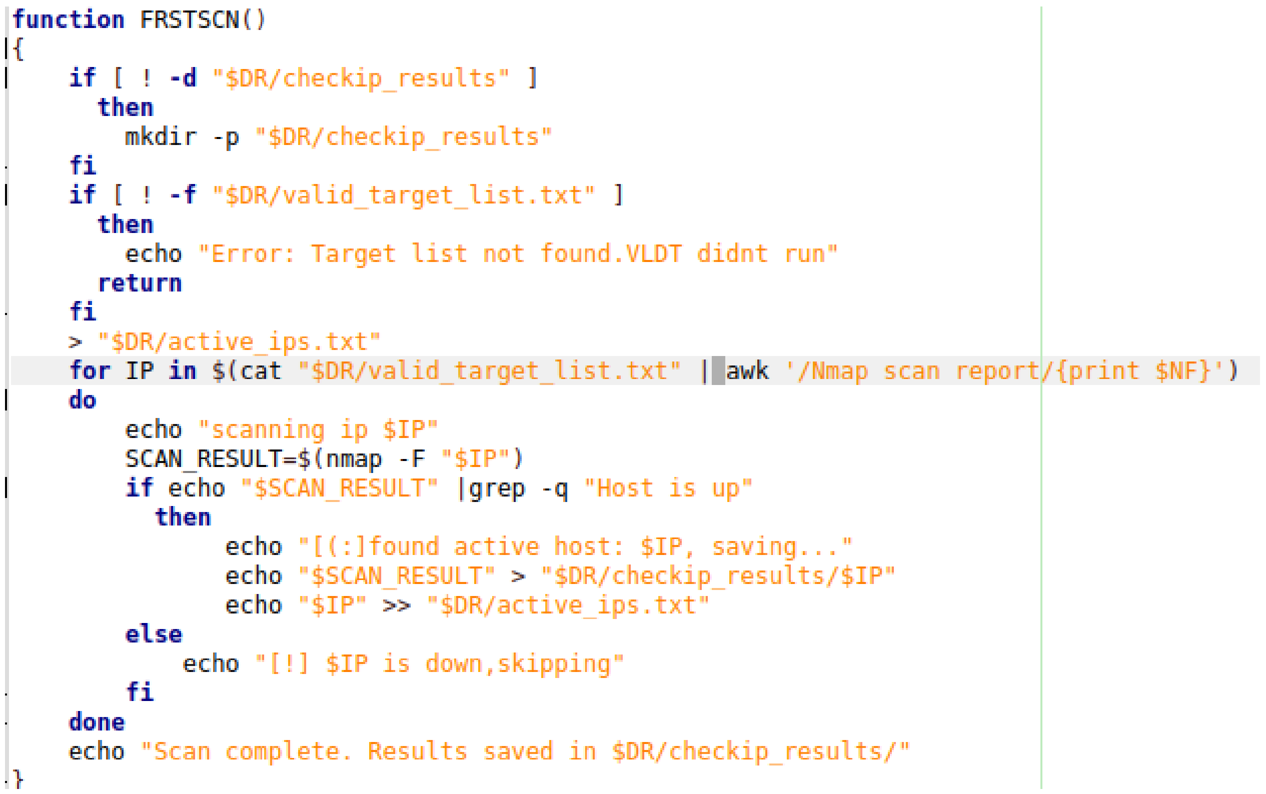
Checks for the existence of "**valid_target_list.txt**" before execution, if the file is missing,

terminates if a valid ip file wasn't found, and displays an error message.

parses the target list to extract IP addresses using "awk NF."

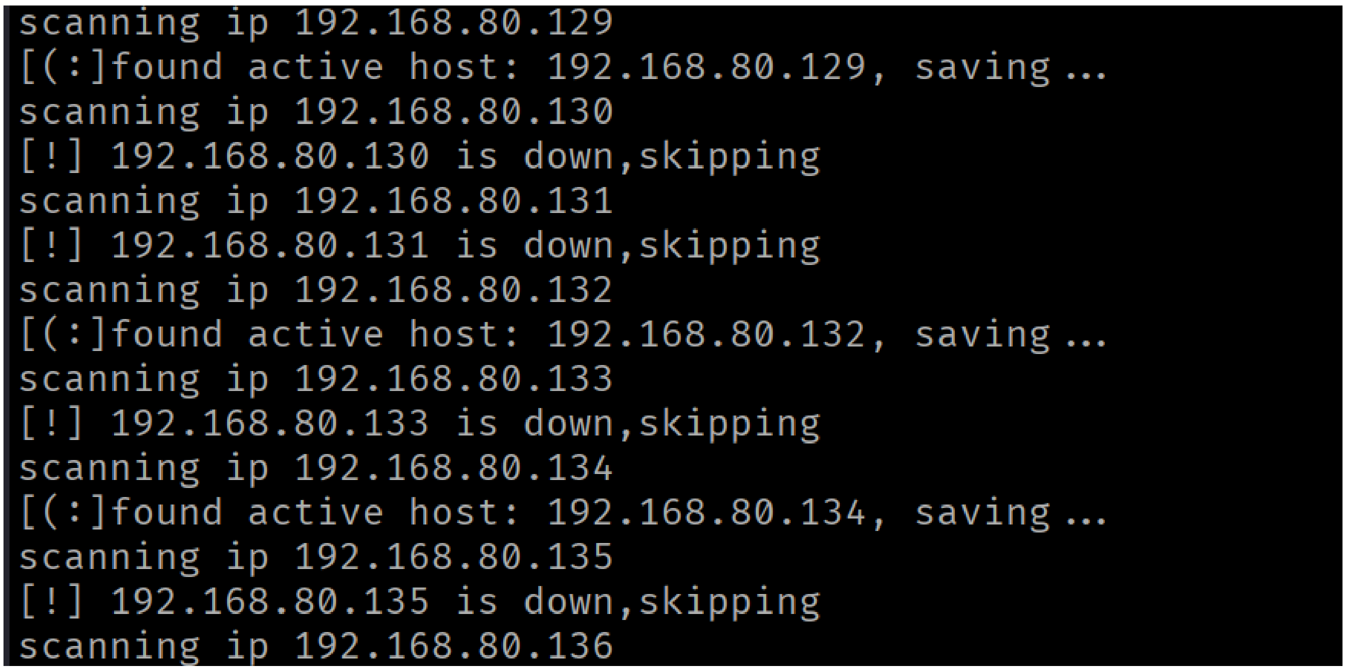
Executes a fast Nmap scan "nmap -F" against each IP to determine its status.

and the final stage of this scan, if a host is identified as active ("Host is up"), the script saves the full Nmap output to a unique file named after the IP and logs the IP to "**active_ips.txt"** for easy use and display.



```
function FRSTSCN()
{
    if [ ! -d "$DR/checkip_results" ]
      then
        mkdir -p "$DR/checkip_results"
    fi
    if [ ! -f "$DR/valid_target_list.txt" ]
      then
        echo "Error: Target list not found.VLDT didnt run"
      return
    fi
    > "$DR/active_ips.txt"
    for IP in $(cat "$DR/valid_target_list.txt" | awk '/Nmap scan report/{print $NF}')
    do
        echo "scanning ip $IP"
        SCAN_RESULT=$(nmap -F "$IP")
        if echo "$SCAN_RESULT" |grep -q "Host is up"
          then
               echo "[(:]found active host: $IP, saving..."
               echo "$SCAN_RESULT" > "$DR/checkip_results/$IP"
               echo "$IP" >> "$DR/active_ips.txt"
        else
            echo "[!] $IP is down,skipping"
        fi
    done
    echo "Scan complete. Results saved in $DR/checkip_results/"
}
```

<u>This is how it looks while running the script:</u>

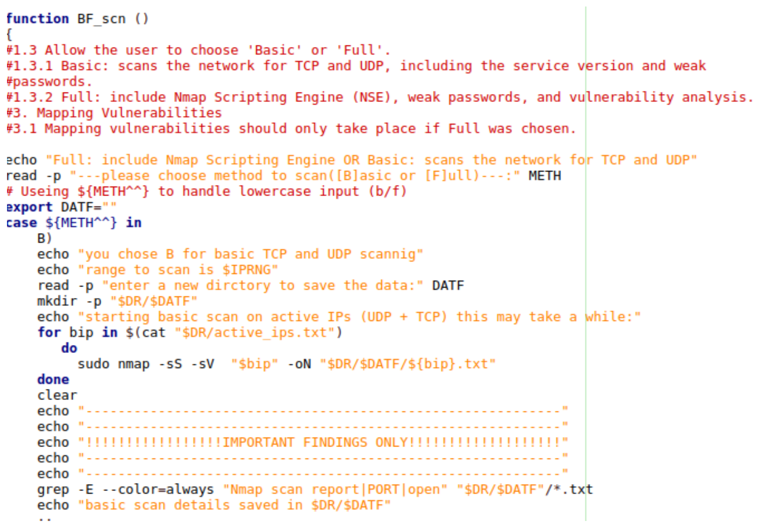


```
scanning ip 192.168.80.129
[(:]found active host: 192.168.80.129, saving...
scanning ip 192.168.80.130
[!] 192.168.80.130 is down,skipping
scanning ip 192.168.80.131
[!] 192.168.80.131 is down,skipping
scanning ip 192.168.80.132
[(:]found active host: 192.168.80.132, saving...
scanning ip 192.168.80.133
[!] 192.168.80.133 is down,skipping
scanning ip 192.168.80.134
[(:]found active host: 192.168.80.134, saving...
scanning ip 192.168.80.135
[!] 192.168.80.135 is down,skipping
scanning ip 192.168.80.136
```

Function: BF_scn (basic or full scan with nmap scripts)

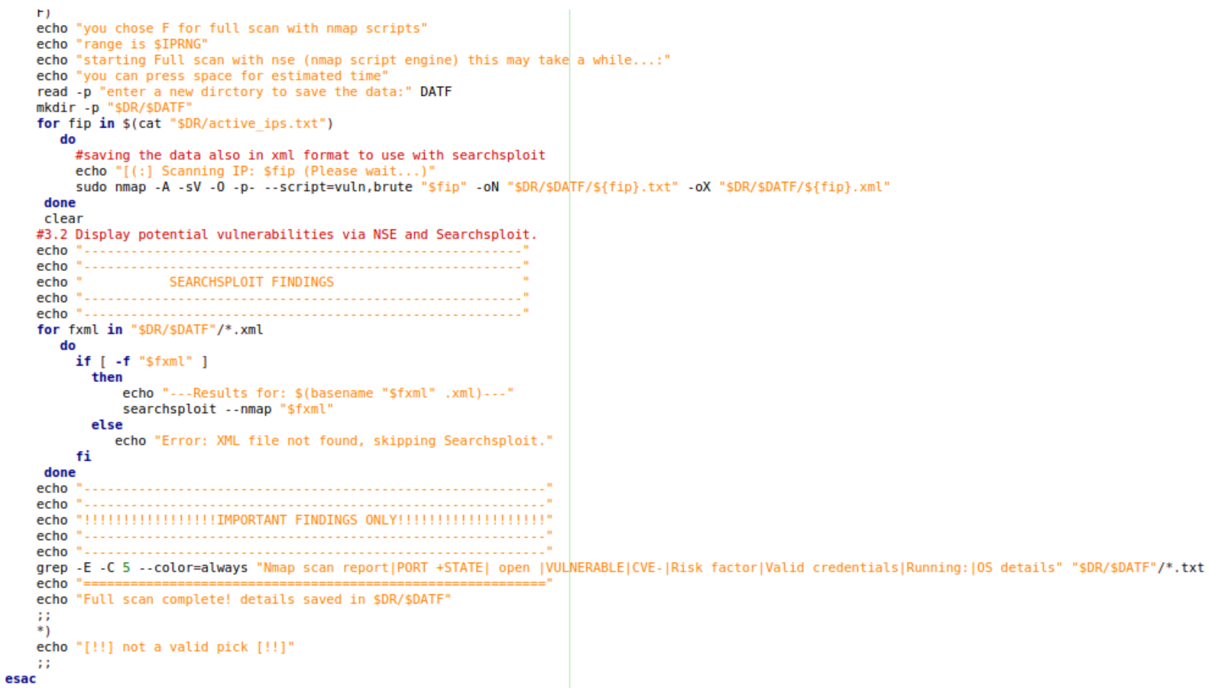
This is the core of the tool. It allows the user to choose between a **Basic** or **Full** scan.

Basic normal nmap scan: scans, shows you on the screen only the important findings based on keywords I defined, and saves the rest of the data to a file



```
function BF_scn ()
{
#1.3 Allow the user to choose 'Basic' or 'Full'.
#1.3.1 Basic: scans the network for TCP and UDP, including the service version and weak
#passwords.
#1.3.2 Full: include Nmap Scripting Engine (NSE), weak passwords, and vulnerability analysis.
#3. Mapping Vulnerabilities
#3.1 Mapping vulnerabilities should only take place if Full was chosen.

echo "Full: include Nmap Scripting Engine OR Basic: scans the network for TCP and UDP"
read -p "---please choose method to scan([B]asic or [F]ull)---:" METH
# Useing ${METH^^} to handle lowercase input (b/f)
export DATF=""
case ${METH^^} in
    B)
    echo "you chose B for basic TCP and UDP scannig"
    echo "range to scan is $IPRNG"
    read -p "enter a new dirctory to save the data:" DATF
    mkdir -p "$DR/$DATF"
    echo "starting basic scan on active IPs (UDP + TCP) this may take a while:"
    for bip in $(cat "$DR/active_ips.txt")
       do
         sudo nmap -sS -sV  "$bip" -oN "$DR/$DATF/${bip}.txt"
    done
    clear
    echo "-----------------------------------------------------------"
    echo "-----------------------------------------------------------"
    echo "!!!!!!!!!!!!!!!!!IMPORTANT FINDINGS ONLY!!!!!!!!!!!!!!!!!!!"
    echo "-----------------------------------------------------------"
    echo "-----------------------------------------------------------"
    grep -E --color=always "Nmap scan report|PORT|open" "$DR/$DATF"/*.txt
    echo "basic scan details saved in $DR/$DATF"
```

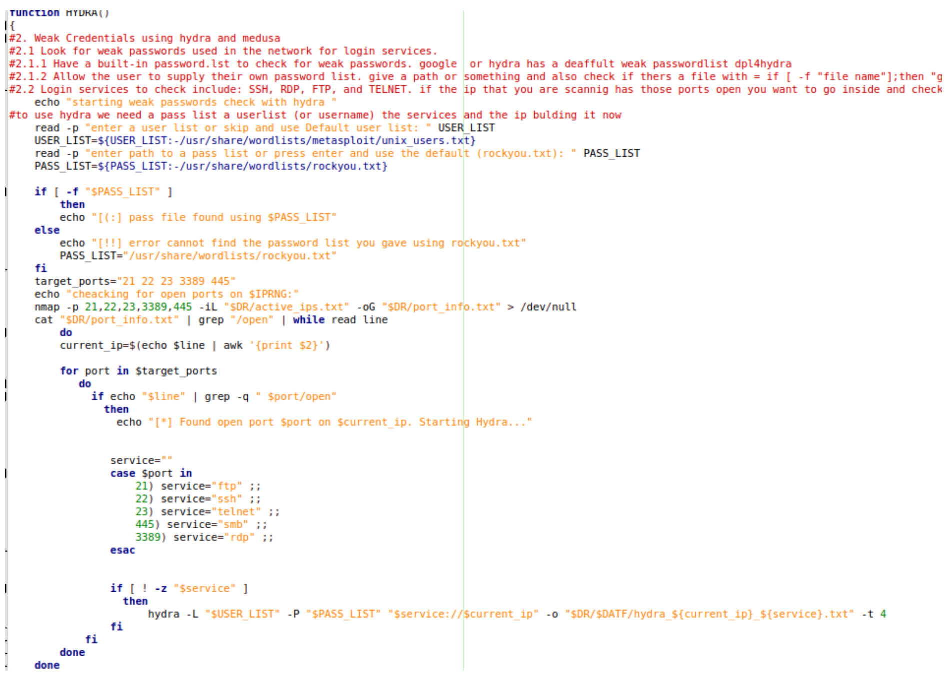
full scan with nmap scripts and searchsploit: the nmap nse finds vulnerabilities on the client, and then searchsploit reads the Nmap XML file directly. It matches the service versions found by Nmap against the Exploit-DB Database automatically.



```
F)
echo "you chose F for full scan with nmap scripts"
echo "range is $IPRNG"
echo "starting Full scan with nse (nmap script engine) this may take a while...:"
echo "you can press space for estimated time"
read -p "enter a new dirctory to save the data:" DATF
mkdir -p "$DR/$DATF"
for fip in $(cat "$DR/active_ips.txt")
   do
     #saving the data also in xml format to use with searchsploit
     echo "[(:] Scanning IP: $fip (Please wait...)"
     sudo nmap -A -sV -O -p- --script=vuln,brute "$fip" -oN "$DR/$DATF/${fip}.txt" -oX "$DR/$DATF/${fip}.xml"
 done
 clear
#3.2 Display potential vulnerabilities via NSE and Searchsploit.
echo "---------------------------------------------------------"
echo "---------------------------------------------------------"
echo "           SEARCHSPLOIT FINDINGS                         "
echo "---------------------------------------------------------"
echo "---------------------------------------------------------"
for fxml in "$DR/$DATF"/*.xml
   do
     if [ -f "$fxml" ]
       then
           echo "---Results for: $(basename "$fxml" .xml)---"
           searchsploit --nmap "$fxml"
       else
          echo "Error: XML file not found, skipping Searchsploit."
     fi
 done
echo "------------------------------------------------------------"
echo "------------------------------------------------------------"
echo "!!!!!!!!!!!!!!!!IMPORTANT FINDINGS ONLY!!!!!!!!!!!!!!!!!!!"
echo "------------------------------------------------------------"
echo "------------------------------------------------------------"
grep -E -C 5 --color=always "Nmap scan report|PORT +STATE| open |VULNERABLE|CVE-|Risk factor|Valid credentials|Running:|OS details" "$DR/$DATF"/*.txt
echo "============================================================"
echo "Full scan complete! details saved in $DR/$DATF"
;;
*)
echo "[!!] not a valid pick [!!]"
;;
esac
```

Function: HYDRA (Credential Testing)

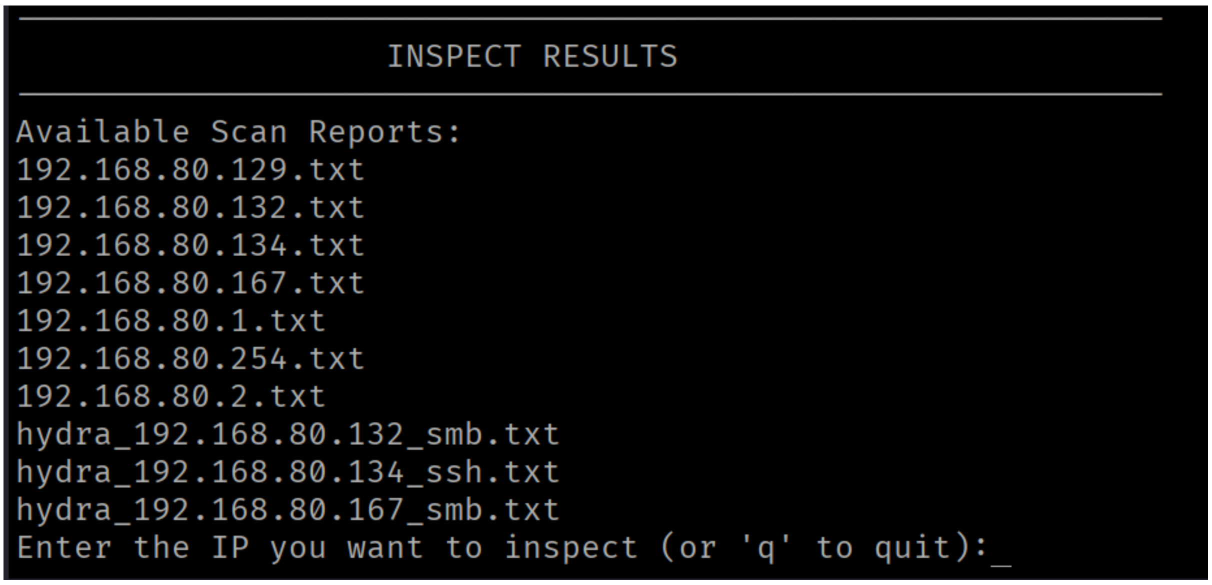
This function automates the mapping of open ports to their corresponding services to facilitate targeted brute-force attacks. It features an intelligent credential-handling logic that defaults to rockyou.txt and standard username lists unless custom paths are provided by the user. The script dynamically generates Hydra commands based on detected service protocols and verifies port accessibility before initiating credential validation.



```
function HYDRA()
{
#2. Weak Credentials using hydra and medusa
#2.1 Look for weak passwords used in the network for login services.
#2.1.1 Have a built-in password.lst to check for weak passwords. google  or hydra has a deafful weak passwordlist dpl4hydra
#2.1.2 Allow the user to supply their own password list. give a path or something and also check if thers a file with = if [ -f "file name"];then "g
#2.2 Login services to check include: SSH, RDP, FTP, and TELNET. if the ip that you are scannig has those ports open you want to go inside and check
    echo "starting weak passwords check with hydra "
#to use hydra we need a pass list a userlist (or username) the services and the ip bulding it now
    read -p "enter a user list or skip and use Default user list: " USER_LIST
    USER_LIST=${USER_LIST:-/usr/share/wordlists/metasploit/unix_users.txt}
    read -p "enter path to a pass list or press enter and use the default (rockyou.txt): " PASS_LIST
    PASS_LIST=${PASS_LIST:-/usr/share/wordlists/rockyou.txt}

    if [ -f "$PASS_LIST" ]
        then
        echo "[(:] pass file found using $PASS_LIST"
    else
        echo "[!!] error cannot find the password list you gave using rockyou.txt"
        PASS_LIST="/usr/share/wordlists/rockyou.txt"
    fi
    target_ports="21 22 23 3389 445"
    echo "cheacking for open ports on $IPRNG:"
    nmap -p 21,22,23,3389,445 -iL "$DR/active_ips.txt" -oG "$DR/port_info.txt" > /dev/null
    cat "$DR/port_info.txt" | grep "/open" | while read line
        do
        current_ip=$(echo $line | awk '{print $2}')

        for port in $target_ports
           do
             if echo "$line" | grep -q " $port/open"
               then
                 echo "[*] Found open port $port on $current_ip. Starting Hydra..."


                service=""
                case $port in
                    21) service="ftp" ;;
                    22) service="ssh" ;;
                    23) service="telnet" ;;
                    445) service="smb" ;;
                    3389) service="rdp" ;;
                esac


                if [ ! -z "$service" ]
                  then
                      hydra -L "$USER_LIST" -P "$PASS_LIST" "$service://$current_ip" -o "$DR/$DATF/hydra_${current_ip}_${service}.txt" -t 4
                fi
            fi
        done
    done
```

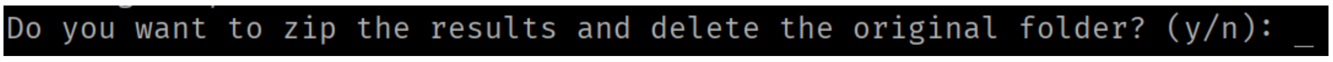
5. Summary of output

At the end of the process, the script provides a structured summary. You can see the total findings in the terminal, similar to the image below:



```
                       INSPECT RESULTS

Available Scan Reports:
192.168.80.129.txt
192.168.80.132.txt
192.168.80.134.txt
192.168.80.167.txt
192.168.80.1.txt
192.168.80.254.txt
192.168.80.2.txt
hydra_192.168.80.132_smb.txt
hydra_192.168.80.134_ssh.txt
hydra_192.168.80.167_smb.txt
Enter the IP you want to inspect (or 'q' to quit):_
```

All evidence is zipped into a timestamped file, and the workspace is cleaned up to maintain Cleanliness.



```
Do you want to zip the results and delete the original folder? (y/n): _
```

Summary:

This project is a tool that makes it easy to check a network for security weaknesses. Instead of running many different programs one by one, this script links them together so they work as a single, automatic process. It is designed to be simple to use, even for complex tasks.

```
(^\.            .^\.)
((\^^.        .^^))
  \ ' \.-----./   /
   \              /
   (            )    _ ._ ^
   ( (o  _  o)  )   \./       )
   (0   \^/   0)  \      ___\
   |            |   .)  \
   /  \    /    \  (). ^
  /  \ \  / /    (  .. )
 (      _) (_/    ). ^
 (                )
  \.             ./
    (__)----(__)
```

```
.------------------------------------------------.
|                                                |
|           Thank you for using this tool!        |
|________________________________________________|
```