Network Research

Title Page

- **Title:** Network Research
- **Name:** Muhammad Fuad Bin Zulkifli
- **Student Code:** S13
- **Unit Code:** CFC060524
- **Trainer Name:** Samson
- **Date:** 25 July 2024

Table of contents

1. Introductions

This report details the functionality of an automated script designed to enhance user anonymity and security. The script performs requisite application checks for Geoip-bin, TOR, SSH pass, checkinstall, and NIPE. It also automates the installation of these applications if they are not already present on the user's system.

The script uses condition statements such as IF-THEN, CASE and Function. These statements handle different scenarios and make decisions based on the conditions set in the statement. The usage of the IF-THEN statement enables the script to execute specific commands only if the conditions are met. While for the CASE statement, it manages more complex tasks, allowing the script to perform multiple actions based on a set of variables. The function stores variable commands to be used when they are needed in the script. These condition statements allow the script to perform multiple inputs and situations.

In addition, the script includes error-handling measures such as verifying whether the user's connection is anonymous. If the user is not anonymous, the script will terminate itself, prompting the user to re-run it to ensure its stability and reliability during execution.

During the script's execution, it uses tools such as WHOIS and NMAP to scan inputs provided by the user, giving detailed information about where the scan data is stored in the machine.

This report will detail the problem of using unsecured protocols such as FTP and elaborate on how it impacts the CIA TRIAD. It will also examine the script's components, tools, and development process, Using Wireshark to analyse and capture network traffic. This report will elaborate on the vulnerabilities of using unsecured FTP vs. secure alternatives like SCP.

The report will also discuss the script's efficiency and potential areas for further improvement.

2. Methodologies

2.1. DPKG

The `dpkg` command ensures that essential packages like Geoip-bin, TOR, sshpass, checkinstall and NIPE are installed on the system. By checking the status of these packages, the script can confirm that all necessary dependencies are present.

`--status` — Verifies the installation status of essential packages.

`--install` — to install the package to the system

2.2. Geoip-bin and the CURL command

```
geoiplookup <IP Address Input>
```

```
curl --silent ifconfig.io
```

The `geoiplookup` command combined with `curl`, helps the script determine the geographical origin of an IP address provided by the user. This information can be crucial for understanding the location-based aspects of network traffic.

The `curl` command is a tool for transferring data from or to a server using URLs.

`--silent` — Suppresses output, including the progress meter and error messages.

2.3. TOR

TOR (The Onion Router) is a free, open-source software that enhances privacy and security by directing internet traffic through encrypted virtual tunnels. **TOR**

uses onion routing, which involves encrypting data multiple times and sending it through a network of relays run by volunteers worldwide. Each relay decrypts a layer of encryption to reveal only the next hop in the path, thus masking the source and destination of the data. This process helps to conceal the user's identity and protect their online activities from surveillance and traffic analysis.

2.4. checkinstall

`checkinstall` is used to create and manage software packages on Debian-based systems. It helps to create a Debian package from manually installed software.

`--install` — Toggle installation of the created package.

`--pkgname` — set the package name

`--pkgversion` — set the package version

`--default` — Default answers to all prompts/questions.

2.5. NIPE.pl

`Nipe.pl` is a Perl tool that lets users use the Tor network as their default gateway. It routes the machine's traffic through the Tor network, helping the user browse the Internet with better privacy and anonymity. *(Gouvêa, H. ,2023)*

`sudo perl nipe.pl status` — To check the status for the nipe connection

`sudo perl nipe.pl stop` — To stop nipe connection

`sudo perl nipe.pl start` — To start nipe connection

`sudo perl nipe.pl restart` — To restart the nipe connection

2.6. SSHPASS & SSH

Using `sshpass` in the script automates SSH logins by providing a non-interactive password. `sshpass` is useful in automated scripts where manual password entry is impractical.

`sshpass -p` — password

`ssh -T` — The command connects to a remote server via the Secure Shell (SSH) protocol. The `-T` option disables pseudo-terminal allocation, which is helpful for executing the script that doesn't require interactive input.

`ssh -o StrictHostKEyChecking=no` — This option instructs SSH not to prompt for confirmation when connecting to a new host for the first time.

2.7. WHOIS

The `WHOIS` command retrieves information about domain names, IP addresses, and network ownership.

`whois <IP address>` — command is utilised in the script to gather metadata about domains and IP addresses. T

2.8. NMAP

`nmap` is a powerful network scanning tool used to discover hosts and services on a computer network by sending packets and analysing the responses.

`nmap -Pn` — Treat all hosts as online (skip host discovery)

`nmap -sV` — Probe open ports to determine service/version info

`-on` — Output scan in normal such as .txt file.

3. Discussions

The primary objective of this project is to develop a script that enhances user anonymity and security by automating the installation of essential tools such as TOR, Geoip-bin, sshpass, checkinstall and NIPE. The script incorporates scanning tools such as NMAP to gather open services in the network and utilise WHOIS for information gathering.

3.1 Installation of essential applications using the IF-THEN statement and dpkg.

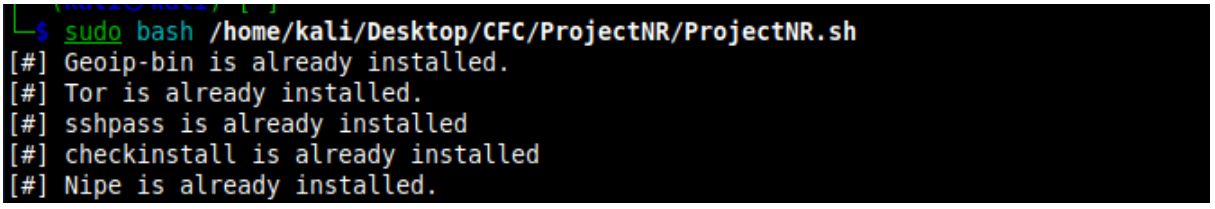
Variables for application checker

```
Geoipbin=$(dpkg --status geoip-bin)
sshpass=$(dpkg --status sshpass)
Tor=$(dpkg --status tor)
NIPE=$(dpkg --status nipe)
Pkgchecker=$(dpkg --status checkinstall)
Cpanminus=$(dpkg --status cpanminus)
```

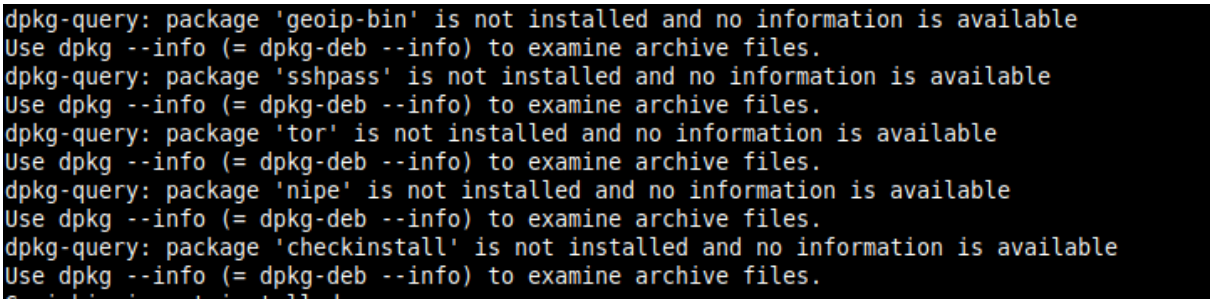
Using the IF-THEN statement to check if the essential application is present in the system. If it's not the condition, it will automate installation for the user.

```
if [[ $Application == *"Status: install ok installed"* ]]
then
    echo '[#] Application is already installed.'
else
    echo 'Application is not installed'
    sleep 1
    echo 'Installing $Application....'
    sleep 1
    sudo apt-get update > /dev/null 2>&1
    sudo apt-get install --assume-yes $Application > /dev/n
ull 2>&1
    echo '$Application is installed.'
fi
```

`if [[ $Application == *"Status: install ok install"* ]]` — This condition cross-checks with `dpkg` to verify and confirm that the essential application is present in the system.



If the package is not present in the package management system.



If the application is not installed or present in the `dpkg`, the script will install the package for the user.

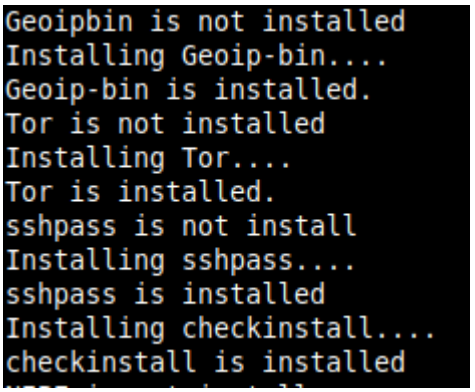
```
#Check required for applications and install if necessary.
if [[ $Geoipbin == *"Status: install ok installed"* ]]
then
    echo '[#] Geoip-bin is already installed.'
else
    echo 'Geoipbin is not installed'
    sleep 1
    echo 'Installing Geoip-bin....'
    sleep 1
    sudo apt-get update > /dev/null 2>&1
    sleep 1
    sudo apt-get install --assume-yes geoip-bin > /dev/null 2>&1
    echo 'Geoip-bin is installed.'
fi

if [[ $Tor == *"Status: install ok installed"* ]]
then
    echo '[#] Tor is already installed.'
else
    echo 'Tor is not installed'
    sleep 1
    echo 'Installing Tor....'
    sleep 1
    sudo apt-get update > /dev/null 2>&1
    sleep 1
    sudo apt-get install --assume-yes tor > /dev/null 2>&1
    echo 'Tor is installed.'
fi

if [[ $sshpass == *"Status: install ok installed"* ]]
then
    echo '[#] sshpass is already installed'
else
    echo 'sshpass is not install'
    sleep 1
    echo 'Installing sshpass....'
    sleep 1
    sudo apt-get update > /dev/null 2>&1
    sleep 1
    sudo apt-get install --assume-yes sshpass > /dev/null 2>&1
    echo 'sshpass is installed'
fi

if [[ $Pkgchecker == *"Status: install ok installed"* ]]
then
    echo '[#] checkinstall is already installed'
else
    echo 'Installing checkinstall....'
    sleep 1
    sudo apt-get install --assume-yes checkinstall > /dev/null 2>&1
    echo 'checkinstall is installed'
fi
```

`sudo apt-get update` — downloads the package lists from the repositories and updates them to get information on the newest versions of packages.

`sudo apt-get install --assume-yes <package name>` — It installs the specified package with the option `--assume-yes` to answer all prompts and run non-interactively.

`sleep 1` — Pausing the script before proceeding to the next line.

`/dev/null 2>&1` — redirects both standard output and standard error to `/dev/null`, effectively discarding any output or error messages and keeping the terminal output clean.



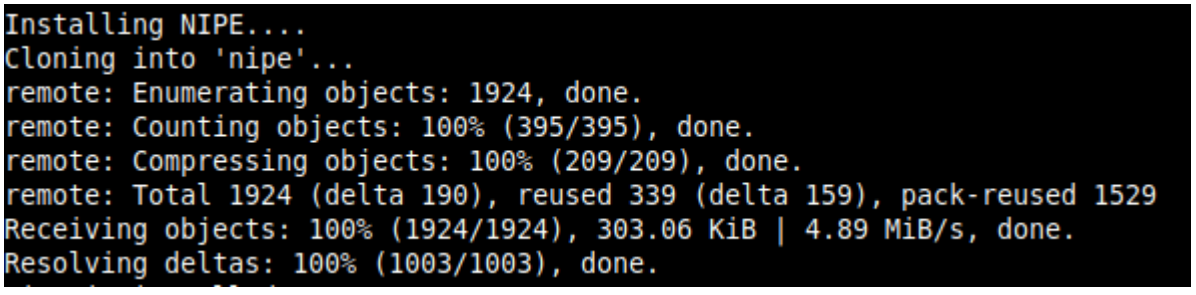
Installation of NIPE using IF-THEN

```
if [[ $NIPE == *"Status: install ok installed"* ]]
then
    echo '[#] Nipe is already installed.'
else
    echo 'NIPE is not install'
    sleep 1
    echo 'Installing NIPE....'
    if [[ $Cpanminus == *"Status: install ok installed"* ]]
    then
    echo
    else
        sudo apt-get update
        sleep 1
```

```
        sudo apt-get -y install cpanminus
    fi
    sleep 1
    git clone https://github.com/htrgouvea/nipe
    sleep 1
    cd "$NIPE_DIR" || { echo "Failed to navigate to Nipe di
rectory."; exit 1; }
    cpanm --installdeps .
    sleep 1
    sudo cpan install Switch JSON LWP::UserAgent Config::Si
mple
    sleep 1
    sudo perl nipe.pl install
    cd $NIPE_DIR
    sudo checkinstall --install=no --pkgname=nipe --pkgvers
ion=1.0.0 --default perl nipe.pl install
    sleep 1
    sudo dpkg --install nipe_1.0.0-1_amd64.deb
    echo 'Nipe is installed'
fi
```

Most of the script's essential applications are standard binary packages *(.deb)*, which can be easily managed using Debian package management tools such as `apt` , `apt-get` and `dpkg`. Unlike the standard binary package, NIPE is an engine that is based on PERL and requires the user to install it manually.

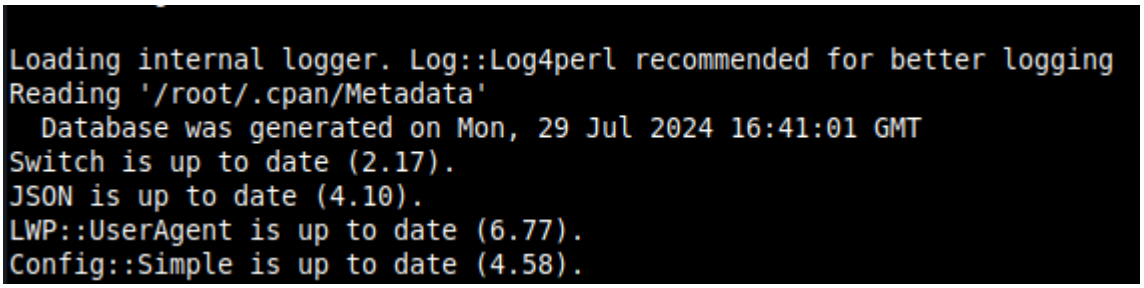
`Sudo git clone https://github.com/htrgouvea/nipe` — This command download and clone NIPE from the domain.



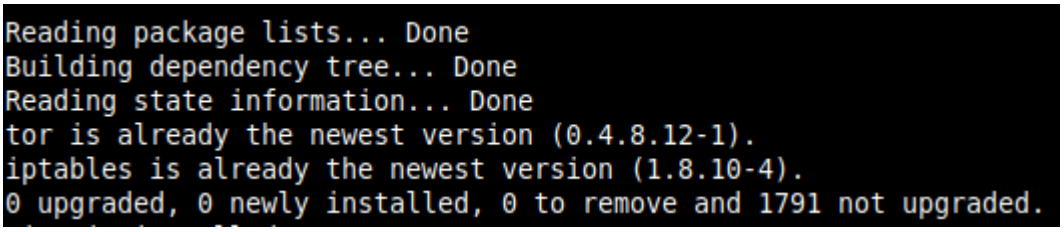
`cpanm --installdeps .` — The `cpanminus` command installs and unpack all dependencies to the current working directory.

```
--> Working on .
Configuring /home/kali/nipe ... OK
==> Found dependencies: Test::MockObject, Test::MockModule
--> Working on Test::MockObject
Fetching http://www.cpan.org/authors/id/C/CH/CHROMATIC/Test-MockObject-1.20200122.tar.gz ... OK
Configuring Test-MockObject-1.20200122 ... OK
==> Found dependencies: UNIVERSAL::can, Test::Warn, Test::Exception, UNIVERSAL::isa
--> Working on UNIVERSAL::can
Fetching http://www.cpan.org/authors/id/C/CH/CHROMATIC/UNIVERSAL-can-1.20140328.tar.gz ... OK
Configuring UNIVERSAL-can-1.20140328 ... OK
Building and testing UNIVERSAL-can-1.20140328 ... OK
Successfully installed UNIVERSAL-can-1.20140328
--> Working on Test::Warn
Fetching http://www.cpan.org/authors/id/B/BI/BIGJ/Test-Warn-0.37.tar.gz ... OK
Configuring Test-Warn-0.37 ... OK
==> Found dependencies: Sub::Uplevel
--> Working on Sub::Uplevel
Fetching http://www.cpan.org/authors/id/D/DA/DAGOLDEN/Sub-Uplevel-0.2800.tar.gz ... OK
Configuring Sub-Uplevel-0.2800 ... OK
Building and testing Sub-Uplevel-0.2800 ... OK
Successfully installed Sub-Uplevel-0.2800
Building and testing Test-Warn-0.37 ... OK
Successfully installed Test-Warn-0.37
--> Working on Test::Exception
Fetching http://www.cpan.org/authors/id/E/EX/EXODIST/Test-Exception-0.43.tar.gz ... OK
Configuring Test-Exception-0.43 ... OK
Building and testing Test-Exception-0.43 ... OK
Successfully installed Test-Exception-0.43
--> Working on UNIVERSAL::isa
Fetching http://www.cpan.org/authors/id/E/ET/ETHER/UNIVERSAL-isa-1.20171012.tar.gz ... OK
Configuring UNIVERSAL-isa-1.20171012 ... OK
Building and testing UNIVERSAL-isa-1.20171012 ... OK
Successfully installed UNIVERSAL-isa-1.20171012
Building and testing Test-MockObject-1.20200122 ... OK
Successfully installed Test-MockObject-1.20200122
--> Working on Test::MockModule
Fetching http://www.cpan.org/authors/id/G/GF/GFRANKS/Test-MockModule-v0.178.0.tar.gz ... OK
Configuring Test-MockModule-v0.178.0 ... OK
==> Found dependencies: Test::Warnings, SUPER
--> Working on Test::Warnings
Fetching http://www.cpan.org/authors/id/E/ET/ETHER/Test-Warnings-0.033.tar.gz ... OK
Configuring Test-Warnings-0.033 ... OK
Building and testing Test-Warnings-0.033 ... OK
Successfully installed Test-Warnings-0.033
--> Working on SUPER
Fetching http://www.cpan.org/authors/id/C/CH/CHROMATIC/SUPER-1.20190531.tar.gz ... OK
Configuring SUPER-1.20190531 ... OK
Building and testing SUPER-1.20190531 ... OK
Successfully installed SUPER-1.20190531
Building and testing Test-MockModule-v0.178.0 ... OK
Successfully installed Test-MockModule-v0.178.0
<== Installed dependencies for .. Finishing.
9 distributions installed
```

`sudo cpan install Switch JSON LWP::UserAgent Config::Simple` — the usage of cpan command is to interact with the perl modules for installation.



`sudo perl nipe.pl install` — this command installs nipe



Using Checkinstall to create Debian package for Nipe and installing Nipe Debian package with dpkg.

Since `nipe.pl` runs on PERL script and for it to work efficiently. The scripts require a `checkinstall` tool to manage `nipe.pl` dependencies and environment correctly. `checkinstall` ensures that all required dependencies for the software are correctly installed.

```
sudo checkinstall --install=no --pkgname=nipe --pkgversion=
1.0.0 --default perl nipe.pl install
```

`--install=no` — This option tells `checkinstall` not to install the generated package immediately. It only creates the package without installing it, allowing the user to review or use it later.

`--pkgname=nipe` — This specifies the package name.

`--pkgversion=1.0.0` — This specifies the version number

`--default` — default answer to the installation of the package.

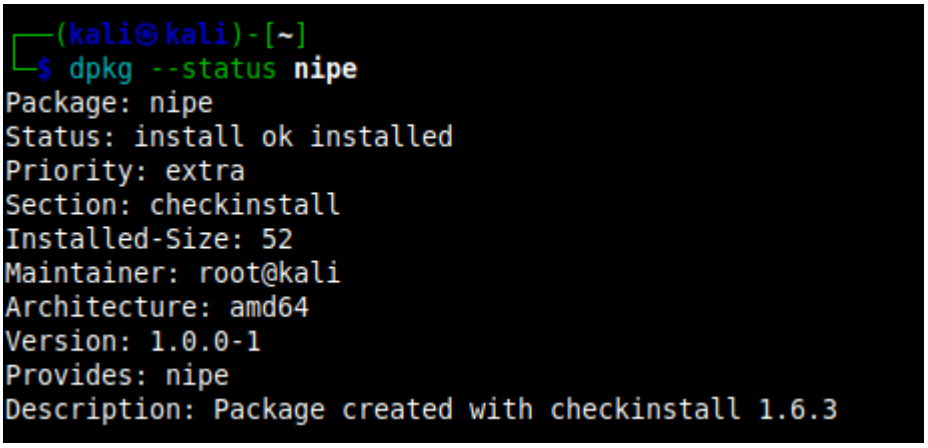
`perl nipe.pl install` — This command specifies the installation command that would normally be run to install Nipe. `checkinstall` uses this command to determine the files to be installed and convert them to the Debian package.

```
*****************************************
**** Debian package creation selected ***
*****************************************

This package will be built according to these values:

0 -  Maintainer: [ root@kali ]
1 -  Summary: [ Package created with checkinstall 1.6.3 ]
2 -  Name:    [ nipe ]
3 -  Version: [ 1.0.0 ]
4 -  Release: [ 1 ]
5 -  License: [ GPL ]
6 -  Group:   [ checkinstall ]
7 -  Architecture: [ amd64 ]
8 -  Source location: [ nipe ]
9 -  Alternate source location: [  ]
10 - Requires: [  ]
11 - Recommends: [  ]
12 - Suggests: [  ]
13 - Provides: [ nipe ]
14 - Conflicts: [  ]
15 - Replaces: [  ]
16 - Prerequires: [  ]

Enter a number to change any of them or press ENTER to continue:

Installing with perl nipe.pl install...
```

By using `checkinstall` , it can create a Debian package for nipe so that it can be managed by the system's package manager like `dpkg` to allow for easy installation, upgrade and removal.

Once the package has been created by `checkinstall`, the user can use `dpkg` to install it to the user's system. This prevents the system from being cluttered with files from manually installed software, which can be challenging to track or remove if needed.

```
sudo dpkg --install nipe_1.0.0-1_amd64.deb
```



Verifying NIPE status and navigating to the NIPE directory

```
#NIPE Directory
NIPE_DIR="/home/kali/nipe"
```

```
#Navigitating and powering NIPE
cd "$NIPE_DIR" || { echo "Failed to navigate to Nipe direct
ory."; exit 1; }
sudo perl nipe.pl start
sleep 2
status=$(sudo perl nipe.pl status)

#checking if the user is connected or not.
if [[ $status == *"Status: true"* ]]
then
    echo '[*] You are anonymous.. Connecting to the remote
Sever.'
else
    echo '**** you are not connected anonymously. Goodbye *
***'
    exit
fi
echo
```

`NIPE_DIR="/home/kali/nipe"` — This is the file path to where NIPE is installed in the system.

`sudo perl nipe.pl start` — To initiate NIPE connection

`if [[ $status == *" status: true"* ]]` — The status must be true to connect successfully with NIPE

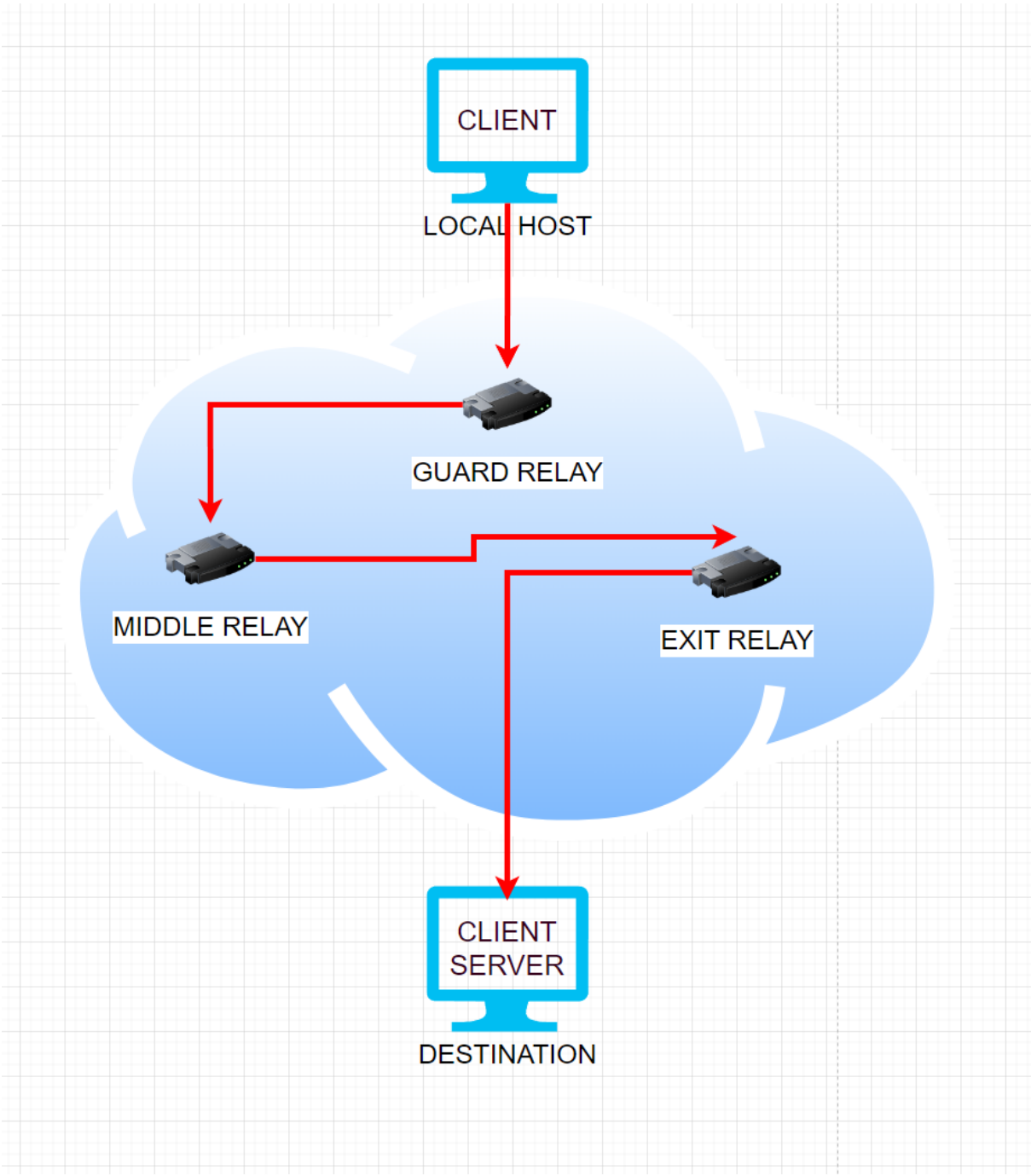
```
┌──(kali㉿kali)-[~/nipe]
└─$ sudo bash /home/kali/Desktop/CFC/ProjectNR/ProjectNR.sh
[#] Geoip-bin is already installed.
[#] Tor is already installed.
[#] sshpass is already installed
[#] checkinstall is already installed
[#] Nipe is already installed.
[*] You are anonymous.. Connecting to the remote Sever.

[*] Your Spoofed IP address is: 185.220.101.6 , Spoofed country: Germany
[?] Specify a Domain/IP address to scan: ^C

┌──(kali㉿kali)-[~/nipe]
└─$ sudo perl nipe.pl status

[+] Status: true
[+] Ip: 185.220.101.6
```

If the connection is otherwise, the script will prompt the user **** you are not connected anonymously. Goodbye ****

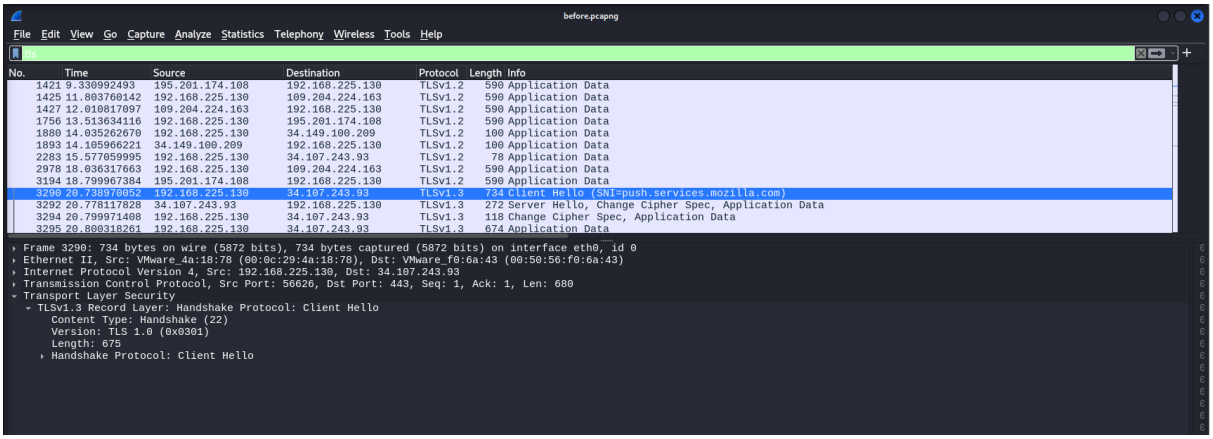
```
┌──(kali㉿kali)-[~/nipe]
└─$ sudo bash /home/kali/Desktop/CFC/ProjectNR/ProjectNR.sh
[#] Geoip-bin is already installed.
[#] Tor is already installed.
[#] sshpass is already installed
[#] checkinstall is already installed
[#] Nipe is already installed.
**** you are not connected anonymously. Goodbye ****
```

3.2 Using NIPE to spoof the user IP address to use tools like NMAP and WHOIS

NIPE is a Perl-based script that routes computer network traffic through the **TOR** network. Any applications in the script that connects to the internet will have it's traffic routed through **TOR**.



Using WIRESHARK to analyse the TLS handshake



TOR initiates the TLS handshake from the client to **TOR** relay nodes. The connection process begins with the client initiating a TLS handshake with the first relay (Guard relay). This handshake is important because it establishes an encrypted communication between the client and the client server. *(CloudFlare, 2023).*

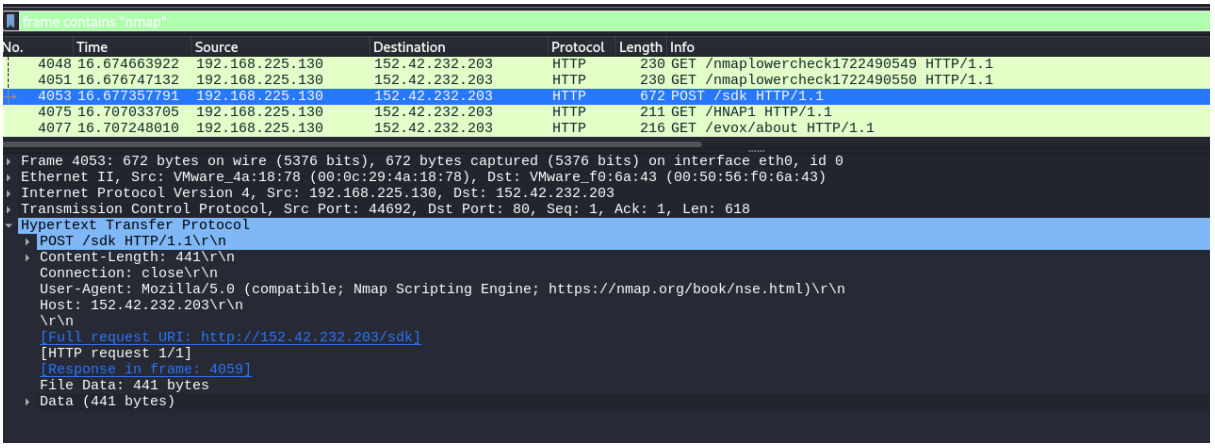
Spoof IP to gather information using tools like NMAP and WHOIS

NMAP is a networking scanning tool to discover hosts, services, versions and vulnerabilities within the scanned network. Using **NMAP** via **NIPE**, the scan might appear to be coming from the **TOR** exit node, masking the user's actual ip address effectively.

WHOIS is used to query databases that store registered users, such as a domain name or an IP address block. When **WHOIS** queries are performed through **TOR**, the originating IP address seen by the **WHOIS** server is that of the **TOR** exit node. This provides an additional layer of anonymity for users who wish to investigate domain information without exposing their IP address.

NMAP and **WHOIS** are tools for networking scanning and domain information gathering. When these tools are used in conjunction with **TOR**, the output reflects the IP address of the **TOR** exit node rather than the user's IP address.

NMAP scan activities are shown in Wireshark without **NIPE**



Limitation when scanning with NMAP via TOR

The limitation of using the **TOR** network can sometimes be slow due to the multiple layers of encryption and the routing through numerous relays. This can impact the speed of **NMAP** scans and inconsistent results from **NMAP.**

NMAP result using TOR

```
┌──(kali㉿kali)-[~/nipe]
└─$ sudo perl nipe.pl status

[+] Status: true
[+] Ip: 45.139.122.176


┌──(kali㉿kali)-[~/nipe]
└─$ sudo nmap -Pn -sV -v 152.42.232.203
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-01 01:31 EDT
NSE: Loaded 46 scripts for scanning.
Initiating Parallel DNS resolution of 1 host. at 01:31
Completed Parallel DNS resolution of 1 host. at 01:31, 0.68s elapsed
Initiating SYN Stealth Scan at 01:31
Scanning 152.42.232.203 [1000 ports]
SYN Stealth Scan Timing: About 15.05% done; ETC: 01:34 (0:02:55 remaining)
SYN Stealth Scan Timing: About 30.05% done; ETC: 01:34 (0:02:22 remaining)
Stats: 0:01:03 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 30.50% done; ETC: 01:34 (0:02:21 remaining)
SYN Stealth Scan Timing: About 46.00% done; ETC: 01:34 (0:01:48 remaining)
SYN Stealth Scan Timing: About 61.00% done; ETC: 01:34 (0:01:18 remaining)
SYN Stealth Scan Timing: About 76.00% done; ETC: 01:34 (0:00:48 remaining)
Completed SYN Stealth Scan at 01:34, 201.35s elapsed (1000 total ports)
Initiating Service scan at 01:34
NSE: Script scanning 152.42.232.203.
Initiating NSE at 01:34
Completed NSE at 01:34, 0.00s elapsed
Initiating NSE at 01:34
Completed NSE at 01:34, 0.00s elapsed
Nmap scan report for 152.42.232.203
Host is up.
All 1000 scanned ports on 152.42.232.203 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 202.40 seconds
           Raw packets sent: 2000 (88.000KB) | Rcvd: 0 (0B)
```

NMAP result without TOR

```
┌──(kali㉿kali)-[~/nipe]
└─$ sudo perl nipe.pl status

[+] Status: false
[+] Ip: 103.6.150.177


┌──(kali㉿kali)-[~/nipe]
└─$ sudo nmap -Pn -sV -v 152.42.232.203
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-01 01:35 EDT
NSE: Loaded 46 scripts for scanning.
Initiating Parallel DNS resolution of 1 host. at 01:35
Completed Parallel DNS resolution of 1 host. at 01:35, 0.43s elapsed
Initiating SYN Stealth Scan at 01:35
Scanning 152.42.232.203 [1000 ports]
Discovered open port 22/tcp on 152.42.232.203
Discovered open port 21/tcp on 152.42.232.203
Discovered open port 80/tcp on 152.42.232.203
Completed SYN Stealth Scan at 01:35, 9.95s elapsed (1000 total ports)
Initiating Service scan at 01:35
Scanning 3 services on 152.42.232.203
Completed Service scan at 01:35, 6.10s elapsed (3 services on 1 host)
NSE: Script scanning 152.42.232.203.
Initiating NSE at 01:35
Completed NSE at 01:35, 0.07s elapsed
Initiating NSE at 01:35
Completed NSE at 01:35, 0.06s elapsed
Nmap scan report for 152.42.232.203
Host is up (0.012s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.5
22/tcp open  ssh     OpenSSH 9.3p1 Ubuntu 1ubuntu3.6 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    Apache httpd 2.4.57 ((Ubuntu))
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.95 seconds
           Raw packets sent: 2003 (88.132KB) | Rcvd: 1317 (52.716KB)
```

3.3 Using Geoip-bin and the curl command for geolocation and data retrieval

```
IP=$(curl --silent ifconfig.io)
country=$(geoiplookup $IP | awk '{print $5}')
echo "[*] Your Spoofed IP address is: $IP , Spoofed countr
y: $country"
read -p "[?] Specify a Domain/IP address to scan: "Domain
```

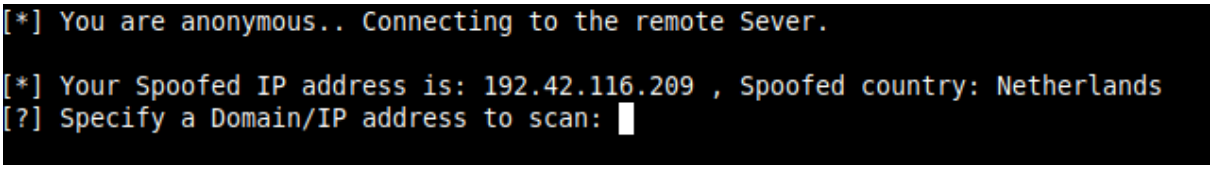
The script uses **Geoip-bin** and the `curl` command to facilitate IP geolocation information gathering.

`curl --silent ifconfig.io` uses `curl` to fetch the user's public IP address. The `--silent` flag suppresses the progress meter and error messages to ensure only

the IP address is captured and stored in the **IP** variable.

`geoiplookup $IP | awk '{print $5}'` uses **Geoip-bin** to determine the geographic location associated with the IP address stored in the **IP** variable. The `geoiplookup` command outputs various details, including the country, organisation and city. In this syntax, the output is piped into `awk '{print $5}'`, which extracts the fifth column corresponding to the country name. This country information is then stored in the **country** variable.

`read -p "[?] Specify a Domain/IP address to scan: "Domain` prompts the user to enter a domain or IP address to scan using **NMAP** and **WHOIS** further within the script.



3.4. Using SSH with SSHPASS to connect to a remote server

SSH (Secure Shell Protocol) is a method to secure a remote login from one device to another. It is an alternative option for stronger authentication, and it aims to protect communications and integrity with strong encryption. It is also an alternative secure way to the non-protected login protocols such as **Telnet** and insecure file transfer methods such as **FTP** *(Tatu Ylonen ,1996)*

```
read -p "[?] Specify a Domain/IP address to scan: " Domain

#Variables for connecting to Remote Server using SSHPASS
USER= "tc"
PASS= "tc"
REMOTE_SERVER="192.168.225.129"

sshpass -p $PASS ssh -T -o StrictHostKeyChecking=no $USER@
$REMOTE_SERVER <<EOF
nmap -Pn -sV $Domain -oN $OUTPUT
EOF
```

```
#Variables for the output
OUTPUT="Nmap_$Domain.txt"
```

The `sshpass` utility is designed to run **SSH** using the keyboard-interactive password authentication mode, but in a non-interactive way. `sshpass` runs **SSH** in a dedicated TTY, allowing **SSH** to think it is getting the password from an interactive user *(Amoany, Evans. 2023)*

`sshpass -p $PASS` — the `-p` flag is used when the password is given on the command line.

The login credentials are stored as variables `$USER` and `$PASS`. The IP address is stored in the `$REMOTE_SERVER` variable. This prevents any typing errors from the user if, in the future, the user decides to make amendments to the script, such as inputting more than one IP address or if the script is running on a different subnet.

Here document & /dev/null

In the script, EOF is used as a marker to specify the beginning and end of the ssh command.

`<<EOF` — this is the marker where it specifies the start

`EOF` — this marker tells the program to end its command.

`sshpass -p $PASS ssh -T -o StrictHostKeyChecking=no $USER@$REMOTE_SERVER <<EOF` — make the connection to the remote server. After a successful connection to the remote server, the script will run this command `nmap -Pn -sV $Domain -oN $OUTPUT` within the remote server. The end marker `EOF` will exit the connection from the remote server and will continue with the commands in the script.

`/dev/null 2>&1` redirects both standard output and standard error to `/dev/null`, discarding any output or error messages and keeping the terminal output clean.

Security concerns for the SSH command

The script utilise `sshpass` to interact with the remote server in a non-interactive way, essentially a hands-free for the user. This process is convenient for the

user. But however when `sshpass` is used in the command, it requires the password to be included in the command or a file that consists of the password. if an unauthorised user has access to the script, that user will able to view the password in the script or where the password list might stored in the system.

With `StrictHostKeyChecking=no`, the client accepts any host key without verification. This can be convenient for the user but introduces a security risk. This process might potentially expose the session to man-in-the-middle attacks.

3.5. FUNCTION with CASE statement

```
#Log file output path
LOGENTRIES="/var/log/nr.log"

#Function to log scan entries
function log_entry()
{
    local DOMAIN=$1
    local SCAN_TYPE=$2
    local TIMESTAMP=$(date '+%a %b %d %T %Z %Y')

    case "$SCAN_TYPE" in
        whois)
            echo "$TIMESTAMP - [*] Whois data collected fo
r: $DOMAIN" >> $LOGENTRIES
            ;;
        nmap)
            echo "$TIMESTAMP - [*] Nmap data collected for:
$DOMAIN" >> $LOGENTRIES
            ;;

    esac
}

#Variables for getting the spoofed IP address and country u
sing ifconfig.io and geoiplookup
```

```
sshpass -p $PASS ssh -T -o StrictHostKeyChecking=no $USER@
$REMOTE_SERVER <<EOF
nmap -Pn -sV $Domain -oN $OUTPUT
EOF

#Log the nmap scan
log_entry $Domain "nmap"
# Log the whois lookup
log_entry $Domain "whois"

#Variables for Whois command
Whois=$(whois $Domain | grep 'Address' | awk -F: '{print
$2}' | head -n 1 | tr -s ' ')
Whoiscountry=$(whois $Domain | grep 'Country' | awk -F: '{p
rint $2}' | tr -s ' ')
Whoisdata="/home/kali/whois_$Domain.txt"

#Execute the WHOIS command and save the output to the file
path
whois $Domain > "$Whoisdata"
```

The script utilises the FUNCTION and integrates the CASE statement to log the results of the **NMAP** and **WHOIS** to `/var/log/nr.log` this directory contains miscellaneous log files. Most logs must be written to this directory or an appropriate subdirectory *(refspecs.linuxfoundation.org, n.d.)* `HOMEDIRECTORY` is a function to be used in the script to navigate to the specify directory.

In the **WHOIS** variables, the command `grep` is utilised to filter **'Address'** lines, `awk -F: '{print $2}'` to remove the first column of **'Address'** and to only show the address output.

The function `log_entry` takes two arguments `DOMAIN` and `SCAN_TYPE`, the first argument represent the domain input, while the second argument specific the type of scan. The function also uses the `date` command to capture the timestamp as DAY:MONTH:DATE:TIME:TIMEZONE

The function uses a CASE statement to indicate the scan type either it is **NMAP** or **WHOIS**.

```
Wed Jul 24 08:36:35 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Wed Jul 24 08:43:19 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Wed Jul 24 08:43:21 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Wed Jul 24 23:56:54 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Wed Jul 24 23:56:59 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Thu Jul 25 00:49:33 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Thu Jul 25 00:49:38 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Sun Jul 28 06:46:57 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Sun Jul 28 06:47:03 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Sun Jul 28 06:54:54 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Sun Jul 28 06:55:12 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Mon Jul 29 05:29:22 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Mon Jul 29 05:29:28 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Mon Jul 29 09:27:12 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Mon Jul 29 09:27:38 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Tue Jul 30 05:20:20 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Tue Jul 30 05:20:31 EDT 2024 - [*] Whois data collected for: 152.42.232.203
Tue Jul 30 05:22:56 EDT 2024 - [*] Nmap data collected for: 152.42.232.203
Tue Jul 30 05:23:07 EDT 2024 - [*] Whois data collected for: 152.42.232.203
```

3.6. Using the FTP service to retrieve NMAP scan result

```
#Variables for connecting to Remote Server
USER="tc"
PASS="tc"
REMOTE_SERVER="192.168.225.129"

#Navigate to Home Directory to execute file transfer using
SCP to the current directory
HOMEDIRECTORY
ftp -n -V $REMOTE_SERVER <<EOF
user $USER $PASS
cd /home/tc
get $OUTPUT
exit
EOF
```

The script utilised the **FTP** service to retrieve the **NMAP** scan result from the remote server.

The `-n` flag disables **FTP** from attempting **“auto-login”**. If auto-login is enabled, **FTP** will check for **.netrc** file in the user’s home directory for an entry describing an account on the remote machine *(linux.die.net, n.d.)*. By default, **FTP** will attempt to automatically log in using the credentials from the **.netrc** file if available. For script automation's sake, this is to stop the **FTP** from prompting for login credentials, effectively allowing the script to run by itself without user intervention.

3.6.1 FTP Key Objective

The key objectives of FTP are: 1) to promote sharing of files (computer programs and/or data), 2) to encourage indirect or implicit (via programs) use of remote computers, 3) to shield a user from variations in file storage systems among hosts, and 4) to transfer data reliably and efficiently. *(Postel & Reynolds, 1985, p. 1)*

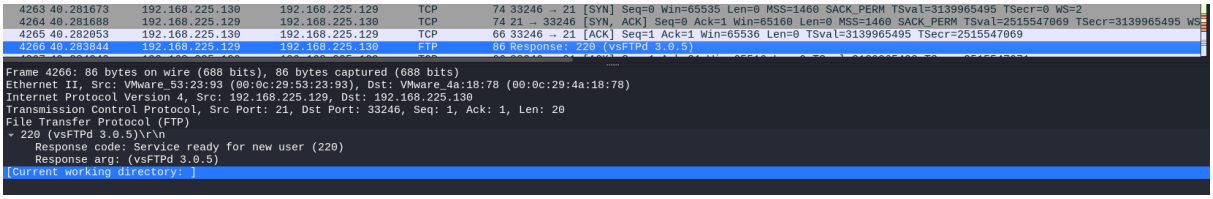
FTP was designed to transfer files via data connection from one computer to another by allowing access to the directories on the remote server/computers. It allows for data, text and software transfer between two different machines. The end user in the connection is known as **localhost**, and the server which provides the data is known as the **remote host** *(deepika92, 2021)*.

3.6.2 Communication Between the Client and Server

FTP connection consists of a command channel and a data channel. FTP commands and command responses go through the command channel, while the data or file transfers themselves pass through the data channel *(Villanueva, 2024)*.

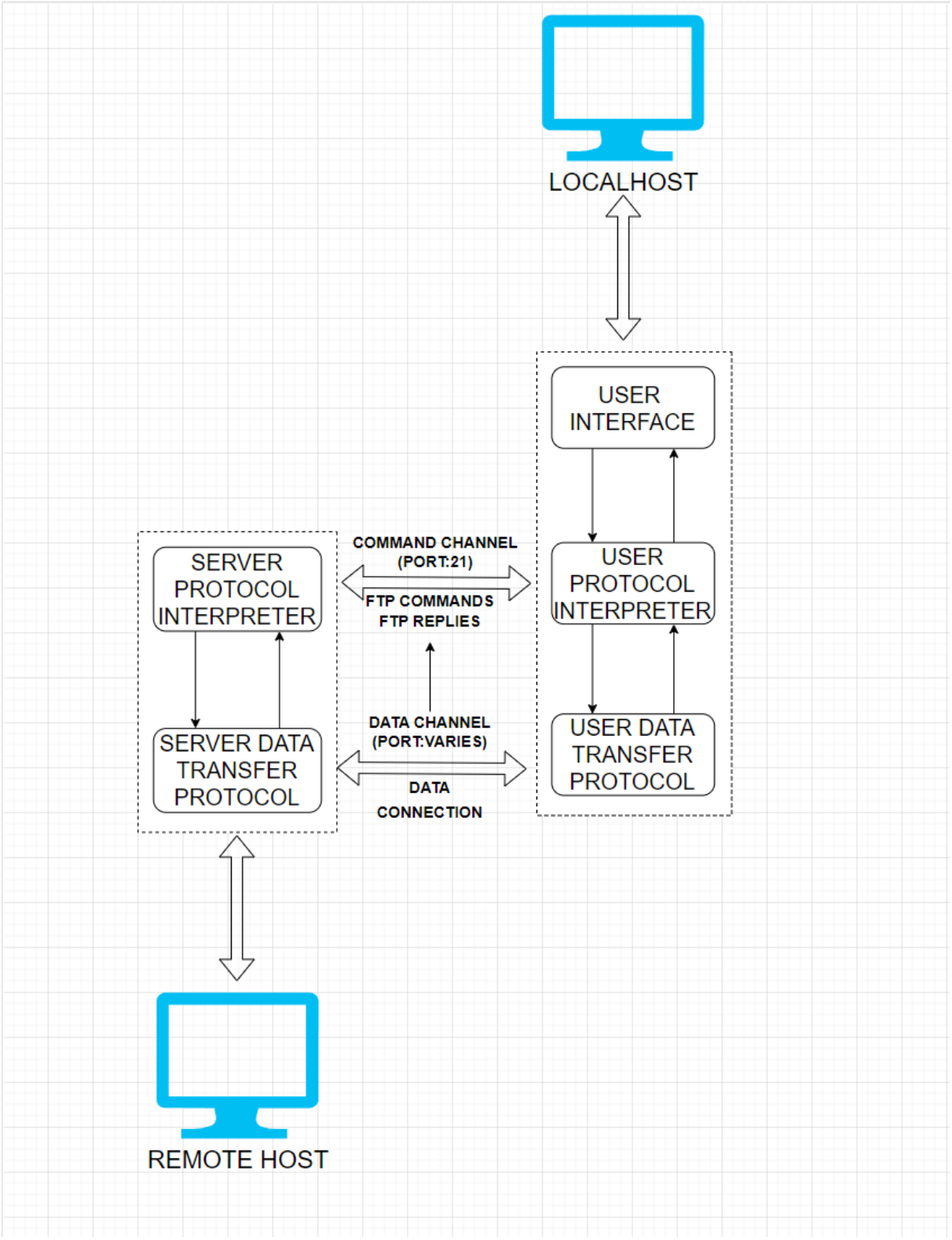
FTP uses two separate channels: the **Command channel** and the **Data channel**. The command channel sends commands and receives responses, while the data channel transfers files. The user protocol interpreter is known as ‘**USER PI**’. It handles communication and commands during the service. The **USER PI** sends commands and interprets the responses from the **SERVER PI**.

3.6.3 Establishing FTP connection



The FTP connection is established via TCP between the **USER PI** and **SERVER PI**. During this connection-establishing step, the **localhost** establishes a TCP handshake (SYN, SYN-ACK, ACK) to verify whether the **remote host** is up. By default, the FTP server listens on port **21** for incoming connections from FTP clients.

After verification, the **remote host** sends a response code (RESPONSE: 220) to the **localhost** indicating that it is ready for the FTP session.

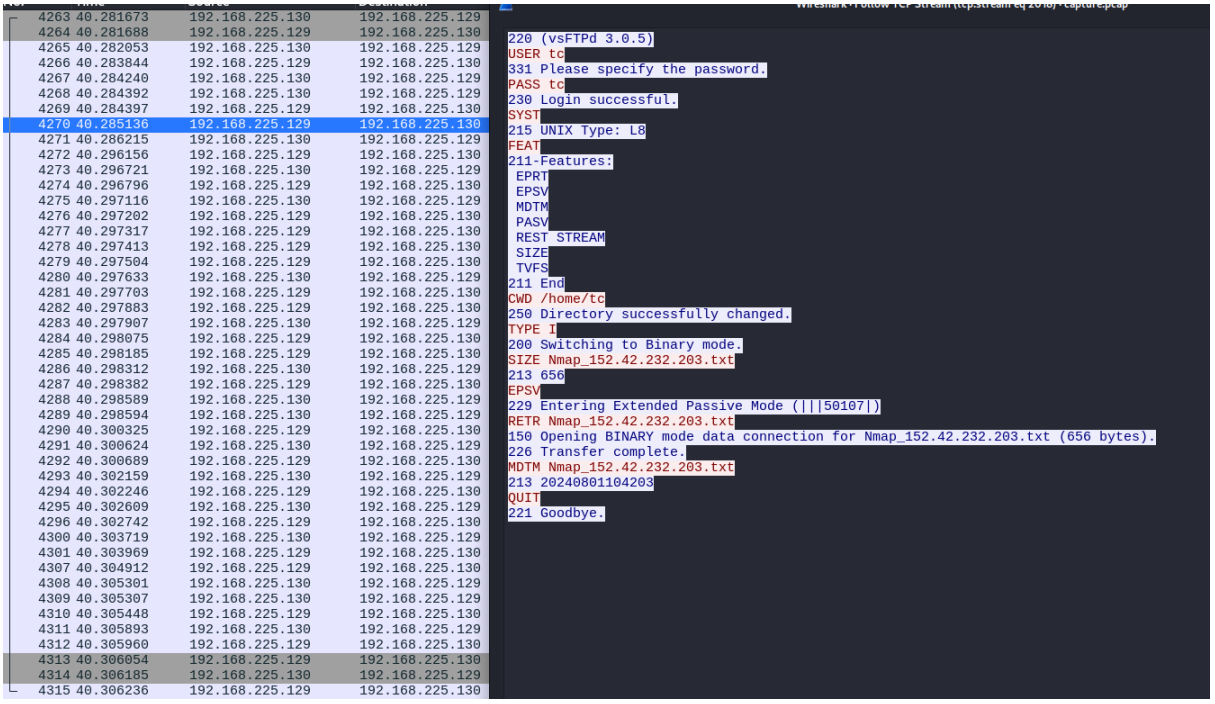


3.6.4 FTP Authentication Methods

FTP offers four different types of authentication:

1. **Anonymous FTP Authentication:** This method allows anyone to access the FTP server without providing any authentication credentials. It is commonly used for public file sharing, where anyone can upload and download files without providing any credentials. This method is not recommended for sensitive data transfer since there is no security layer.
2. **Basic FTP Authentication:** This method requires users to provide a username and password to access an FTP server. The username is transmitted in clear text, which makes it vulnerable to interception and eavesdropping. It is recommended to use this method with SSL/TLS encryption to enhance security.
3. **Digest FTP Authentication:** This method is an improvement over **Basic FTP Authentication** as it uses a hashed password instead of transmitting it in clear text. It provides better security than **Basic FTP Authentication**, but it is not widely supported by FTP clients and servers.
4. **SSH FTP Authentication**: This method uses **Secure Shell** to provide secure authentication and encryption for FTP connections. It is considered the most secure method of FTP authentication as it provides end-to-end encryption for data transfer. *(FasterCapital, n.d.)*

FTP authentication plays an essential role in safeguarding sensitive data from unauthorised access. Authentication is the process of making sure that only authorised users have access to sensitive data and providing an extra layer of security to ensure the verification of the user. Without this layer of authentication, an unauthorised user can exploit and manipulate the system for malicious purposes. *(FasterCapital, n.d.)*



3.6.5 Impact on CIA Triad

Confidentiality: When using FTP to transmits data, the login credentials is exposure in plaintext. This means that any data sent over FTP can be intercepted and read by an attacker, compromising confidentiality.

Integrity: FTP lacks mechanisms to verify the integrity of data during transfer. Without checksums or hashes, data could be tampered and potentially exploited by malicious users, compromising the integrity of the files.

Availability: FTP servers can be targeted by various attacks such as Denial of Service (DoS) attacks, it can disrupt the service availability. In addition, the lack of security configuration in the FTP server can lead to unauthorise access.

3.6.6 Secure Alternative Method: SCP

SCP (Secure Copy Protocol) is a secure alternative method to FTP for transferring files. It utilise **SSH** to ensure that data is encrypted during transmission. it provides a secure method for file transfer.

Confidentiality: SCP encrypts both command and data channels using **SSH**, ensuring that sensitive information cannot be intercepted

Integrity: SSH includes integrity checks to ensure that the data has not been tampered with during transmission

Availability: Secure authentication methods reduce the risk of unauthorised access.

3.6.7 FTP vs SCP Conclusion

In conclusion, the method on FTP to transfer files highlight potential vulnerabilities especially in regards to **CIA TRIAD**. By adopting the **SCP** method, these issues can be mitigated in providing a secure method for file transfer. **SCP** protects against interception and tampering.

4. Conclusion

As a cybersecurity practitioners, it is crucial to maintain the **CIA TRIAD** of data during transmission. This project aimed to address these aspects by developing an automated script to enhance user anonymity and security, it manages the installation and operation of essential applications such as Geoip-bin, TOR, sshpass, checkinstall and NIPE. By allowing the automation of these processes, the script ensures that the user can maintain a high level of privacy and security with minimal intervention.

Through the use of conditional statements in the script, it can handles various scenario making it more robust and adaptable for further improvement.

With the evaluation of FTP usage for file transfer. This report highlights the potential risk and vulnerabilities that undermine the **CIA TRIAD.** The exposure of login credentials in plain text during the FTP transmission compromising confidentiality, the lacking of data integrity checks allow for potential tampering from anuthorised user. FTP is also susceptible to threats such as **Denial of Service (DoS)**

To migitate this vulnerabilities, the report suggest the use of **SCP (Secure Copy Protocol)** as a secure alternative. SCP leverage on **SSH** to encrypt data during transmission to ensure that data remains confidential and intact. The use of secure authentication methods in **SCP** further enhances security by preventing unauthorised access. This transition from **FTP** to **SCP** is a crucial step in aligning the best practices in cybersecurity and safeguarding sensitivite information during file transfers.

The usage of NIPE in the script brings additional layer of security and anonymity, essential for protecting user identity and activities from a potential surveillance and traffic analysis. The limitation to this approach is a slower network due to multiple layers of relay and encryption.

Despite the script robustness in its handling for various scenario, the use of **sshpass** and disabling of **StrictHostKeyChecking** might potentially bring security risks. As a cybersecurity practitioners, we must prioritise the implementation of more secure method for handling SSH passwords and ensure that the host key verification is properly secured to mitigate the risks of potential attacks.

5. Recommendations

Based on the findings, several recommendations can be made to enhance the security and functionality of the script:

1. Implementation of secure methods for handling SSH passwords suchs as SSH Key-Based Authentication and avoiding disabling **StrictHostKeyChecking**
2. Explore other options to optimise NMAP scans when routing through TOR network to address potential inconsistent results.
3. Further development of the scripts error-handling mechanism to ensure its stability and realibility.

References

Ask Ubuntu. (n.d.). *Running ‘dpkg -s’ or ‘-l’ silent*. [online] Available at: https://askubuntu.com/questions/703160/running-dpkg-s-or-l-silent.

Stack Overflow. (n.d.). *How can I check if a package is installed and install it if not?* [online] Available at: https://stackoverflow.com/questions/1298066/how-can-i-check-if-a-package-is-installed-and-install-it-if-not.

What does 2>/dev/null mean (2013). *What does 2>/dev/null mean?* [online] Ask Ubuntu. Available at: https://askubuntu.com/questions/350208/what-does-2-dev-null-mean.

Amoany, E. (2020). *SSH password automation in Linux with sshpass*. [online] Enable Sysadmin. Available at: https://www.redhat.com/sysadmin/ssh-automation-sshpass.

OpenAI. (2024).
ChatGPT language model. Available at: https://chat.openai.com (Accessed: 24 July 2024).

wiki.debian.org. (n.d.). CheckInstall - Debian Wiki. [online] Available at: https://wiki.debian.org/CheckInstall [Accessed 28 Jul. 2024].

Tutorialspoint.com. (2020). Unix / Linux - Shell Input/Output Redirections - Tutorialspoint. [online] Available at: https://www.tutorialspoint.com/unix/unix-io-redirections.htm

tb-manual.torproject.org. (n.d.). ABOUT TOR BROWSER | Tor Project | Tor Browser Manual. [online] Available at: https://tb-manual.torproject.org/about/#:~:text=Tor is a network of.

wiki.debian.org. (n.d.). CheckInstall - Debian Wiki. [online] Available at: https://wiki.debian.org/CheckInstall.

Gouvêa, H. (2023). htrgouvea/nipe. [online] GitHub. Available at: https://github.com/htrgouvea/nipe.

Super User. (n.d.). What are the restrictions to ssh StrictHostKeyChecking=no? [online] Available at: https://superuser.com/questions/1751932/what-are-the-restrictions-to-ssh-stricthostkeychecking-no.

Wikipedia. (2024). Tor (network). [online] Available at: https://en.m.wikipedia.org/wiki/Tor_(network) [Accessed 29 Jul. 2024].

community.torproject.org. (n.d.). Tor Project | Types of relays on the Tor network. [online] Available at: https://community.torproject.org/relay/types-of-relays/.

Electronic Frontier Foundation. (2011). What is a Tor Relay? [online] Available at: https://www.eff.org/pages/what-tor-relay#:~:text=Tor relays are also referred.

Domain.com | Blog. (2020). What is WHOIS and How Is It Used? [online] Available at: https://www.domain.com/blog/what-is-whois-and-how-is-it-used/#:~:text=WHOIS is a public database.

Wikipedia Contributors (2019). Nmap. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Nmap.

Maurushat, Alana. “Ethical Hacking.” University of Ottawa Press eBooks, 2019, https://doi.org/10.1515/9780776627922.

Tatu Ylonen: SSH - Secure Login Connections over the Internet.
Proceedings of the 6th USENIX Security Symposium, pp. 37-42, USENIX, 1996.

DebianPackageManagement - Debian Wiki.
wiki.debian.org/DebianPackageManagement.

refspecs.linuxfoundation.org. (n.d.). 5.10. /var/log : Log files and directories. [online] Available at: https://refspecs.linuxfoundation.org/FHS_3.0/fhs/ch05s10.html [Accessed 1 Aug. 2024].

linux.die.net. (n.d.). ftp(1): Internet file transfer program - Linux man page. [online] Available at: https://linux.die.net/man/1/ftp.

deepika92 (2021). File Transfer Protocol (FTP). [online] GeeksforGeeks.

Available at:
https://www.geeksforgeeks.org/file-transfer-protocol-ftp/.

Villanueva, J.C. (2024). Active vs. Passive FTP Simplified: Understanding FTP Ports. [online] JSCAPE. Available at: https://www.jscape.com/blog/active-v-s-passive-ftp-simplified#:~:text=FTP command channel and data channel&text=Unless you configure it differently [Accessed 2 Aug. 2024].

FasterCapital. (n.d.). Types Of Ftp Authentication. [online] Available at: https://fastercapital.com/topics/types-of-ftp-authentication.html [Accessed 2 Aug. 2024].

Postel, J. and Reynolds, J. (1985). *rfc959*. [online] datatracker.ietf.org. Available at: https://datatracker.ietf.org/doc/html/rfc959.