Network Research Project

Ng Jing Ren

Table Of Content

Introduction

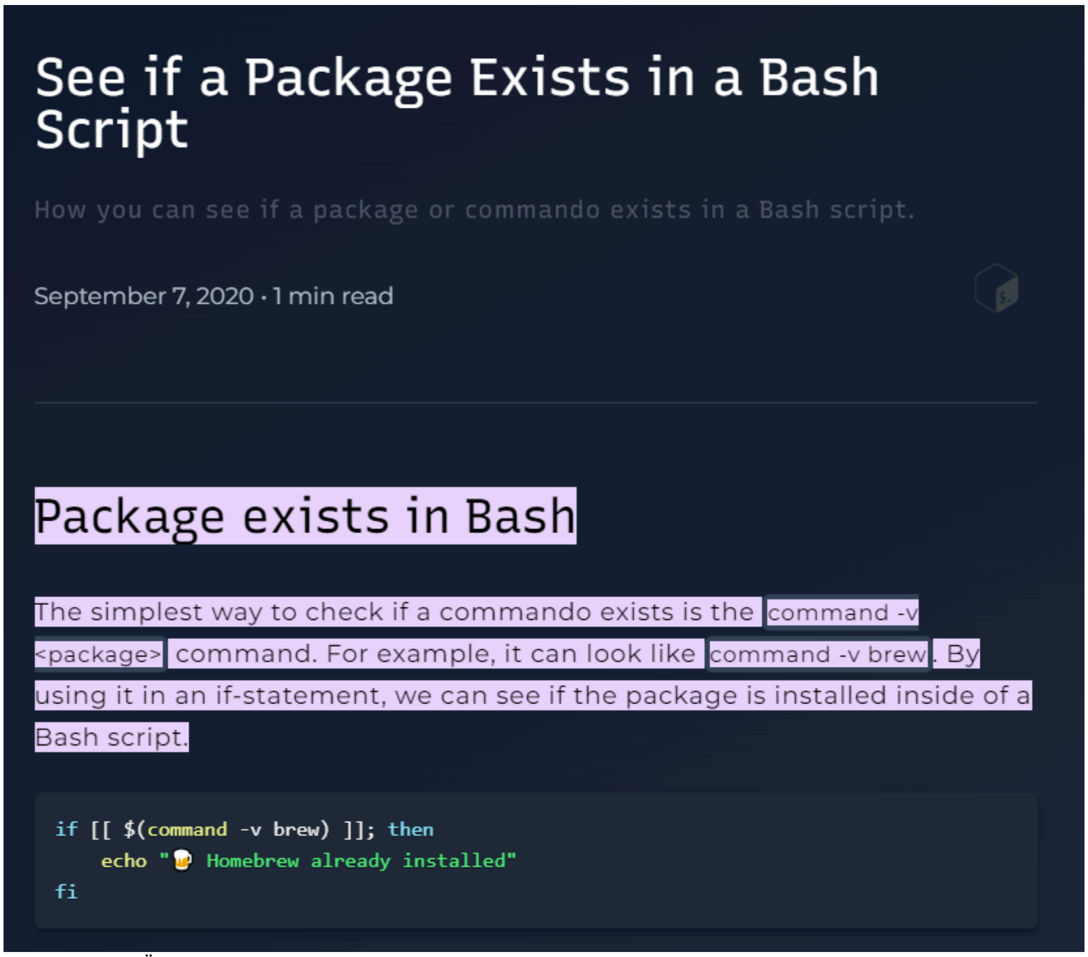
The introduction involves research for both Scope 1 and Scope 2 of the project. Research is crucial as it allows me to learn new things and execute the assigned tasks effectively.

For Scope 1, the research involves finding codes that will demonstrate how they will appear and be executed in Kali Linux.

For Scope 2, I have primarily researched websites to identify keywords that can summarize my protocol report. The websites researched for Scope 2 will be listed in the references at the end of the document.

Introduction

Reference from using installed application on bash script



See if a Package Exists in a Bash Script

How you can see if a package or commando exists in a Bash script.

September 7, 2020 · 1 min read

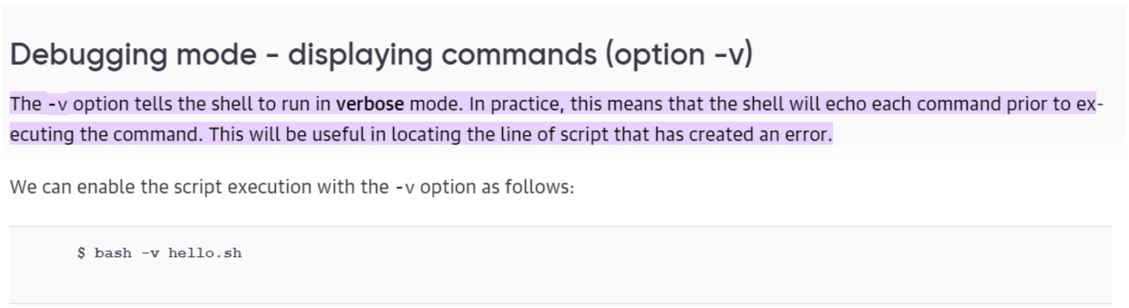
Package exists in Bash

The simplest way to check if a commando exists is the `command -v <package>` command. For example, it can look like `command -v brew`. By using it in an if-statement, we can see if the package is installed inside of a Bash script.

```
if [[ $(command -v brew) ]]; then
    echo "🍺 Homebrew already installed"
fi
```

Ödman, Anton. "See If a Package Exists in a Bash Script." *Banjocode*, 7 Sept. 2020, www.banjocode.com/post/bash/package-exists-bash. Accessed 7 Dec. 2023.

Explaination of using -v flag



Debugging mode - displaying commands (option -v)

The -v option tells the shell to run in **verbose** mode. In practice, this means that the shell will echo each command prior to ex-ecuting the command. This will be useful in locating the line of script that has created an error.

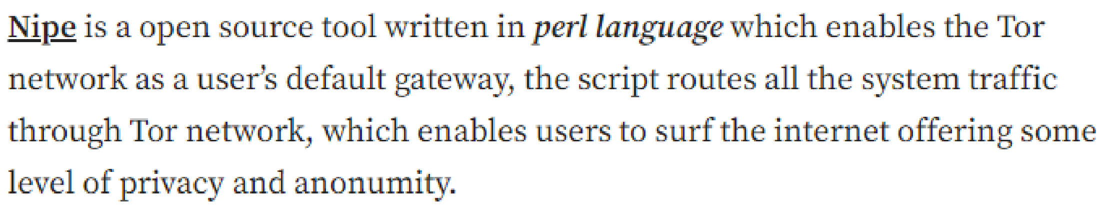
We can enable the script execution with the -v option as follows:

```
$ bash -v hello.sh
```

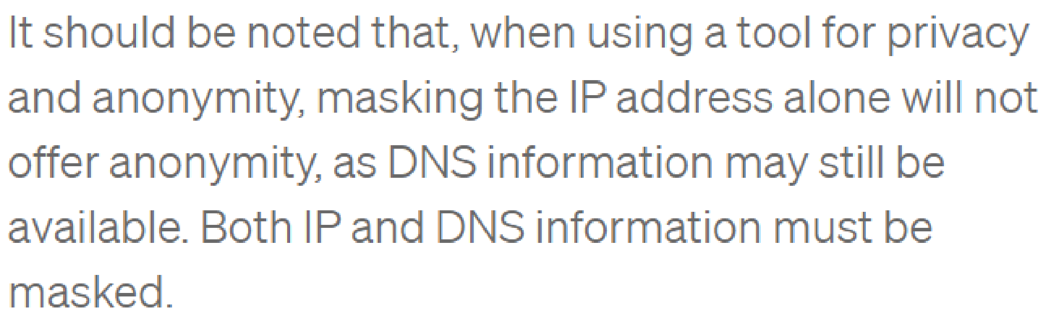
Naik, Ganesh. "Learning Linux Shell Scripting - Second Edition by Ganesh Naik." *O'Reilly Online Learning*, Packt Publishing, May 2018, www.oreilly.com/library/view/learning-linux-shell/9781788993197/0faa519d-98cd-4949-b2a7-454bbb87a204.xhtml#:~:text=The%20%2Dv%20option%20tells%20the,that%20has%20created%20an%20error. Accessed 7 Dec. 2023

Introduction

Anonymous Network reference



Nipe is a open source tool written in ***perl language*** which enables the Tor network as a user's default gateway, the script routes all the system traffic through Tor network, which enables users to surf the internet offering some level of privacy and anonumity.



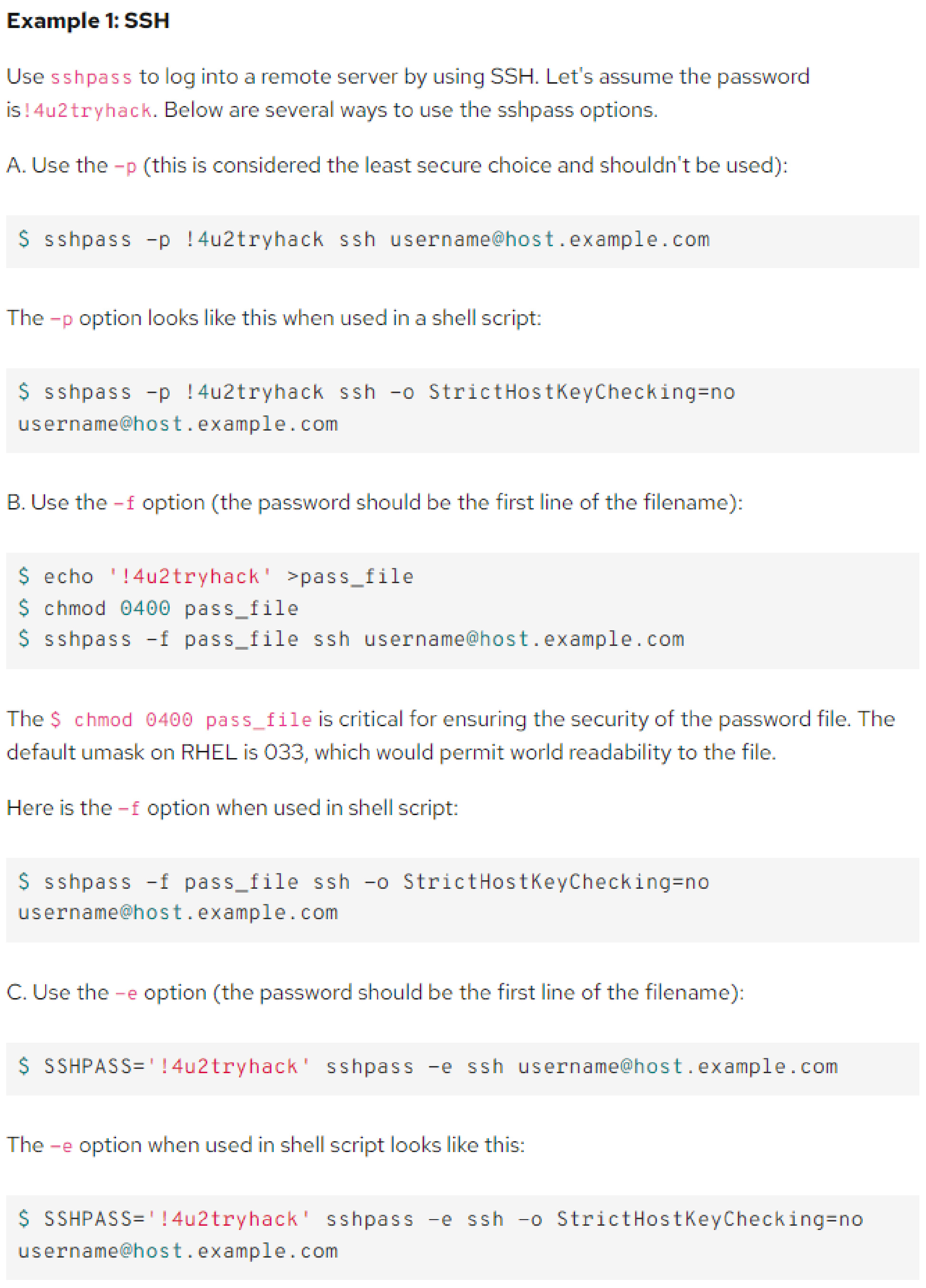
It should be noted that, when using a tool for privacy and anonymity, masking the IP address alone will not offer anonymity, as DNS information may still be available. Both IP and DNS information must be masked.

Santos, Cleber J. "[Pentest] Stay Anonymous Using NIPE." *Medium*, Medium, 28 Feb. 2021, cleberjsantos.medium.com/pentest-stay-anonymous-using-nipe-4d0a7c3bed1. Accessed 8 Dec. 2023

Nipe is an application that allows users to browse the internet anonymously. It acts as a Virtual Private Network (VPN) and utilizes The Onion Router (Tor), which encrypts and uses a multi-network run by volunteers to connect to the darknet.

Introduction

SSHpass reference on connecting to remote server



Example 1: SSH

Use `sshpass` to log into a remote server by using SSH. Let's assume the password is `!4u2tryhack`. Below are several ways to use the sshpass options.

A. Use the `-p` (this is considered the least secure choice and shouldn't be used):

```
$ sshpass -p !4u2tryhack ssh username@host.example.com
```

The `-p` option looks like this when used in a shell script:

```
$ sshpass -p !4u2tryhack ssh -o StrictHostKeyChecking=no
username@host.example.com
```

B. Use the `-f` option (the password should be the first line of the filename):

```
$ echo '!4u2tryhack' >pass_file
$ chmod 0400 pass_file
$ sshpass -f pass_file ssh username@host.example.com
```

The `$ chmod 0400 pass_file` is critical for ensuring the security of the password file. The default umask on RHEL is 033, which would permit world readability to the file.

Here is the `-f` option when used in shell script:

```
$ sshpass -f pass_file ssh -o StrictHostKeyChecking=no
username@host.example.com
```

C. Use the `-e` option (the password should be the first line of the filename):

```
$ SSHPASS='!4u2tryhack' sshpass -e ssh username@host.example.com
```

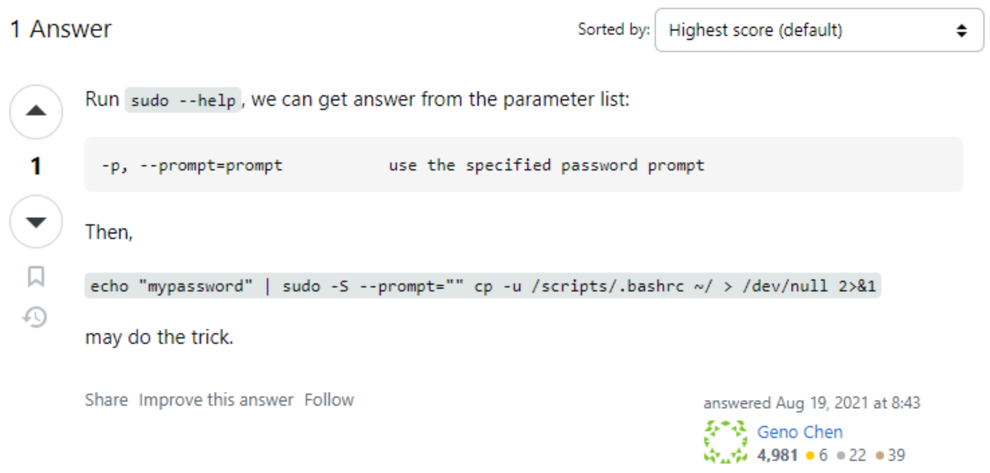
The `-e` option when used in shell script looks like this:

```
$ SSHPASS='!4u2tryhack' sshpass -e ssh -o StrictHostKeyChecking=no
username@host.example.com
```

Amoany, Evans. “SSH Password Automation in Linux with Sshpass.” *Enable Sysadmin*, Red Hat, Inc., 31 Aug. 2020, www.redhat.com/sysadmin/ssh-automation-sshpass. Accessed 7 Dec. 2023

Introduction

SSHpass reference on connecting to remote server


1 Answer

Sorted by: Highest score (default)

1

Run `sudo --help`, we can get answer from the parameter list:

```
-p, --prompt=prompt          use the specified password prompt
```

Then,

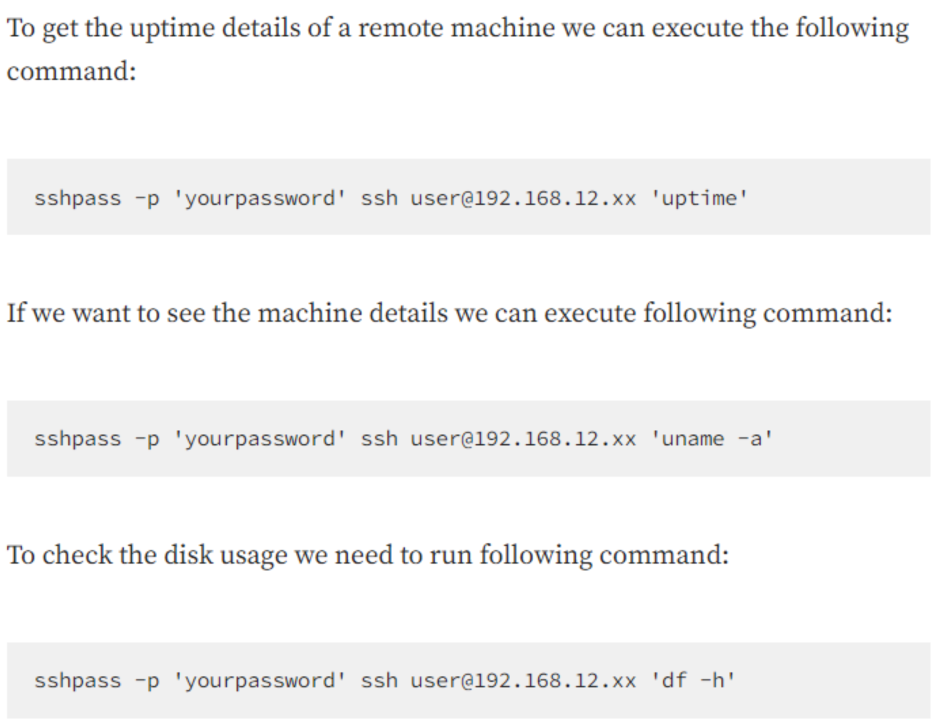
```
echo "mypassword" | sudo -S --prompt="" cp -u /scripts/.bashrc ~/ > /dev/null 2>&1
```

may do the trick.

Share Improve this answer Follow

answered Aug 19, 2021 at 8:43
Geno Chen
4,981 ●6 ●22 ●39

Geno, Chen. “Executing Sudo Command in Bash Script without Displaying It.” *Stack Overflow*, 19 Aug. 2021, stackoverflow.com/questions/68843636/executing-sudo-command-in-bash-script-without-displaying-it. Accessed on 9 Dec. 2023.


To get the uptime details of a remote machine we can execute the following command:

```
sshpass -p 'yourpassword' ssh user@192.168.12.xx 'uptime'
```

If we want to see the machine details we can execute following command:

```
sshpass -p 'yourpassword' ssh user@192.168.12.xx 'uname -a'
```

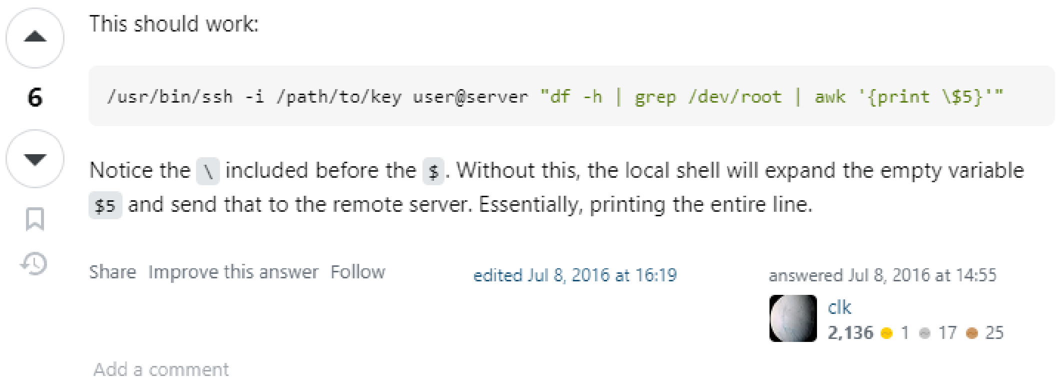
To check the disk usage we need to run following command:

```
sshpass -p 'yourpassword' ssh user@192.168.12.xx 'df -h'
```

Srivastava, Anurag. “Execute Commands on Remote Machines Using Sshpass.” *Medium*, Level Up Coding, 23 May 2020, levelup.gitconnected.com/execute-commands-on-remote-machines-using-sshpass-1f9bc4452e15. Accessed on 10 Dec. 2023.

Introduction

Awk reference on displaying information


6

This should work:

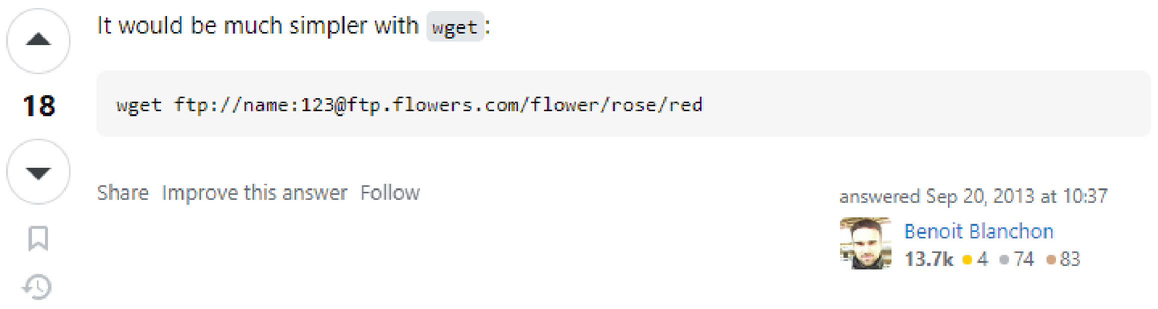
```
/usr/bin/ssh -i /path/to/key user@server "df -h | grep /dev/root | awk '{print \$5}'"
```

Notice the `\` included before the `$`. Without this, the local shell will expand the empty variable `$5` and send that to the remote server. Essentially, printing the entire line.

Share Improve this answer Follow
edited Jul 8, 2016 at 16:19
answered Jul 8, 2016 at 14:55
clk
2,136 ● 1 ● 17 ● 25
Add a comment

clk. "How to Print a Specific Column with AWK on a Remote SSH Session?" *Unix & Linux Stack Exchange*, 8 July 2016, unix.stackexchange.com/questions/294676/how-to-print-a-specific-column-with-awk-on-a-remote-ssh-session. Accessed 10 Dec. 2023.

Reference of downloading via FTP


18

It would be much simpler with `wget`:

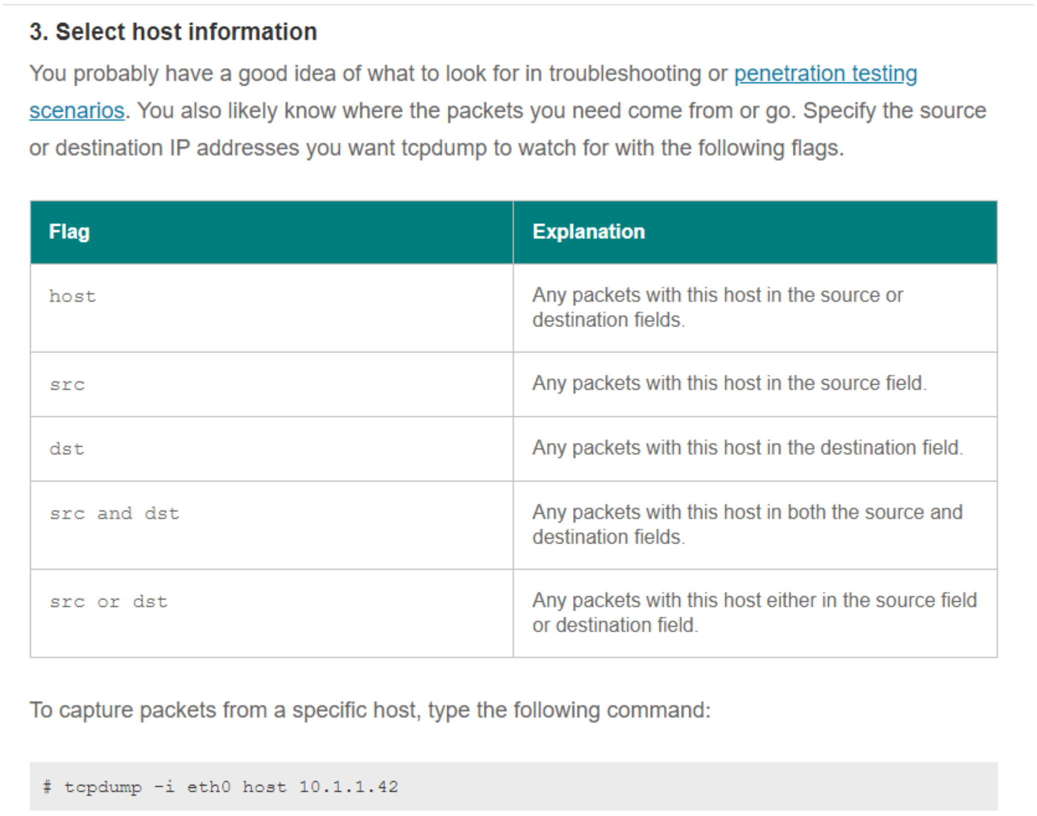
```
wget ftp://name:123@ftp.flowers.com/flower/rose/red
```

Share Improve this answer Follow
answered Sep 20, 2013 at 10:37
Benoit Blanchon
13.7k ● 4 ● 74 ● 83

Blanchon, Benoit. "Shell Script to Download Files from Remote Machine Using FTP." *Stack Overflow*, 20 Sept. 2013, stackoverflow.com/questions/18914451/shell-script-to-download-files-from-remote-machine-using-ftp. Accessed 11 Dec. 2023.

Introduction

Reference of executing command for tcpdump capture



3. Select host information

You probably have a good idea of what to look for in troubleshooting or penetration testing scenarios. You also likely know where the packets you need come from or go. Specify the source or destination IP addresses you want tcpdump to watch for with the following flags.

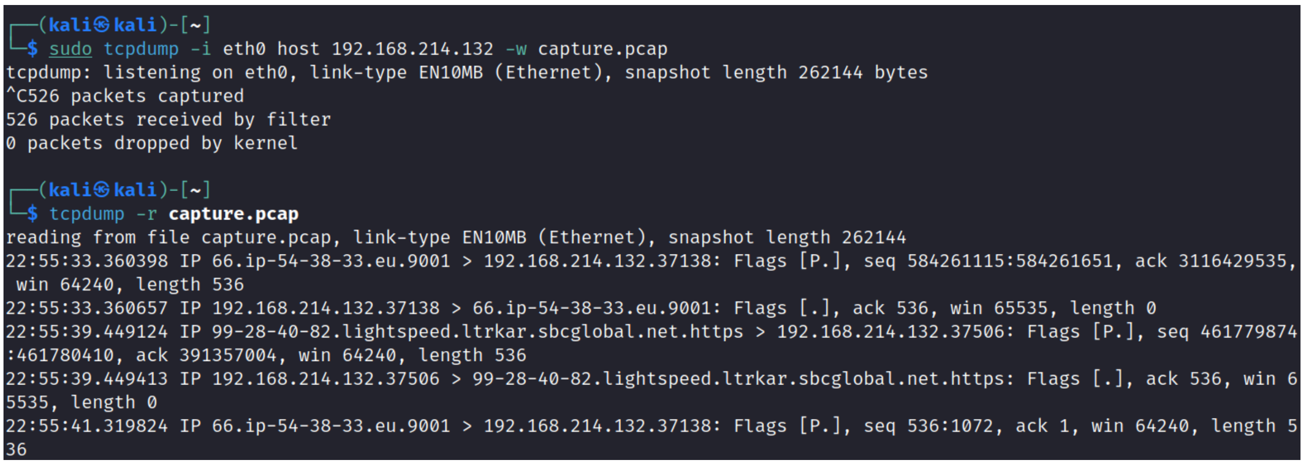
Flag	Explanation
`host`	Any packets with this host in the source or destination fields.
`src`	Any packets with this host in the source field.
`dst`	Any packets with this host in the destination field.
`src and dst`	Any packets with this host in both the source and destination fields.
`src or dst`	Any packets with this host either in the source field or destination field.

To capture packets from a specific host, type the following command:

```
# tcpdump -i eth0 host 10.1.1.42
```

Garn, Damon. “How to Capture and Analyze Traffic with Tcpdump: TechTarget.” *Networking*, TechTarget, 16 Aug. 2023, www.techtarget.com/searchnetworking/tutorial/How-to-capture-and-analyze-traffic-with-tcpdump. Accessed 12 Dec. 2023.

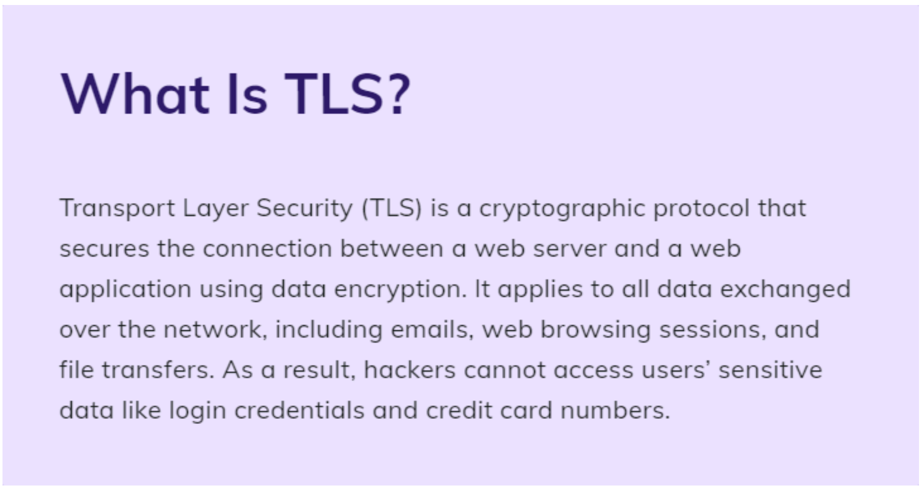
Example of executing the command in kali linux.



```
┌──(kali㉿kali)-[~]
└─$ sudo tcpdump -i eth0 host 192.168.214.132 -w capture.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C526 packets captured
526 packets received by filter
0 packets dropped by kernel

┌──(kali㉿kali)-[~]
└─$ tcpdump -r capture.pcap
reading from file capture.pcap, link-type EN10MB (Ethernet), snapshot length 262144
22:55:33.360398 IP 66.ip-54-38-33.eu.9001 > 192.168.214.132.37138: Flags [P.], seq 584261115:584261651, ack 3116429535,
 win 64240, length 536
22:55:33.360657 IP 192.168.214.132.37138 > 66.ip-54-38-33.eu.9001: Flags [.], ack 536, win 65535, length 0
22:55:39.449124 IP 99-28-40-82.lightspeed.ltrkar.sbcglobal.net.https > 192.168.214.132.37506: Flags [P.], seq 461779874
:461780410, ack 391357004, win 64240, length 536
22:55:39.449413 IP 192.168.214.132.37506 > 99-28-40-82.lightspeed.ltrkar.sbcglobal.net.https: Flags [.], ack 536, win 6
5535, length 0
22:55:41.319824 IP 66.ip-54-38-33.eu.9001 > 192.168.214.132.37138: Flags [P.], seq 536:1072, ack 1, win 64240, length 5
36
```

Introduction

Reference of TLS

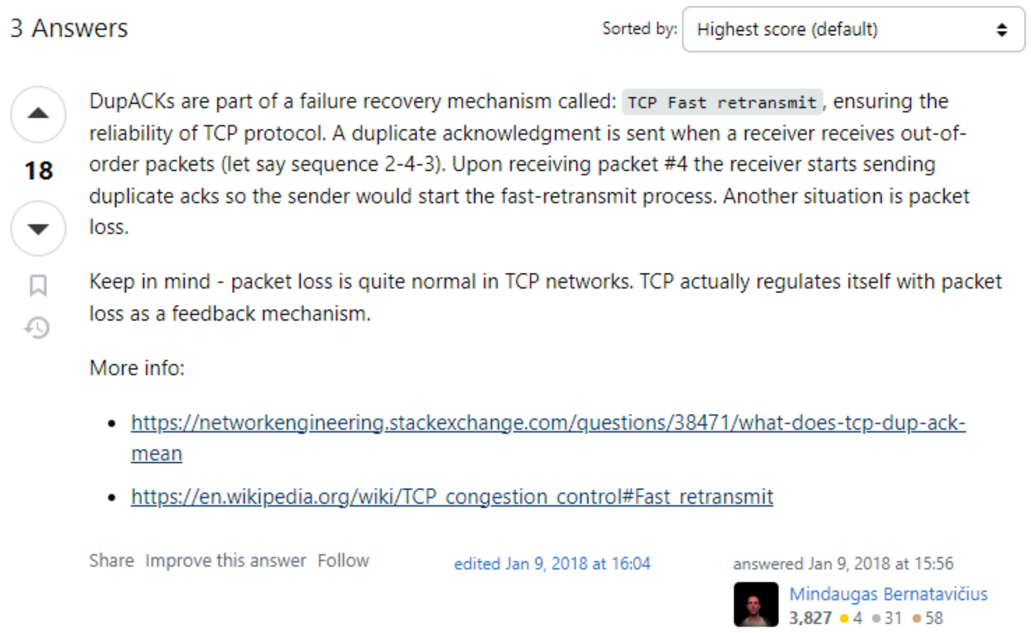


What Is TLS?

Transport Layer Security (TLS) is a cryptographic protocol that secures the connection between a web server and a web application using data encryption. It applies to all data exchanged over the network, including emails, web browsing sessions, and file transfers. As a result, hackers cannot access users' sensitive data like login credentials and credit card numbers.

M. Sopha. "What Is TLS? Understanding Transport Layer Security and How It Works." *Hostinger Tutorials*, 27 Sept. 2023, www.hostinger.com/tutorials/what-is-tls#What_Is_TLS. Accessed 12 Dec. 2023

Reference of duplicate ACK in wireshark



3 Answers

Sorted by: Highest score (default)

18

DupACKs are part of a failure recovery mechanism called: `TCP Fast retransmit`, ensuring the reliability of TCP protocol. A duplicate acknowledgment is sent when a receiver receives out-of-order packets (let say sequence 2-4-3). Upon receiving packet #4 the receiver starts sending duplicate acks so the sender would start the fast-retransmit process. Another situation is packet loss.

Keep in mind - packet loss is quite normal in TCP networks. TCP actually regulates itself with packet loss as a feedback mechanism.

More info:

- https://networkengineering.stackexchange.com/questions/38471/what-does-tcp-dup-ack-mean
- https://en.wikipedia.org/wiki/TCP_congestion_control#Fast_retransmit

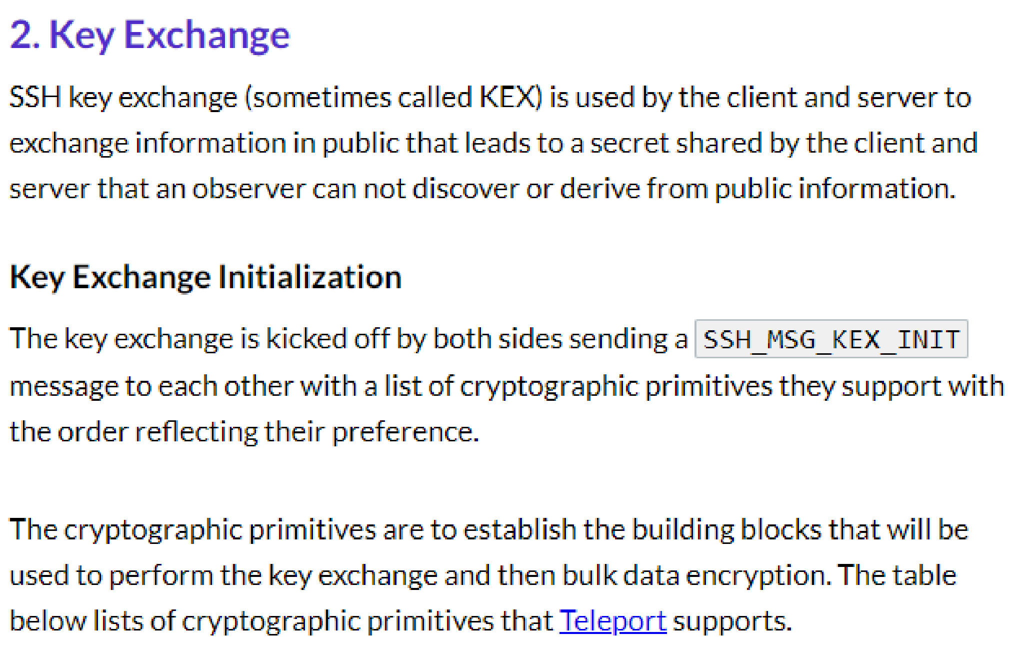
Share Improve this answer Follow edited Jan 9, 2018 at 16:04 answered Jan 9, 2018 at 15:56

Mindaugas Bernatavičius
3,827 ● 4 ● 31 ● 58

Bernatavičius, Mindaugas. "What Is Duplicate Ack When Does It Occur?" *Stack Overflow*, 9 Jan. 2018, stackoverflow.com/questions/48148820/what-is-duplicate-ack-when-does-it-occur. Accessed on 12 Dec. 2023.

Introduction

Reference of SSH key exchange



2. Key Exchange

SSH key exchange (sometimes called KEX) is used by the client and server to exchange information in public that leads to a secret shared by the client and server that an observer can not discover or derive from public information.

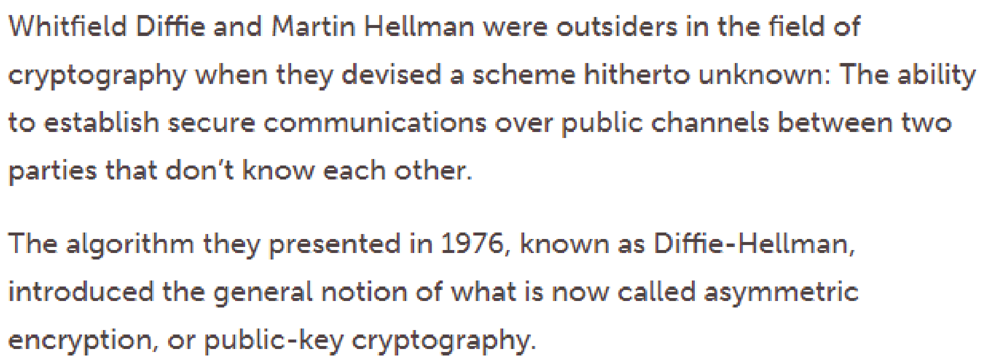
Key Exchange Initialization

The key exchange is kicked off by both sides sending a `SSH_MSG_KEX_INIT` message to each other with a list of cryptographic primitives they support with the order reflecting their preference.

The cryptographic primitives are to establish the building blocks that will be used to perform the key exchange and then bulk data encryption. The table below lists of cryptographic primitives that Teleport supports.

Jones, Russell. “SSH Handshake Explained.” *What Is SSH Handshake?*, 31 Mar. 2022, goteleport.com/blog/ssh-handshake-explained/. Accessed 12 Dec. 2023.

Reference of Diffie Hellman key exchange



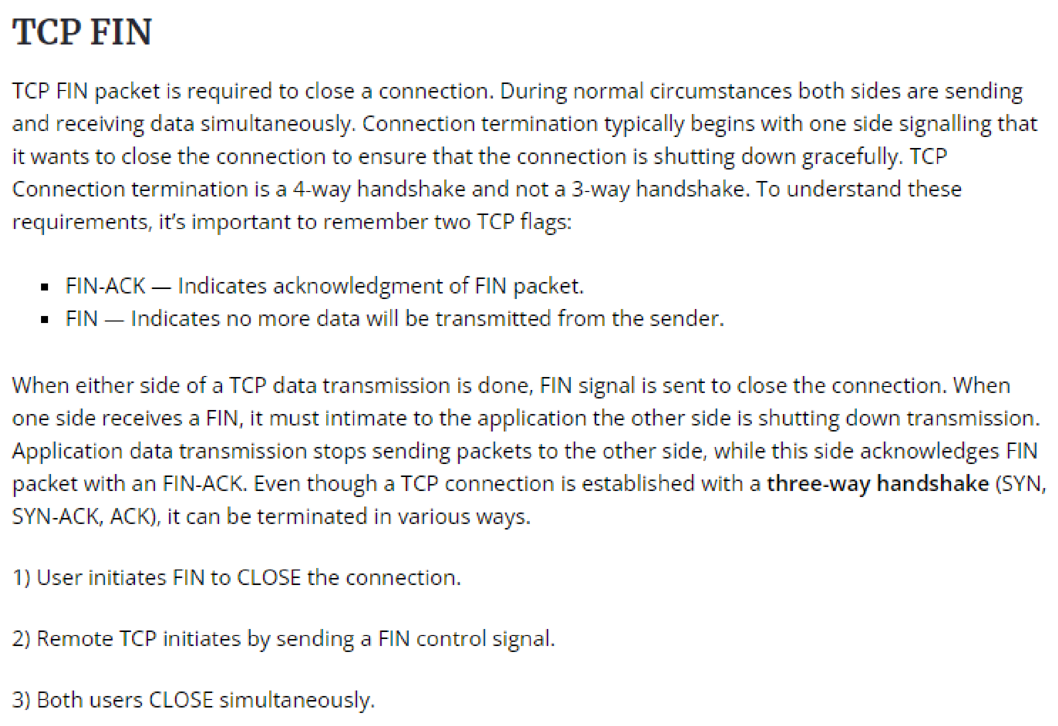
Whitfield Diffie and Martin Hellman were outsiders in the field of cryptography when they devised a scheme hitherto unknown: The ability to establish secure communications over public channels between two parties that don’t know each other.

The algorithm they presented in 1976, known as Diffie-Hellman, introduced the general notion of what is now called asymmetric encryption, or public-key cryptography.

Tyson, Matthew. “Understand Diffie-Hellman Key Exchange.” *InfoWorld*, InfoWorld, 20 Jan. 2022, www.infoworld.com/article/3647751/understand-diffie-hellman-key-exchange.html. Accessed 12 Dec. 2023.

Introduction

Reference of TCP FIN



TCP FIN

TCP FIN packet is required to close a connection. During normal circumstances both sides are sending and receiving data simultaneously. Connection termination typically begins with one side signalling that it wants to close the connection to ensure that the connection is shutting down gracefully. TCP Connection termination is a 4-way handshake and not a 3-way handshake. To understand these requirements, it's important to remember two TCP flags:

- FIN-ACK — Indicates acknowledgment of FIN packet.
- FIN — Indicates no more data will be transmitted from the sender.

When either side of a TCP data transmission is done, FIN signal is sent to close the connection. When one side receives a FIN, it must intimate to the application the other side is shutting down transmission. Application data transmission stops sending packets to the other side, while this side acknowledges FIN packet with an FIN-ACK. Even though a TCP connection is established with a **three-way handshake** (SYN, SYN-ACK, ACK), it can be terminated in various ways.

1) User initiates FIN to CLOSE the connection.

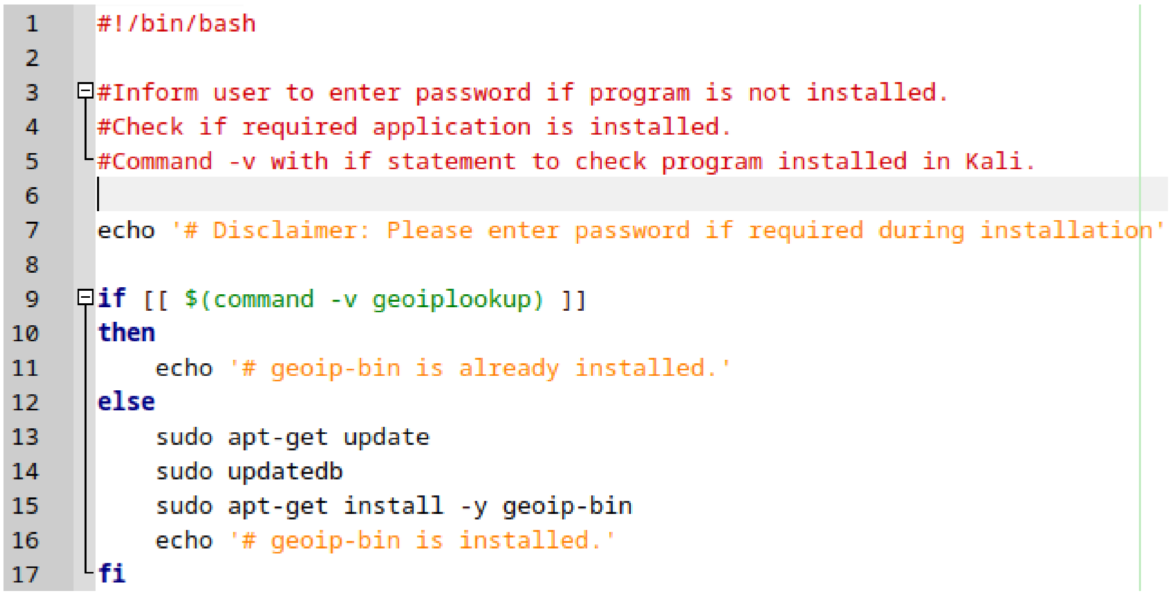
2) Remote TCP initiates by sending a FIN control signal.

3) Both users CLOSE simultaneously.

Bhardwaj, Rashmi. "What Is TCP Fin Packet?" *IP With Ease*, 17 Jan. 2022, ipwithease.com/what-is-tcp-fin-packet/. Accessed 12 Dec. 2023.

Methodologies - Installation Check

Example of detailed command in Kali Geany



```bash
#!/bin/bash

#Inform user to enter password if program is not installed.
#Check if required application is installed.
#Command -v with if statement to check program installed in Kali.

echo '# Disclaimer: Please enter password if required during installation'

if [[ $(command -v geoiplookup) ]]
then
    echo '# geoip-bin is already installed.'
else
    sudo apt-get update
    sudo updatedb
    sudo apt-get install -y geoip-bin
    echo '# geoip-bin is installed.'
fi
```

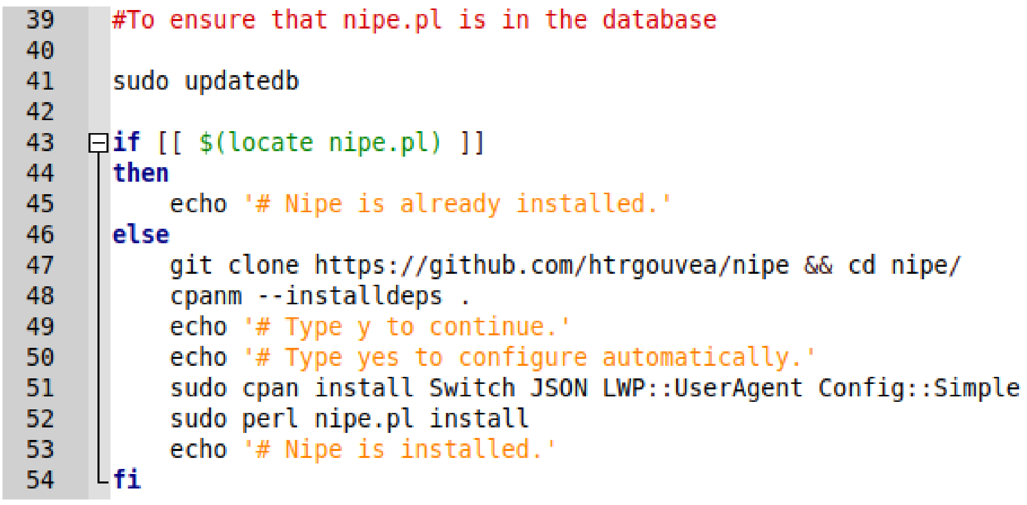
For the installation of the required application, I have opted to use the command -v because it shows the package or command that is inside the bash script itself. It also indicates whether the application is installed in the terminal.

For this example, we are using geoip-bin.

By using an if statement, the script can determine whether the application is already installed in the terminal. If geoip-bin is already installed, the script will echo '# geoip-bin is already installed'.

If the application is not installed, the script will prompt the user to enter their password to proceed with the installation. The message will say, 'Disclaimer: Please enter your password if required during installation.' The -y flag under sudo apt-get install allows the user to install the application immediately without being prompted for disk space usage alerts.

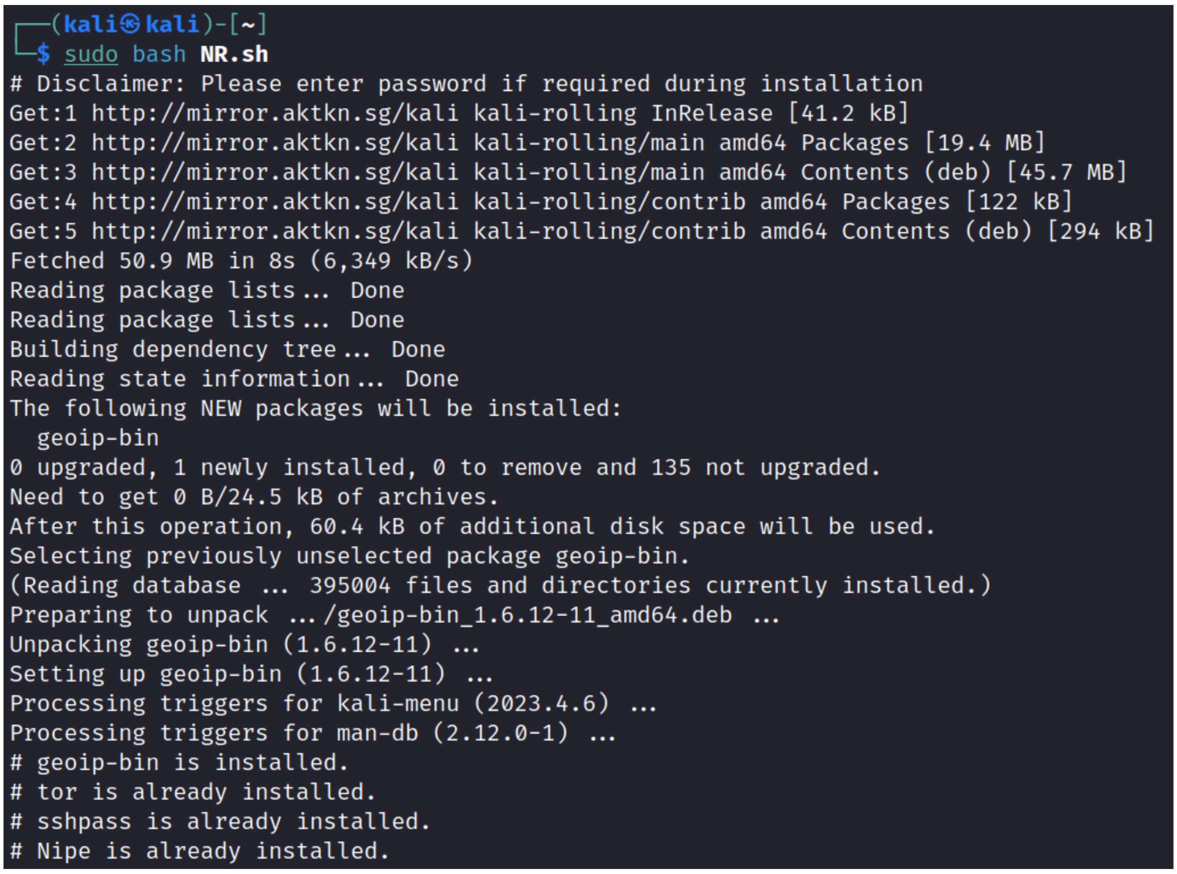
When installing nipe, I need to use sudo updatedb to ensure that the nipe tool is updated in the database. Subsequently, I use the locate command to find the file where it exists, instead of using command -v, because nipe's folder is not in /usr/bin.



```bash
#To ensure that nipe.pl is in the database

sudo updatedb

if [[ $(locate nipe.pl) ]]
then
    echo '# Nipe is already installed.'
else
    git clone https://github.com/htrgouvea/nipe && cd nipe/
    cpanm --installdeps .
    echo '# Type y to continue.'
    echo '# Type yes to configure automatically.'
    sudo cpan install Switch JSON LWP::UserAgent Config::Simple
    sudo perl nipe.pl install
    echo '# Nipe is installed.'
fi
```

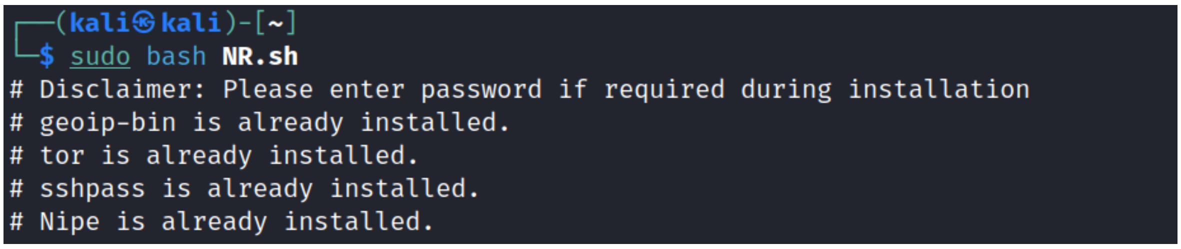
Methodologies - Installation Check

Example of how the script executes the installation



```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
Get:1 http://mirror.aktkn.sg/kali kali-rolling InRelease [41.2 kB]
Get:2 http://mirror.aktkn.sg/kali kali-rolling/main amd64 Packages [19.4 MB]
Get:3 http://mirror.aktkn.sg/kali kali-rolling/main amd64 Contents (deb) [45.7 MB]
Get:4 http://mirror.aktkn.sg/kali kali-rolling/contrib amd64 Packages [122 kB]
Get:5 http://mirror.aktkn.sg/kali kali-rolling/contrib amd64 Contents (deb) [294 kB]
Fetched 50.9 MB in 8s (6,349 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  geoip-bin
0 upgraded, 1 newly installed, 0 to remove and 135 not upgraded.
Need to get 0 B/24.5 kB of archives.
After this operation, 60.4 kB of additional disk space will be used.
Selecting previously unselected package geoip-bin.
(Reading database ... 395004 files and directories currently installed.)
Preparing to unpack .../geoip-bin_1.6.12-11_amd64.deb ...
Unpacking geoip-bin (1.6.12-11) ...
Setting up geoip-bin (1.6.12-11) ...
Processing triggers for kali-menu (2023.4.6) ...
Processing triggers for man-db (2.12.0-1) ...
# geoip-bin is installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
```

Example of a script installing geo-ip bin.

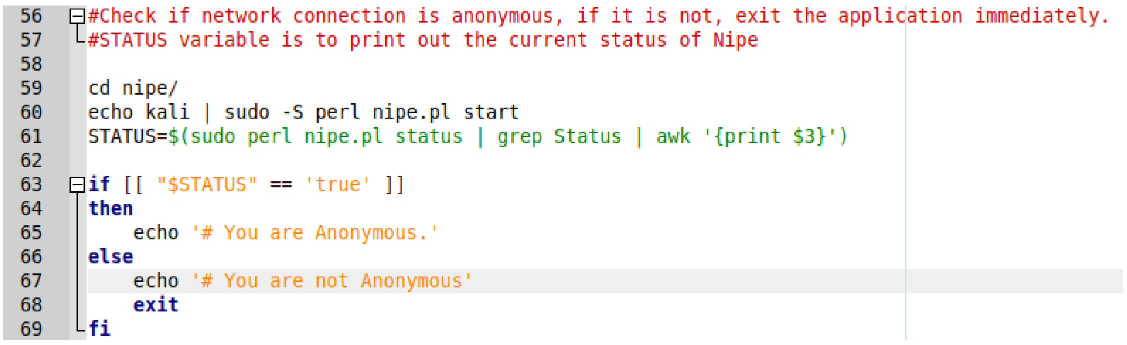


```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
```

After the script has installed every required application.

Methodologies - Checking if network is Anonymous

Example of detailed command in Kali Geany

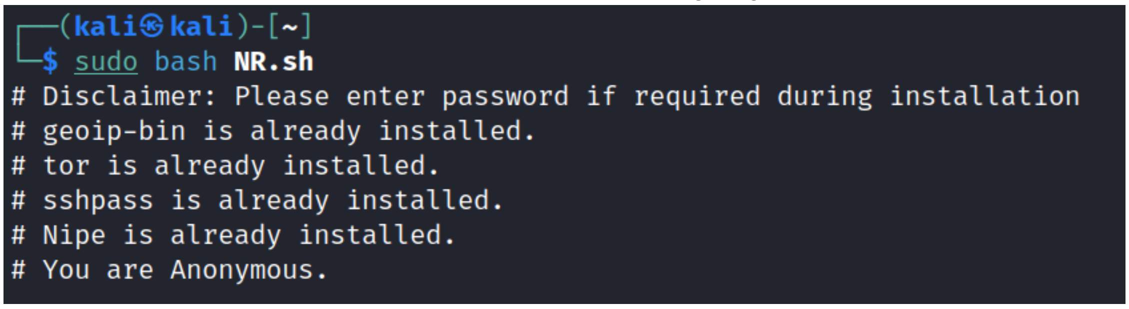


```
#Check if network connection is anonymous, if it is not, exit the application immediately.
#STATUS variable is to print out the current status of Nipe

cd nipe/
echo kali | sudo -S perl nipe.pl start
STATUS=$(sudo perl nipe.pl status | grep Status | awk '{print $3}')

if [[ "$STATUS" == 'true' ]]
then
    echo '# You are Anonymous.'
else
    echo '# You are not Anonymous'
    exit
fi
```

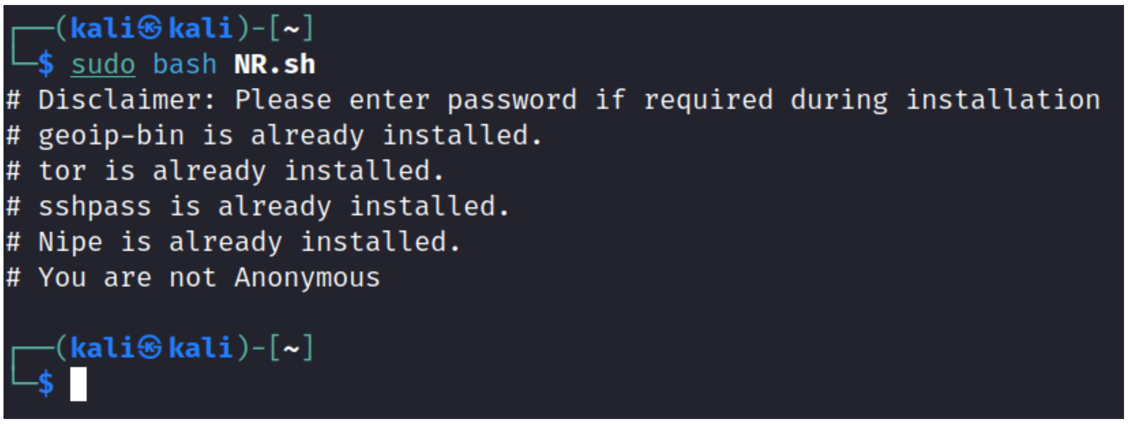
To enable Nipe, you need to navigate to the Nipe directory using cd nipe/. This allows you to start Nipe. The STATUS variable is used to indicate the current status of Nipe.

Using an if statement, if the status is true, it means that Nipe is turned on, and it will display that you are anonymous. If the status is false, indicating that Nipe is not turned on, it will display that you are not anonymous and exit the application immediately.

Example of how the script executes to check for anonymity



```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are Anonymous.
```

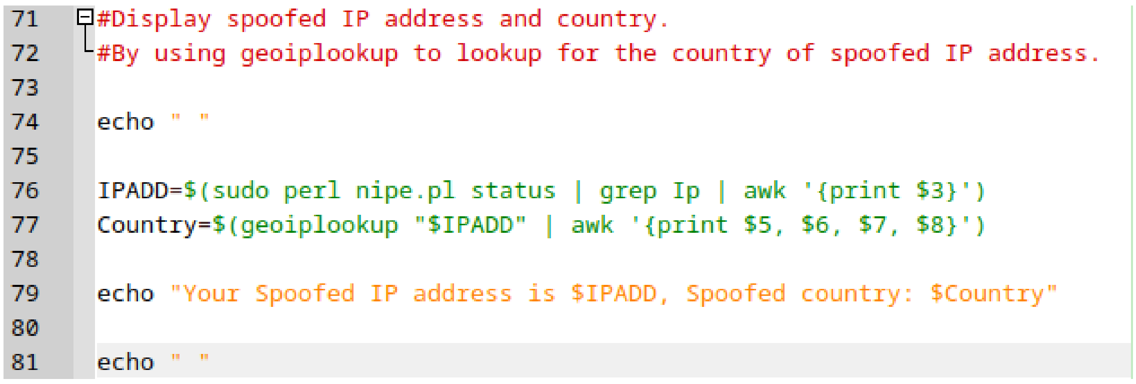


```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are not Anonymous

┌──(kali㉿kali)-[~]
└─$ 
```

An example where anonymity is not achieved would cause the script to exit itself.

Methodologies - Spoofed Country Check

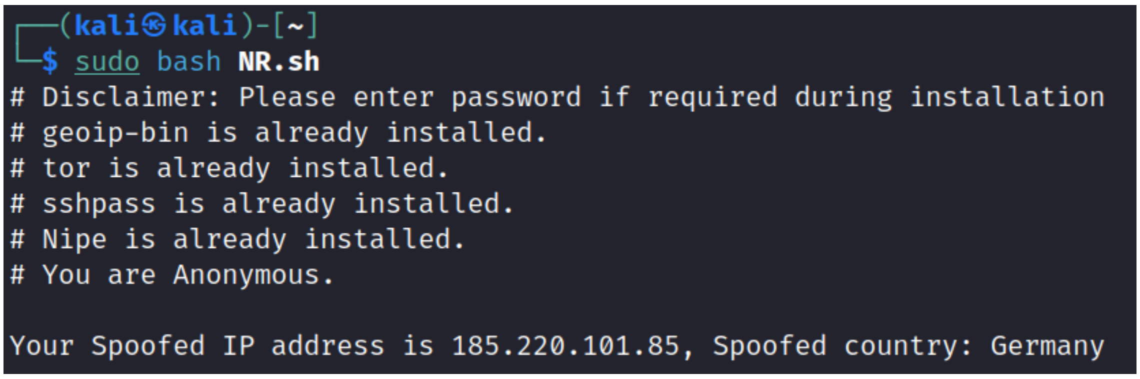
Example of detailed command in Kali Geany



```
#Display spoofed IP address and country.
#By using geoiplookup to lookup for the country of spoofed IP address.

echo " "

IPADD=$(sudo perl nipe.pl status | grep Ip | awk '{print $3}')
Country=$(geoiplookup "$IPADD" | awk '{print $5, $6, $7, $8}')

echo "Your Spoofed IP address is $IPADD, Spoofed country: $Country"

echo " "
```

I have decided to use the geoiplookup command to check the spoofed country, as it displays the country's full name. The IPADD variable prints out the spoofed IP address specifically. The Country variable prints out the country name as stated by geoiplookup. Using awk '{print $5, $6, $7, $8}', allows geoiplookup to print more than one word, for example, 'United States'.

Example of how the script checks for spoofed country

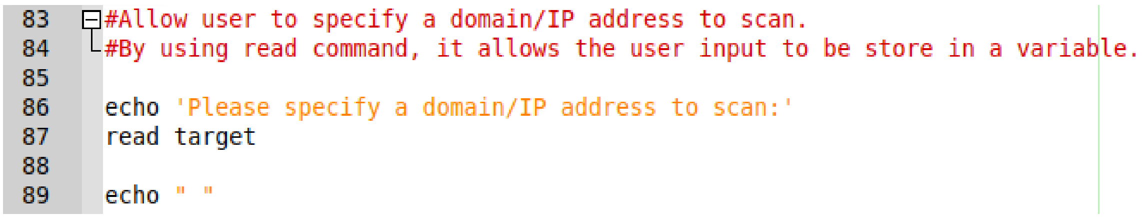


```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are Anonymous.

Your Spoofed IP address is 185.220.101.85, Spoofed country: Germany
```

For example, the IP address 185.220.101.85 is spoofed to appear as if originating from Germany.

Methodologies - Input of domain/IP address

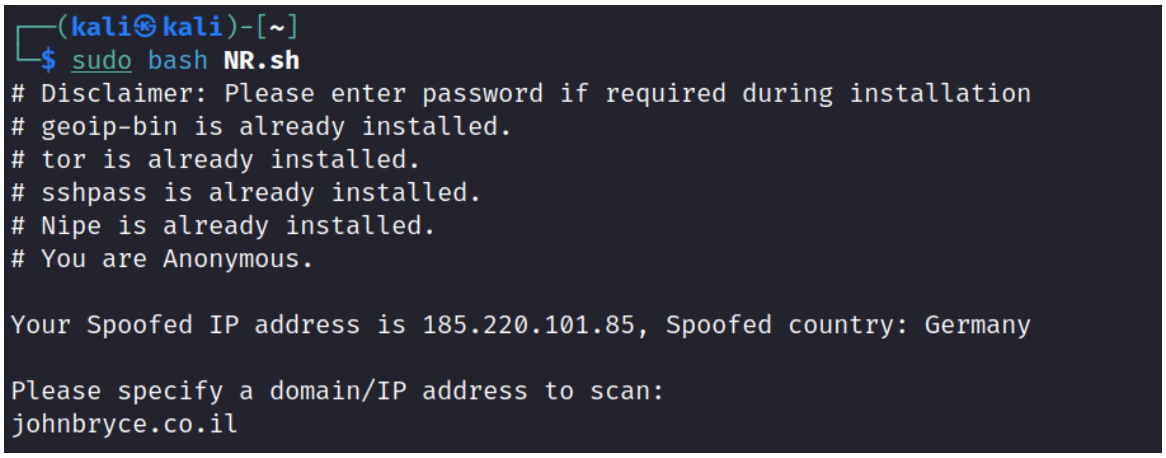
Example of detailed command in Kali Geany



```
#Allow user to specify a domain/IP address to scan.
#By using read command, it allows the user input to be store in a variable.

echo 'Please specify a domain/IP address to scan:'
read target

echo " "
```

To allow the user to input the domain/IP address, I use the read command. This command stores the input as a variable for further use in subsequent steps.

Example of how the script checks for user input of domain/IP address

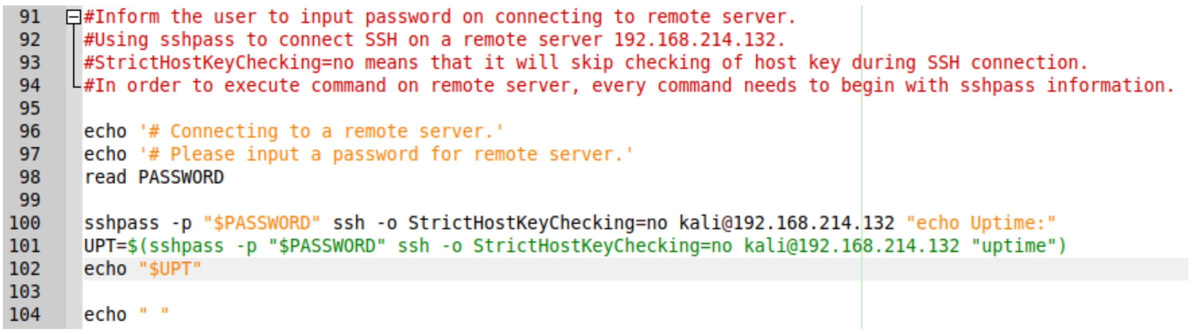


```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are Anonymous.

Your Spoofed IP address is 185.220.101.85, Spoofed country: Germany

Please specify a domain/IP address to scan:
johnbryce.co.il
```

For example, the user inputs 'johnbryce.co.il' as the domain/IP address.

Methodologies - Connecting to remote server and displaying information

Example of detailed command in Kali Geany



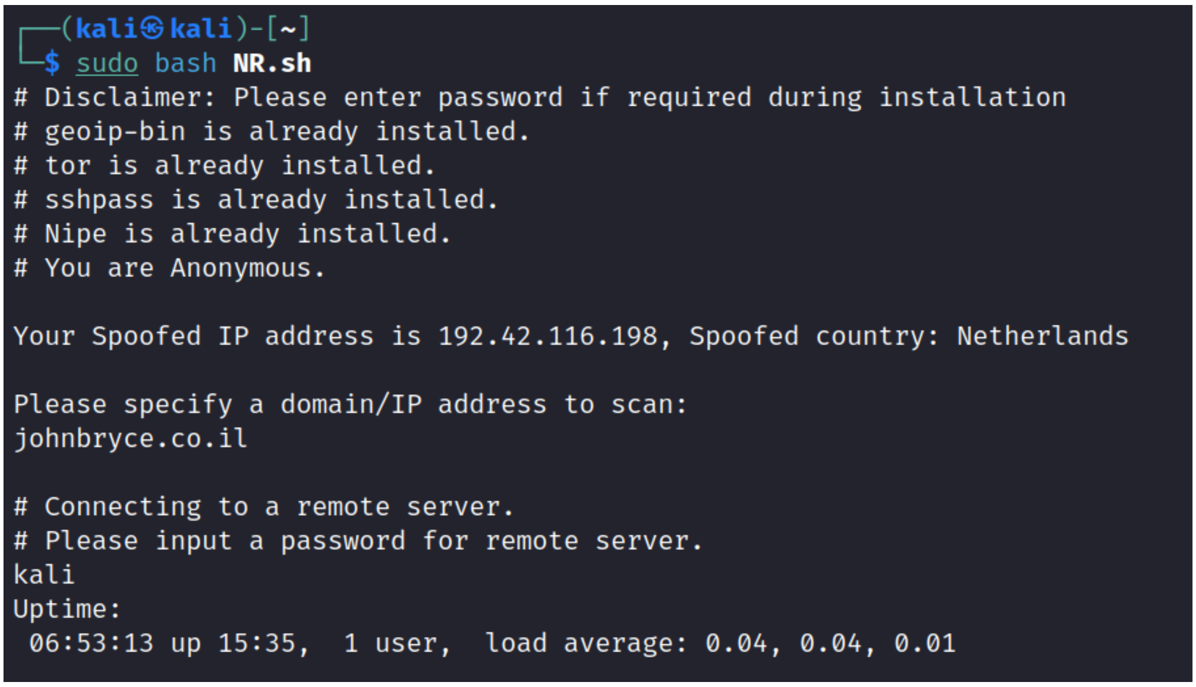
```
#Inform the user to input password on connecting to remote server.
#Using sshpass to connect SSH on a remote server 192.168.214.132.
#StrictHostKeyChecking=no means that it will skip checking of host key during SSH connection.
#In order to execute command on remote server, every command needs to begin with sshpass information.

echo '# Connecting to a remote server.'
echo '# Please input a password for remote server.'
read PASSWORD

sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "echo Uptime:"
UPT=$(sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "uptime")
echo "$UPT"

echo " "
```

Using sshpass allows the user to automatically connect to a remote server without needing to authenticate by entering a password directly in the script.

I have added a prompt informing the user that the script is about to connect to a remote server, where the user needs to input a password for authentication. By using -p "$PASSWORD", the user does not have to re-enter the password. Immediately after the ssh command, I included -o StrictHostKeyChecking=no to skip the prompt asking 'Are you sure you want to continue connecting (yes/no)?'.

To ensure the execution of commands on the remote server, I placed the command right after the remote server's IP address. For example, for the uptime command, I used UPT as a variable to store the remote server's uptime command.

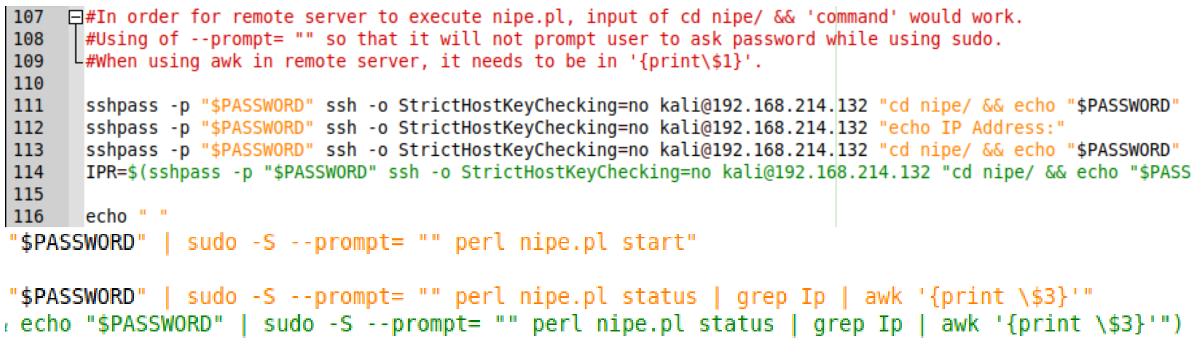
An example of the script successfully connecting to a remote server and displaying its uptime



```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are Anonymous.

Your Spoofed IP address is 192.42.116.198, Spoofed country: Netherlands

Please specify a domain/IP address to scan:
johnbryce.co.il

# Connecting to a remote server.
# Please input a password for remote server.
kali
Uptime:
 06:53:13 up 15:35,  1 user,  load average: 0.04, 0.04, 0.01
```

Methodologies - Connecting to remote server and displaying information

Example of detailed command in Kali Geany



```
#In order for remote server to execute nipe.pl, input of cd nipe/ && 'command' would work.
#Using of --prompt= "" so that it will not prompt user to ask password while using sudo.
#When using awk in remote server, it needs to be in '{print\$1}'.

sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "cd nipe/ && echo "$PASSWORD"
sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "echo IP Address:"
sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "cd nipe/ && echo "$PASSWORD"
IPR=$(sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "cd nipe/ && echo "$PASS

echo " "
"$PASSWORD" | sudo -S --prompt= "" perl nipe.pl start"

"$PASSWORD" | sudo -S --prompt= "" perl nipe.pl status | grep Ip | awk '{print \$3}'"
echo "$PASSWORD" | sudo -S --prompt= "" perl nipe.pl status | grep Ip | awk '{print \$3}'")
```

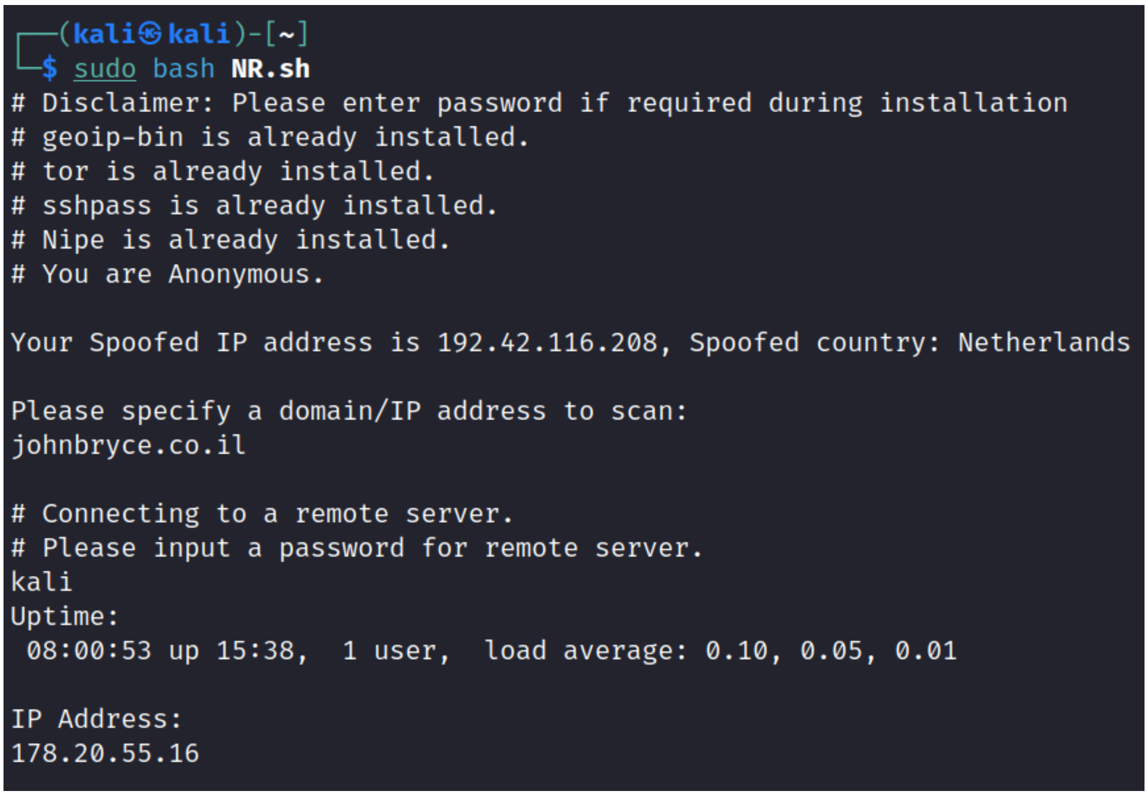
To display the IP address on a remote server, I need to use the nipe.pl tool, so I have to change directory (cd) into the nipe folder for it to work. I use && to ensure the script changes directory and then executes the next command.

When using sudo -S, I opt to use --prompt="" to prevent displaying [sudo] password for user:. I've also saved the command for retrieving the remote IP address as IPR so that it can be used to look up the country in the next step.

To extract the IP address from the remote server using awk, I use grep IP | '{print\$3}' to print the value. This allows the local shell to expand the empty variable and send the complete IP address.

These changes should make the text clearer and more grammatically correct.

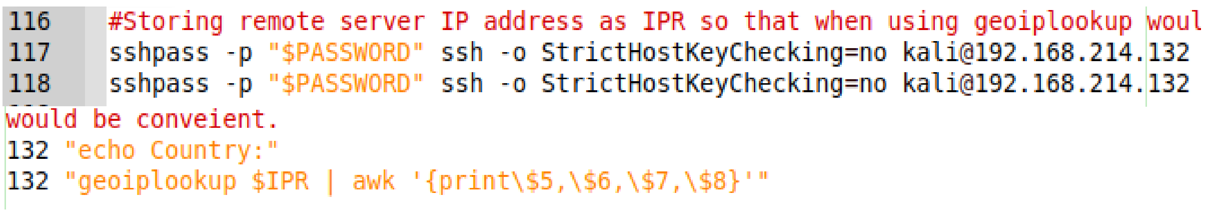
Example of how the script executes commands on a remote server and displays the IP address



```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are Anonymous.

Your Spoofed IP address is 192.42.116.208, Spoofed country: Netherlands

Please specify a domain/IP address to scan:
johnbryce.co.il

# Connecting to a remote server.
# Please input a password for remote server.
kali
Uptime:
 08:00:53 up 15:38,  1 user,  load average: 0.10, 0.05, 0.01

IP Address:
178.20.55.16
```

Methodologies - Connecting to remote server and displaying information

Example of detailed command in Kali Geany

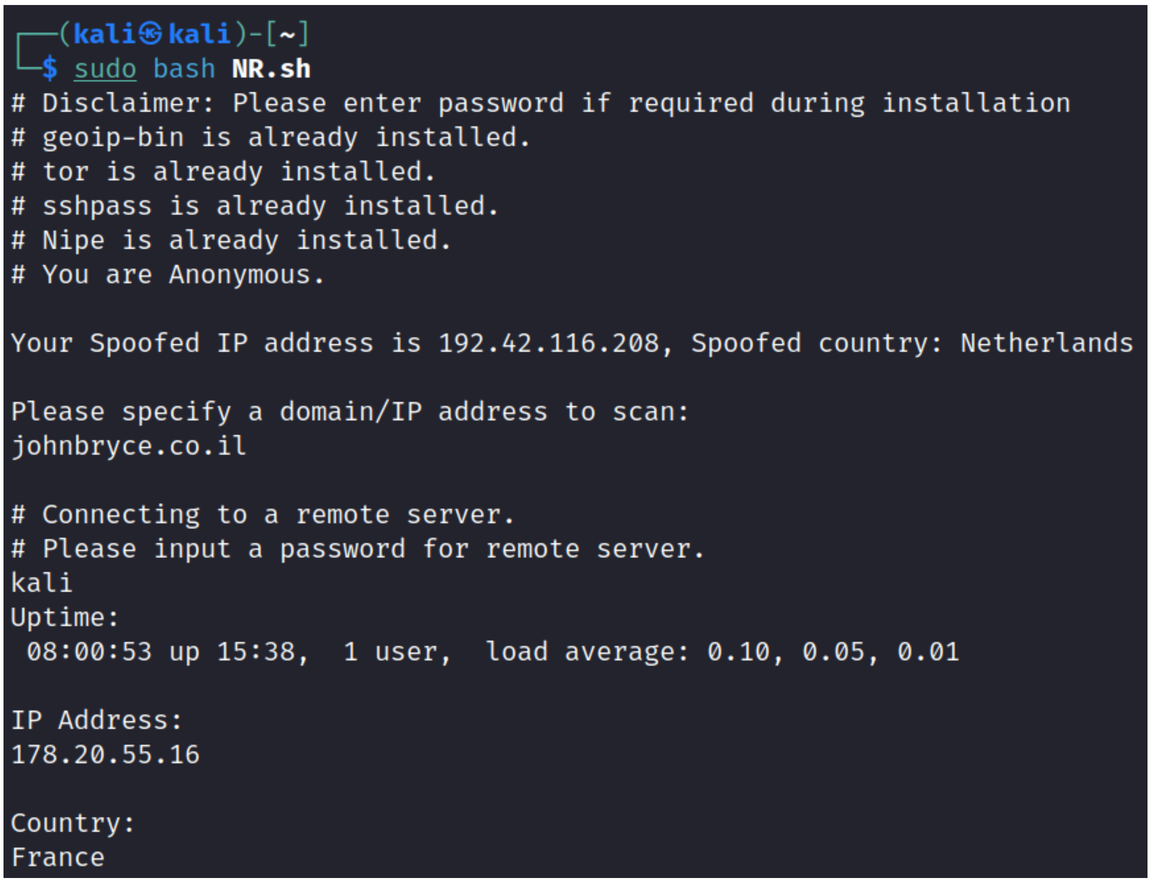


```
#Storing remote server IP address as IPR so that when using geoiplookup woul
sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132
sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132
would be conveient.
132 "echo Country:"
132 "geoiplookup $IPR | awk '{print\$5,\$6,\$7,\$8}'"
```

For displaying the country name, the command would be geoiplookup $IPR, where $IPR is the variable storing the remote server's IP address. The country variable is used to print out the country name returned by geoiplookup.

To ensure geoiplookup prints multiple words, such as "United States," I use awk '{print\$5,\$6,\$7,\$8}'. This allows geoiplookup to display the complete country name.

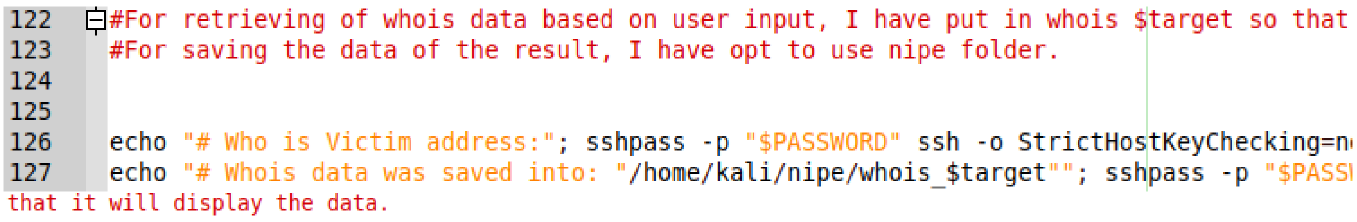
Example of how the script executes commands on a remote server and displays the country name



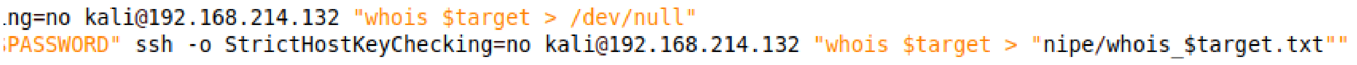
```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are Anonymous.

Your Spoofed IP address is 192.42.116.208, Spoofed country: Netherlands

Please specify a domain/IP address to scan:
johnbryce.co.il

# Connecting to a remote server.
# Please input a password for remote server.
kali
Uptime:
 08:00:53 up 15:38,  1 user,  load average: 0.10, 0.05, 0.01

IP Address:
178.20.55.16

Country:
France
```

Methodologies - Displaying and saving data of whois information

Example of detailed command in Kali Geany



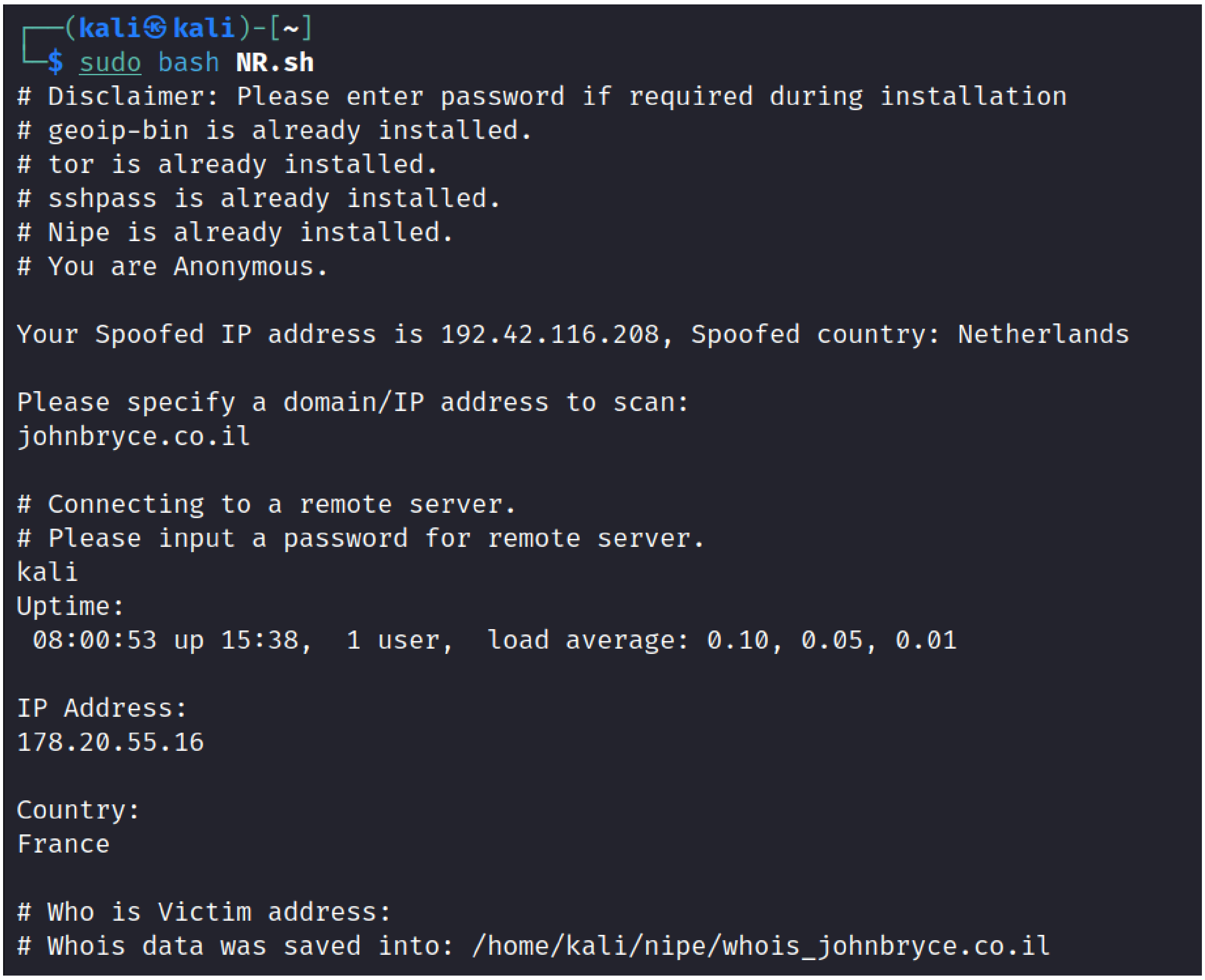
```
#For retrieving of whois data based on user input, I have put in whois $target so that
#For saving the data of the result, I have opt to use nipe folder.


echo "# Who is Victim address:"; sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=n
echo "# Whois data was saved into: "/home/kali/nipe/whois_$target""; sshpass -p "$PASS
that it will display the data.
```



```
ng=no kali@192.168.214.132 "whois $target > /dev/null"
PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "whois $target > "nipe/whois_$target.txt""
```

To display information from whois, I start by echoing the title. Then, I use sshpass commands to execute the script on the remote server. For saving the data, I choose to use the nipe file, so I echo the location where it will be saved, which is "/home/kali/nipe/". Using whois $target > "nipe/whois_$target" allows the script to save the input as whois_$target inside the nipe folder.

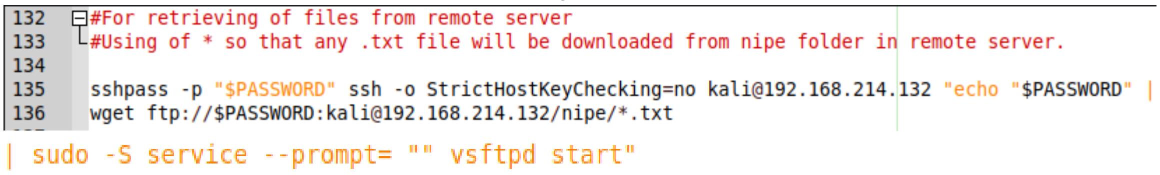
Example of how the script displays information from a remote server



```
┌──(kali㉿kali)-[~]
└─$ sudo bash NR.sh
# Disclaimer: Please enter password if required during installation
# geoip-bin is already installed.
# tor is already installed.
# sshpass is already installed.
# Nipe is already installed.
# You are Anonymous.

Your Spoofed IP address is 192.42.116.208, Spoofed country: Netherlands

Please specify a domain/IP address to scan:
johnbryce.co.il

# Connecting to a remote server.
# Please input a password for remote server.
kali
Uptime:
 08:00:53 up 15:38,  1 user,  load average: 0.10, 0.05, 0.01

IP Address:
178.20.55.16

Country:
France

# Who is Victim address:
# Whois data was saved into: /home/kali/nipe/whois_johnbryce.co.il
```

Methodologies - Downloading file from remote server using FTP

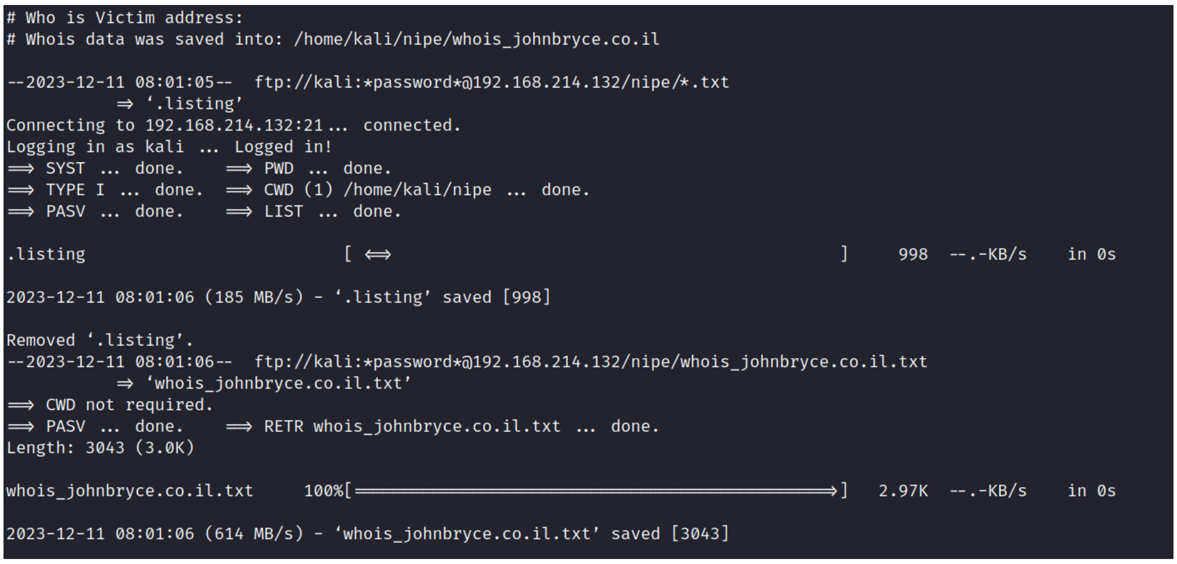
Example of detailed command in Kali Geany



```
#For retrieving of files from remote server
#Using of * so that any .txt file will be downloaded from nipe folder in remote server.

sshpass -p "$PASSWORD" ssh -o StrictHostKeyChecking=no kali@192.168.214.132 "echo "$PASSWORD" |
wget ftp://$PASSWORD:kali@192.168.214.132/nipe/*.txt
| sudo -S service --prompt= "" vsftpd start"
```

Firstly, I have used the command to turn on vsftpd to enable FTP downloads. To download files from the remote server to the main machine using FTP, I have opted to use wget followed by the command ftp://[username]:[password]@[host address]/nipe/*.txt. Using *.txt ensures that all files ending with .txt are universally downloaded using FTP.

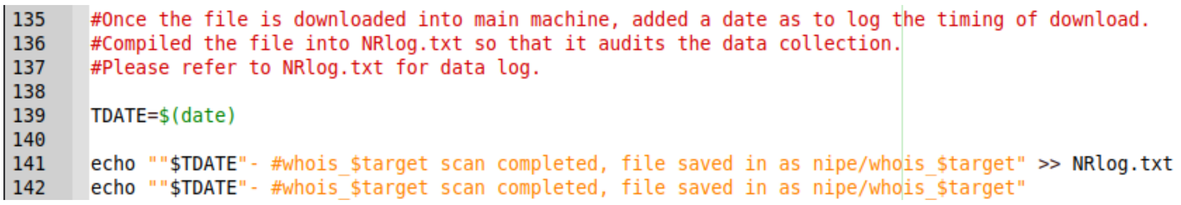
Example of how the script performs the FTP download



```
# Who is Victim address:
# Whois data was saved into: /home/kali/nipe/whois_johnbryce.co.il

--2023-12-11 08:01:05--  ftp://kali:*password*@192.168.214.132/nipe/*.txt
           ⇒ '.listing'
Connecting to 192.168.214.132:21... connected.
Logging in as kali ... Logged in!
⟹ SYST ... done.    ⟹ PWD ... done.
⟹ TYPE I ... done.  ⟹ CWD (1) /home/kali/nipe ... done.
⟹ PASV ... done.    ⟹ LIST ... done.

.listing                          [ ⟺                                              ]     998  --.-KB/s    in 0s

2023-12-11 08:01:06 (185 MB/s) - '.listing' saved [998]

Removed '.listing'.
--2023-12-11 08:01:06--  ftp://kali:*password*@192.168.214.132/nipe/whois_johnbryce.co.il.txt
           ⇒ 'whois_johnbryce.co.il.txt'
⟹ CWD not required.
⟹ PASV ... done.    ⟹ RETR whois_johnbryce.co.il.txt ... done.
Length: 3043 (3.0K)

whois_johnbryce.co.il.txt     100%[=================================================>]   2.97K  --.-KB/s    in 0s

2023-12-11 08:01:06 (614 MB/s) - 'whois_johnbryce.co.il.txt' saved [3043]
```

Methodologies - Compiling of downloaded data

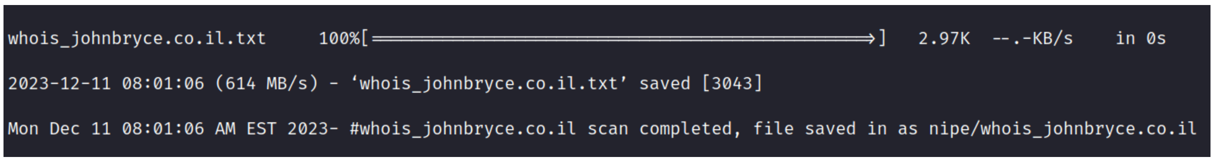
Example of detailed command in Kali Geany



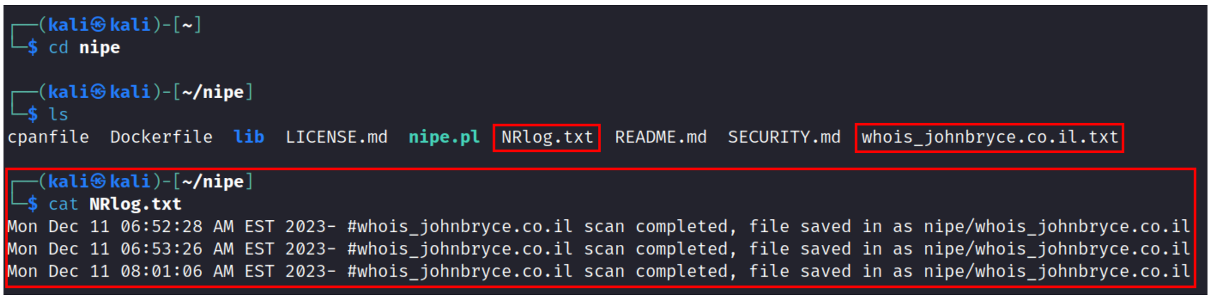
```
#Once the file is downloaded into main machine, added a date as to log the timing of download.
#Compiled the file into NRlog.txt so that it audits the data collection.
#Please refer to NRlog.txt for data log.

TDATE=$(date)

echo ""$TDATE"- #whois_$target scan completed, file saved in as nipe/whois_$target" >> NRlog.txt
echo ""$TDATE"- #whois_$target scan completed, file saved in as nipe/whois_$target"
```

In order to compile the data collected from the remote server, I have added a date command to store the time and date of the downloaded file. Next, I echo out the details: '#whois_$target scan completed, file saved in nipe/whois_$target', so that it stores the information in NRlog.txt.

Example of how the script compiles the downloaded data

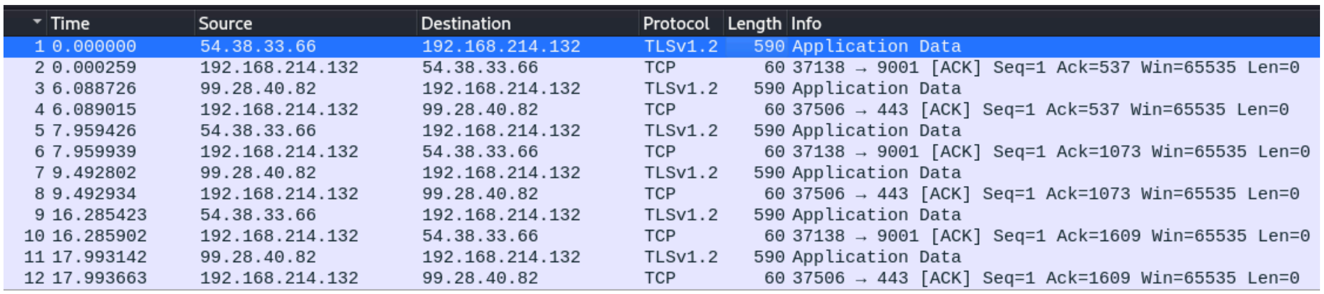


```
whois_johnbryce.co.il.txt     100%[=================================================>]   2.97K  --.-KB/s    in 0s

2023-12-11 08:01:06 (614 MB/s) - ‘whois_johnbryce.co.il.txt’ saved [3043]

Mon Dec 11 08:01:06 AM EST 2023- #whois_johnbryce.co.il scan completed, file saved in as nipe/whois_johnbryce.co.il
```



```
┌──(kali㉿kali)-[~]
└─$ cd nipe

┌──(kali㉿kali)-[~/nipe]
└─$ ls
cpanfile  Dockerfile  lib  LICENSE.md  nipe.pl  NRlog.txt  README.md  SECURITY.md  whois_johnbryce.co.il.txt

┌──(kali㉿kali)-[~/nipe]
└─$ cat NRlog.txt
Mon Dec 11 06:52:28 AM EST 2023- #whois_johnbryce.co.il scan completed, file saved in as nipe/whois_johnbryce.co.il
Mon Dec 11 06:53:26 AM EST 2023- #whois_johnbryce.co.il scan completed, file saved in as nipe/whois_johnbryce.co.il
Mon Dec 11 08:01:06 AM EST 2023- #whois_johnbryce.co.il scan completed, file saved in as nipe/whois_johnbryce.co.il
```

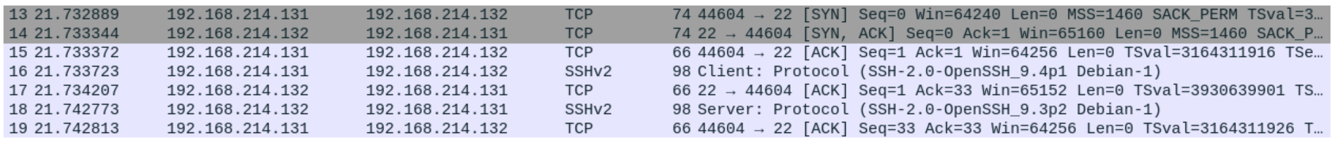
Wireshark Analysis



```
    Time        Source          Destination     Protocol Length Info
 1 0.000000    54.38.33.66     192.168.214.132 TLSv1.2  590 Application Data
 2 0.000259    192.168.214.132 54.38.33.66     TCP       60 37138 → 9001 [ACK] Seq=1 Ack=537 Win=65535 Len=0
 3 6.088726    99.28.40.82     192.168.214.132 TLSv1.2  590 Application Data
 4 6.089015    192.168.214.132 99.28.40.82     TCP       60 37506 → 443 [ACK] Seq=1 Ack=537 Win=65535 Len=0
 5 7.959426    54.38.33.66     192.168.214.132 TLSv1.2  590 Application Data
 6 7.959939    192.168.214.132 54.38.33.66     TCP       60 37138 → 9001 [ACK] Seq=1 Ack=1073 Win=65535 Len=0
 7 9.492802    99.28.40.82     192.168.214.132 TLSv1.2  590 Application Data
 8 9.492934    192.168.214.132 99.28.40.82     TCP       60 37506 → 443 [ACK] Seq=1 Ack=1073 Win=65535 Len=0
 9 16.285423   54.38.33.66     192.168.214.132 TLSv1.2  590 Application Data
10 16.285902   192.168.214.132 54.38.33.66     TCP       60 37138 → 9001 [ACK] Seq=1 Ack=1609 Win=65535 Len=0
11 17.993142   99.28.40.82     192.168.214.132 TLSv1.2  590 Application Data
12 17.993663   192.168.214.132 99.28.40.82     TCP       60 37506 → 443 [ACK] Seq=1 Ack=1609 Win=65535 Len=0
```

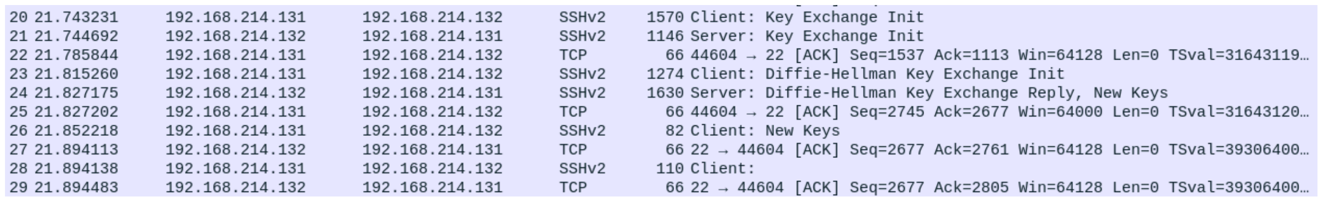
Firstly, the source shows that it is from 54.38.33.66. The source IP is from France; therefore, the nipe tool is turned on to change the IP to that country. The destination IP is the remote server's IP, so it is linking up using TLSv1.2 protocol. TLS, also known as Transport Layer Security, provides network communication security between computers. The information appears as Application data in Wireshark.

The remote server acknowledges the request from 54.38.33.66, but due to packet loss in TCP fast transmit, the connection was unable to establish. The same issue occurred with 99.28.40.82.



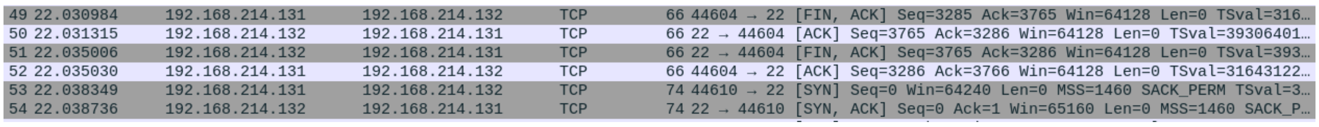
```
13 21.732889   192.168.214.131   192.168.214.132   TCP     74 44604 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3…
14 21.733344   192.168.214.132   192.168.214.131   TCP     74 22 → 44604 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_P…
15 21.733372   192.168.214.131   192.168.214.132   TCP     66 44604 → 22 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3164311916 TSe…
16 21.733723   192.168.214.131   192.168.214.132   SSHv2   98 Client: Protocol (SSH-2.0-OpenSSH_9.4p1 Debian-1)
17 21.734207   192.168.214.132   192.168.214.131   TCP     66 22 → 44604 [ACK] Seq=1 Ack=33 Win=65152 Len=0 TSval=3930639901 TS…
18 21.742773   192.168.214.132   192.168.214.131   SSHv2   98 Server: Protocol (SSH-2.0-OpenSSH_9.3p2 Debian-1)
19 21.742813   192.168.214.131   192.168.214.132   TCP     66 44604 → 22 [ACK] Seq=33 Ack=33 Win=64256 Len=0 TSval=3164311926 T…
```

Next, when using SSH to connect, the connection is established as shown by SYN, SYN-ACK, and ACK. Using the three-way handshake via TCP, it successfully connects to the remote server at 192.168.214.132. On line 16, it indicates the SSHv2 protocol and its version (SSH-2.0-OpenSSH_9.4p1 Debian-1). Lines 17 and 18 show acknowledgements from the remote server to the local machine attempting the SSH connection.



```
20 21.743231   192.168.214.131   192.168.214.132   SSHv2   1570 Client: Key Exchange Init
21 21.744692   192.168.214.132   192.168.214.131   SSHv2   1146 Server: Key Exchange Init
22 21.785844   192.168.214.131   192.168.214.132   TCP       66 44604 → 22 [ACK] Seq=1537 Ack=1113 Win=64128 Len=0 TSval=31643119…
23 21.815260   192.168.214.131   192.168.214.132   SSHv2   1274 Client: Diffie-Hellman Key Exchange Init
24 21.827175   192.168.214.132   192.168.214.131   SSHv2   1630 Server: Diffie-Hellman Key Exchange Reply, New Keys
25 21.827202   192.168.214.131   192.168.214.132   TCP       66 44604 → 22 [ACK] Seq=2745 Ack=2677 Win=64000 Len=0 TSval=31643120…
26 21.852218   192.168.214.131   192.168.214.132   SSHv2     82 Client: New Keys
27 21.894113   192.168.214.132   192.168.214.131   TCP       66 22 → 44604 [ACK] Seq=2677 Ack=2761 Win=64128 Len=0 TSval=39306400…
28 21.894138   192.168.214.131   192.168.214.132   SSHv2    110 Client:
29 21.894483   192.168.214.132   192.168.214.131   TCP       66 22 → 44604 [ACK] Seq=2677 Ack=2805 Win=64128 Len=0 TSval=39306400…
```

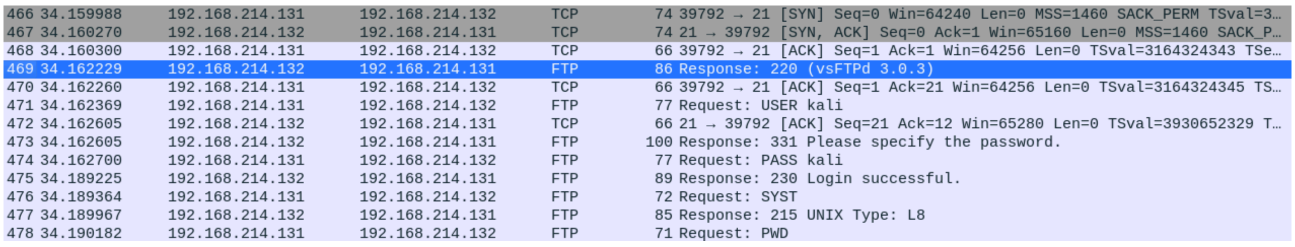
At frames 20 and 21, SSH initiates Key Exchange to encrypt data, ensuring information security during data transfer between the client and server. Using Diffie-Hellman Key Exchange allows for establishing a secure connection and generating new keys. These keys consist of the client key and server key respectively. The absence of information on the client side at frame 28 indicates it is awaiting further action.



```
49 22.030984   192.168.214.131   192.168.214.132   TCP   66 44604 → 22 [FIN, ACK] Seq=3285 Ack=3765 Win=64128 Len=0 TSval=316…
50 22.031315   192.168.214.132   192.168.214.131   TCP   66 22 → 44604 [ACK] Seq=3765 Ack=3286 Win=64128 Len=0 TSval=39306401…
51 22.035006   192.168.214.132   192.168.214.131   TCP   66 22 → 44604 [FIN, ACK] Seq=3765 Ack=3286 Win=64128 Len=0 TSval=393…
52 22.035030   192.168.214.131   192.168.214.132   TCP   66 44604 → 22 [ACK] Seq=3286 Ack=3766 Win=64128 Len=0 TSval=31643122…
53 22.038349   192.168.214.131   192.168.214.132   TCP   74 44610 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3…
54 22.038736   192.168.214.132   192.168.214.131   TCP   74 22 → 44610 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_P…
```

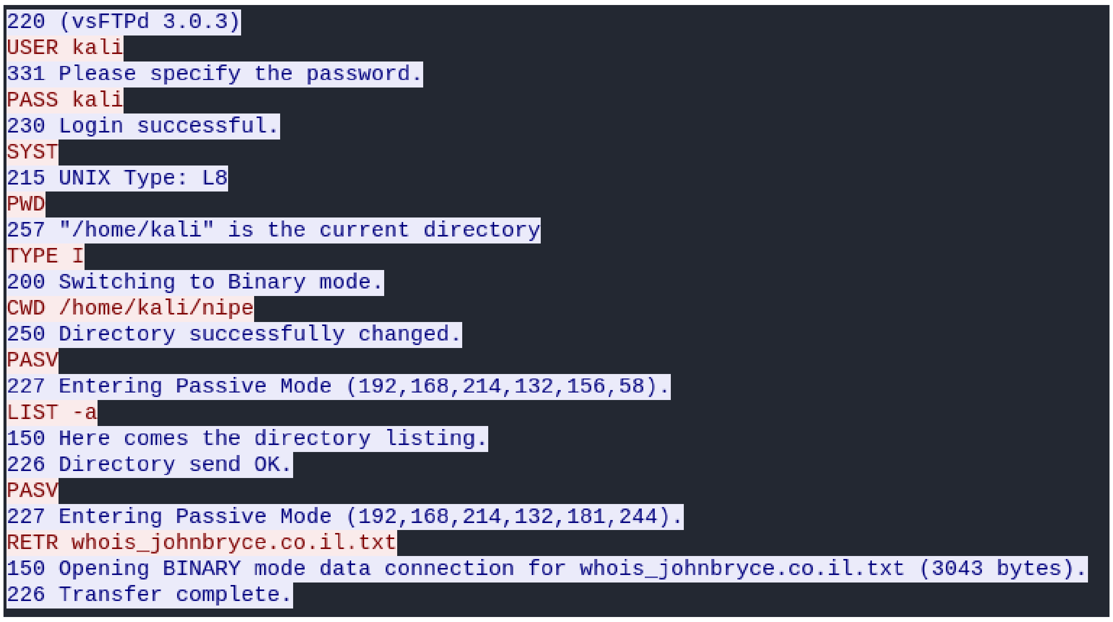
At frame 49, TCP displays information with FIN, ACK, indicating acknowledgment of the FIN packet and signaling the closure of the current connection. Frame 51 shows a similar pattern. Frame 53 indicates the re-establishment of the connection with SYN, ACK. These events occur due to executing commands on the remote server.

Wireshark Analysis



```
466 34.159988    192.168.214.131    192.168.214.132    TCP     74 39792 → 21 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3…
467 34.160270    192.168.214.132    192.168.214.131    TCP     74 21 → 39792 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_P…
468 34.160300    192.168.214.131    192.168.214.132    TCP     66 39792 → 21 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3164324343 TSe…
469 34.162229    192.168.214.132    192.168.214.131    FTP     86 Response: 220 (vsFTPd 3.0.3)
470 34.162260    192.168.214.131    192.168.214.132    TCP     66 39792 → 21 [ACK] Seq=1 Ack=21 Win=64256 Len=0 TSval=3164324345 TS…
471 34.162369    192.168.214.131    192.168.214.132    FTP     77 Request: USER kali
472 34.162605    192.168.214.132    192.168.214.131    TCP     66 21 → 39792 [ACK] Seq=21 Ack=12 Win=65280 Len=0 TSval=3930652329 T…
473 34.162605    192.168.214.132    192.168.214.131    FTP    100 Response: 331 Please specify the password.
474 34.162700    192.168.214.131    192.168.214.132    FTP     77 Request: PASS kali
475 34.189225    192.168.214.132    192.168.214.131    FTP     89 Response: 230 Login successful.
476 34.189364    192.168.214.131    192.168.214.132    FTP     72 Request: SYST
477 34.189967    192.168.214.132    192.168.214.131    FTP     85 Response: 215 UNIX Type: L8
478 34.190182    192.168.214.131    192.168.214.132    FTP     71 Request: PWD
```

Lastly, for FTP, I would have to follow the TCP stream to gain clearer information on how the FTP process is executed via the Wireshark platform.



```
220 (vsFTPd 3.0.3)
USER kali
331 Please specify the password.
PASS kali
230 Login successful.
SYST
215 UNIX Type: L8
PWD
257 "/home/kali" is the current directory
TYPE I
200 Switching to Binary mode.
CWD /home/kali/nipe
250 Directory successfully changed.
PASV
227 Entering Passive Mode (192,168,214,132,156,58).
LIST -a
150 Here comes the directory listing.
226 Directory send OK.
PASV
227 Entering Passive Mode (192,168,214,132,181,244).
RETR whois_johnbryce.co.il.txt
150 Opening BINARY mode data connection for whois_johnbryce.co.il.txt (3043 bytes).
226 Transfer complete.
```

By following the TCP stream of the FTP process, it clearly shows that FTP is an insecure method of downloading files. It reveals the username and password in the first five lines of the remote server. Additionally, other information such as UNIX type, current directory, directory listing, and file downloads are transmitted without encryption.

Discussion - File Transfer Protocol

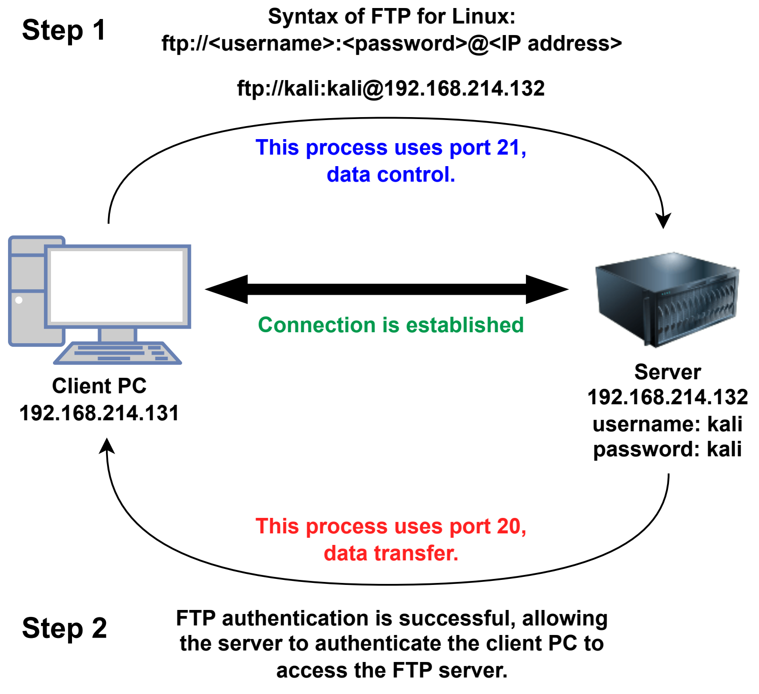
FTP Analysis

FTP, also known as File Transfer Protocol, resides within the application layer of the OSI model. It allows users to access a server and transfer files by logging in with the server's username and password. If the server host shuts down the connection or closes the port, the user may not be able to access the FTP server.

FTP is commonly used for uploading or downloading files for individual or business purposes. The request for comments for FTP is RFC 959, while the specification for the FTP protocol is RFC 765. FTP enables the transfer of multiple files in a single process, rather than manually uploading or downloading files one by one. This functionality saves time when gathering or uploading data across servers. Additionally, FTP requires an established internet connection to function.

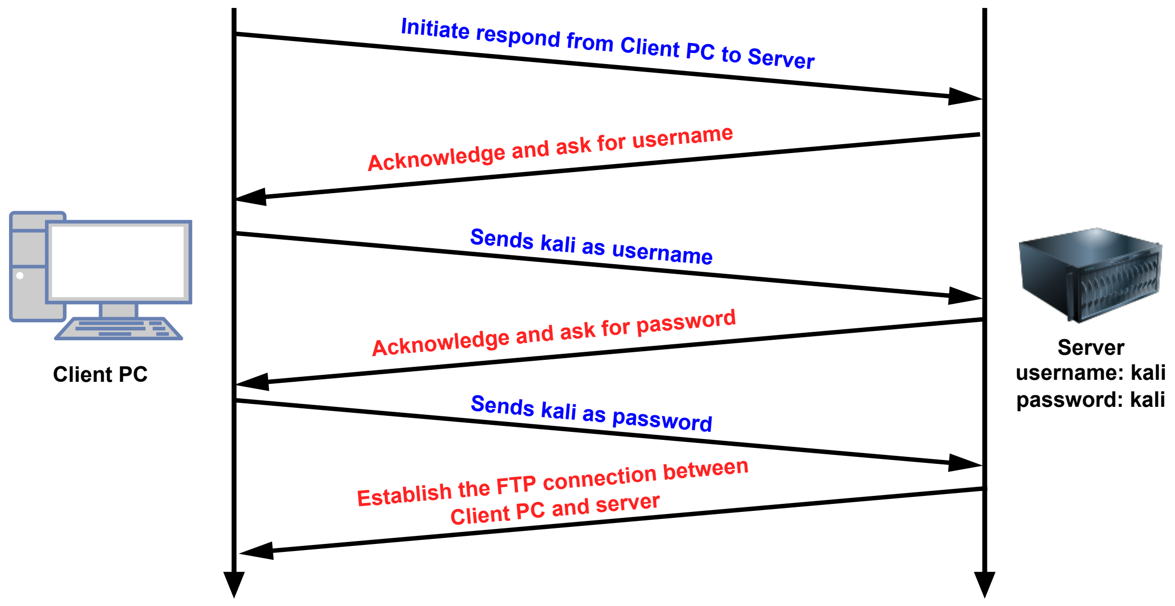
To initiate an FTP connection, the user must first have the server's username, password, IP address, and the default port, which is port 21. Below is an example of how the user should input these details.

FTP via Kali Command



Step 1
Syntax of FTP for Linux:
ftp://<username>:<password>@<IP address>
ftp://kali:kali@192.168.214.132
This process uses port 21, data control.
Connection is established
Client PC
192.168.214.131
Server
192.168.214.132
username: kali
password: kali
This process uses port 20, data transfer.
Step 2
FTP authentication is successful, allowing the server to authenticate the client PC to access the FTP server.

FTP breakdown of sequence



Initiate respond from Client PC to Server
Acknowledge and ask for username
Sends kali as username
Acknowledge and ask for password
Sends kali as password
Establish the FTP connection between Client PC and server
Client PC
Server
username: kali
password: kali

Discussion - File Transfer Protocol

FTP Analysis

The FTP header length needs to be 4, while the data length needs to be 0. Only for the access control page, the header length needs to be 5. There are four different data types used by FTP for transferring data: ASCII, image, EBCDIC, and local. ASCII and EBCDIC both handle text, though EBCDIC specifically supports plain text. Image mode transfers data in binary form byte by byte, while local mode uses a 36-bit system to facilitate file transfers between systems. Unicode mode employs UTF-8 encoding.

Within text files, FTP offers three control options for printing methods: non-print, telnet, and ASA. File structures in FTP include FRP, where 'F' stands for file structure, 'R' for record structure, and 'P' for page structure; the default structure is file. Data transfer modes are categorized into three types: stream mode, block mode, and compressed mode, with stream mode being the default.

FTP operates in both active and passive modes. In active mode FTP, the client uses a random port to connect to the server's port 21 of the control channel, while the server sends the connection back to the client's computer on port 20. In passive mode FTP, the client also connects via a random port to the server's port 21 of the control channel.

Regarding the strengths and weaknesses of FTP, its strength lies in its ability to transfer multiple files without size restrictions, manage queued transfers sequentially, and resume interrupted transfers. However, its weaknesses include the exposure of usernames, passwords, and files in plain text, making it vulnerable to brute-force attacks. FTP lacks data encryption and does not scan for viruses during file transfers, potentially exposing servers and clients to malware.

In terms of the CIA (Confidentiality, Integrity, and Availability) model, FTP's confidentiality is compromised due to the exposure of usernames and passwords, which can lead to unauthorized access and data theft. FTP's lack of encryption compromises data integrity, allowing unauthorized access to data. Availability is easily accessible once authenticated, but lacks additional layers of security for user access authentication.

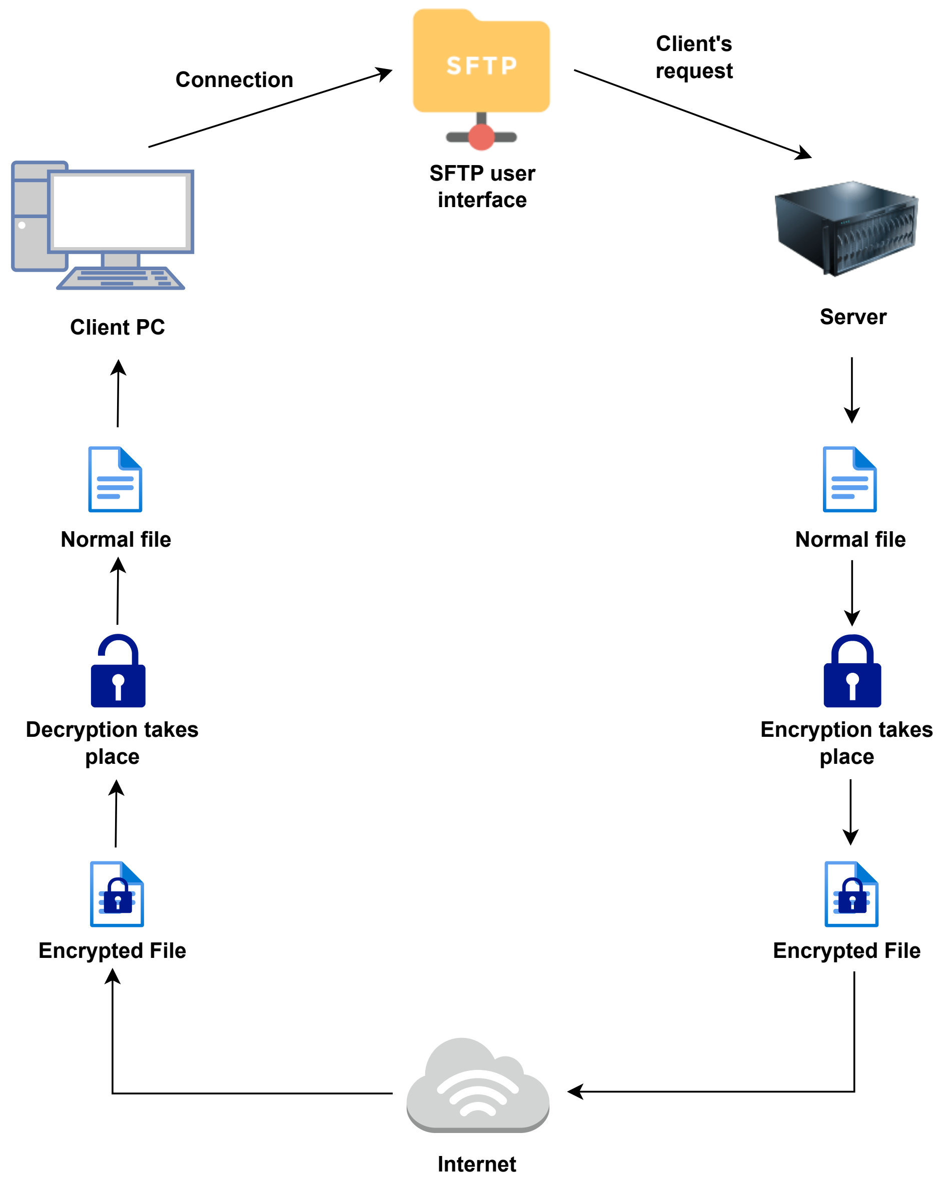
Discussion - Secure Network Protocol (SFTP)

SFTP analysis

By using a more secure network protocol such as SFTP, it provides a much safer way to transfer files and data when needed. SFTP stands for Secure File Transfer Protocol, adding an additional layer of encryption compared to FTP itself, thereby enhancing its security. SFTP operates on port 22, the same port used by SSH (Secure Shell). It encrypts data to secure the movement of sensitive information such as passwords or file contents from unauthorized access. SFTP establishes a secure connection between the client and the server, ensuring all data undergoes encryption during processing. It offers various authentication methods, including usernames, passwords, and SSH keys, to strengthen security measures according to the server's requirements.

Based on the principles of the CIA triad (Confidentiality, Integrity, and Availability), I believe that what SFTP offers minimizes the risk of data leaks compared to using FTP as discussed previously.

The diagram below provides a brief breakdown of the Secure File Transfer Protocol.



Conclusion

In conclusion, I have learned that FTP provides an easy way for users and servers to transfer files between each other, but it carries the risk of leaking sensitive information during the process. The exposure of sensitive information on FTP could lead to attacks from outside the connection, as it does not encrypt file data during transfer.

In the recommendation section, I will explain which protocol is better to use and why it offers enhanced security compared to FTP.

Recommendations

If I were to ensure safer file transfers, I would choose to use SFTP. The additional encryption it provides for data and file protection prevents attackers from unauthorized access, making it a significant factor in cybersecurity safety. While it may involve additional passwords or processes to obtain the desired files or data, it safeguards our computers against unauthorized access to sensitive information.

Alternatively, HTTPS is another protocol similar to SFTP that also includes built-in data encryption, ensuring confidentiality of information.

References

Scope 1 research website

Ödman, Anton. “See If a Package Exists in a Bash Script.” *Banjocode*, 7 Sept. 2020, www.banjocode.com/post/bash/package-exists-bash. Accessed 7 Dec. 2023.

Naik, Ganesh. “Learning Linux Shell Scripting - Second Edition by Ganesh Naik.” *O’Reilly Online Learning*, Packt Publishing, May 2018, www.oreilly.com/library/view/learning-linux-shell/9781788993197/0faa519d-98cd-4949-b2a7-454bbb87a204.xhtml#:~:text=The%20%2Dv%20option%20tells%20the,that%20has%20created%20an%20error. Accessed 7 Dec. 2023

Santos, Cleber J. “[Pentest] Stay Anonymous Using NIPE.” *Medium*, Medium, 28 Feb. 2021, cleberjsantos.medium.com/pentest-stay-anonymous-using-nipe-4d0a7c3bed1. Accessed 8 Dec. 2023

Amoany, Evans. “SSH Password Automation in Linux with Sshpass.” *Enable Sysadmin*, Red Hat, Inc., 31 Aug. 2020, www.redhat.com/sysadmin/ssh-automation-sshpass. Accessed 7 Dec. 2023

Geno, Chen. “Executing Sudo Command in Bash Script without Displaying It.” *Stack Overflow*, 19 Aug. 2021, stackoverflow.com/questions/68843636/executing-sudo-command-in-bash-script-without-displaying-it. Accessed on 9 Dec. 2023.

Srivastava, Anurag. “Execute Commands on Remote Machines Using Sshpass.” *Medium*, Level Up Coding, 23 May 2020, levelup.gitconnected.com/execute-commands-on-remote-machines-using-sshpass-1f9bc4452e15. Accessed on 10 Dec. 2023.

clk. “How to Print a Specific Column with AWK on a Remote SSH Session?” *Unix & Linux Stack Exchange*, 8 July 2016, unix.stackexchange.com/questions/294676/how-to-print-a-specific-column-with-awk-on-a-remote-ssh-session. Accessed 10 Dec. 2023.

Blanchon, Benoit. “Shell Script to Download Files from Remote Machine Using FTP.” *Stack Overflow*, 20 Sept. 2013, stackoverflow.com/questions/18914451/shell-script-to-download-files-from-remote-machine-using-ftp. Accessed 11 Dec. 2023.

Garn, Damon. “How to Capture and Analyze Traffic with Tcpdump: TechTarget.” *Networking*, TechTarget, 16 Aug. 2023, www.techtarget.com/searchnetworking/tutorial/How-to-capture-and-analyze-traffic-with-tcpdump. Accessed 12 Dec. 2023.

M. Sopha. “What Is TLS? Understanding Transport Layer Security and How It Works.” *Hostinger Tutorials*, 27 Sept. 2023, www.hostinger.com/tutorials/what-is-tls#What_Is_TLS. Accessed 12 Dec. 2023

Bernatavičius, Mindaugas. “What Is Duplicate Ack When Does It Occur?” *Stack Overflow*, 9 Jan. 2018, stackoverflow.com/questions/48148820/what-is-duplicate-ack-when-does-it-occur. Accessed on 12 Dec. 2023.

Jones, Russell. “SSH Handshake Explained.” *What Is SSH Handshake?*, 31 Mar. 2022, goteleport.com/blog/ssh-handshake-explained/. Accessed 12 Dec. 2023.

Tyson, Matthew. “Understand Diffie-Hellman Key Exchange.” *InfoWorld*, InfoWorld, 20 Jan. 2022, www.infoworld.com/article/3647751/understand-diffie-hellman-key-exchange.html. Accessed 12 Dec. 2023.

Bhardwaj, Rashmi. “What Is TCP Fin Packet?” *IP With Ease*, 17 Jan. 2022, ipwithease.com/what-is-tcp-fin-packet/. Accessed 12 Dec. 2023.

Scope 2 research website

Mitchell, Cory. “What Is File Transfer Protocol (FTP) and What Is It Used For?” *Investopedia*, Investopedia, 31 July 2023, www.investopedia.com/terms/f/ftp-file-transfer-protocol.asp#:~:text=File%20transfer%20protocol%20(FTP)%20is,order%20to%20execute%20FTP%20transfers.

Postel, J., and J. Reynolds. “RFC 959: File Transfer Protocol.” *IETF Datatracker*, Oct. 1985, datatracker.ietf.org/doc/html/rfc959.

Gallardo, Estefanía García. “What Is File Transfer Protocol (FTP)?” *Built In*, 28 Feb. 2023, builtin.com/software-engineering-perspectives/file-transfer-protocol.

https://en.wikipedia.org/wiki/File_Transfer_Protocol#:~:text=RFC%20959%20%E2%80%93%20(Standard)%20File,Reynolds.

Villanueva, John Carl. “Active vs. Passive FTP Simplified: Understanding FTP Ports.” *JSCAPE*, 16 Nov. 2023, www.jscape.com/blog/active-v-s-passive-ftp-simplified.

Villanueva, John Carl. “SFTP Simplified.” *JSCAPE*, 11 Dec. 2022, www.jscape.com/blog/sftp-simplified.