Centre for Cybersecurity Network Research Project: Remote Control

Aux-User

S5

CFC020223

Contents

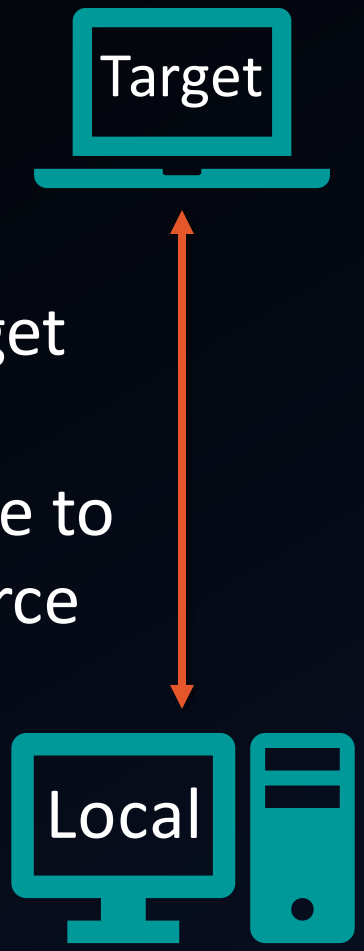
Objective

Purpose:
To automate scanning a target from a remote server while the local machine remains anonymous.

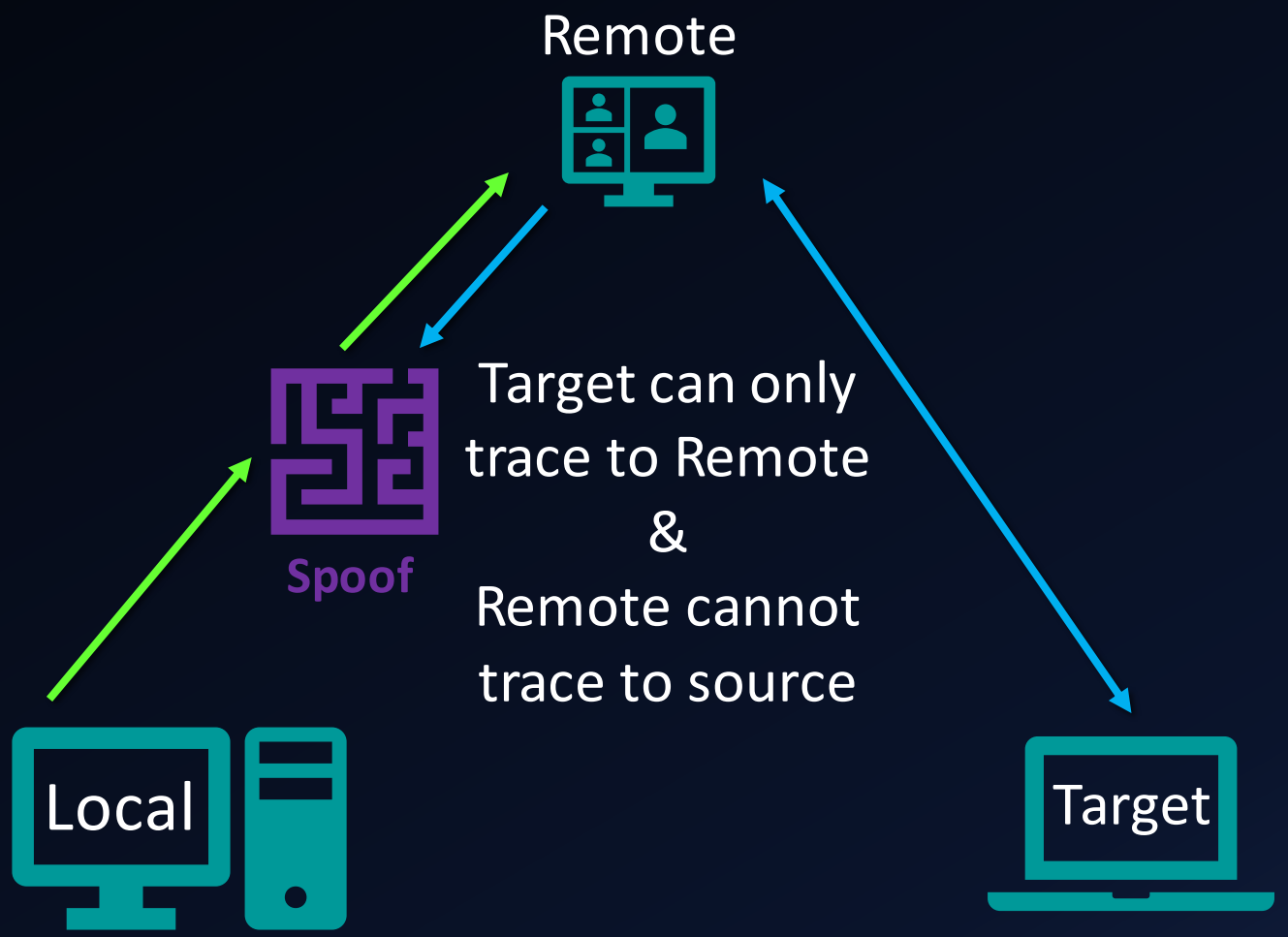
Expected outcome:
- Local IP spoofed to prevent tracing back to source
- Scans done from remote server so any trace leads only to remote server
- Results of scan are saved locally for follow up tasks

Comparison

DIRECT SCAN



SCAN VIA REMOTE SERVER



Script Overview 1 of 2 (Local)

```bash
#!/bin/bash

#This is the start of the script, a standard greeting followed by an
#output telling the user what the script is doing.
echo 'Greetings, User.
Now checking for installed programs before proceeding.'
echo ' '

#These colour codes are for some quality of life enhancements to
#highlight some outputs from the script.
RED='\033[0;31m'
GRN='\033[0;32m'
YLW='\033[0;33m'
BGRN='\033[1;32m'
BCYN='\033[1;36m'
CLR='\033[0m'

#This portion will contain the part of the script that handles the
#checking and preparation of the local host prior to attacking
#the remote system.
#This will check for and if need be, advise how to the following programs
#for the attack: nipe, sshpass, geoip-bin.
#The script will also be stopped from executing further steps, until the
#required programs have been installed.

#This checks for nipe.
echo ' '
CHKNPL=$(find ~ -name nipe.pl)
if [ -z $CHKNPL ]
then
    echo -e "Nipe NOT found!
Please refer to the instructions at ${RED}https://github.com/htrgouvea/nipe${CLR} and run this script again"
    exit
else
    echo -e "Nipe ${GRN}detected${CLR}."
fi
```

```bash
#This checks for sshpass.
CHKSSHP=$(ls /usr/bin | grep sshpass)
if [ -z $CHKSSHP ]
then
    echo -e "sshpass NOT found!
Please install using ${RED}sudo apt-get install sshpass${CLR} and run this script again"
    exit
else
    echo -e "sshpass ${GRN}detected${CLR}."
fi

#This checks for geoip-bin.
CHKGEOIP=$(ls /usr/share | grep GeoIP)
if [ -z $CHKGEOIP ]
then
    echo -e "geoip-bin NOT found!
Please install using ${RED}sudo apt-get install geoip-bin${CLR} and run this script again."
    exit
else
    echo -e "geoip-bin ${GRN}detected${CLR}."
fi
echo ' '

#Once the required programs are confirmed/installed on the host system,
#the script will then begin the next phase which is to spoof the IP and
#confirm that the spoof is active.
echo ' '
echo 'Proceeding to spoof IP...'
cd ~/nipe
#nipe can only be executed from the nipe directory so the script must go
#there first before the following can be executed
sudo perl nipe.pl restart
#restart is used since it stops and restarts the service regardless of current status.

PNPSTAT=$(sudo perl nipe.pl status | grep -i true)
SPFIP=$(sudo perl nipe.pl status | grep -i ip | awk '{print$3}')
#This portion checks if the spoof is active or not.
if [ -z "$PNPSTAT" ]
#Double quotation marks were used to prevent the "[:too many arguments"
#error arising from spaces in the output
then
    echo -e "Spoof ${RED}NOT${CLR} active!
Please check nipe installation and/or network connections and run this script again."
    exit

else
    echo -e "Spoof ${GRN}ACTIVE${CLR}. The world is blind (mostly) to your origin."
    echo "Spoofed IP Address: $SPFIP"
    geoiplookup "$SPFIP"

fi
echo ' '
```

Script Overview 2 of 2 (Remote)

```
#This portion will contain the part of the script that handles communication with the remote server.
#For the purposes of this script we shall assume that the remote server already has whois and nmap.

#The below will prompt the user to input remote login details and the target details for scanning.
echo -e "Please enter ${YLW}remote${CLR} User login"
read REMLOG
echo -e "Please enter ${YLW}remote${CLR} IP"
read REMIP
echo -e "Please enter ${YLW}remote${CLR} User password"
read REMPW
echo -e "Please provide ${RED}TARGET${CLR} IP/Domain to scan"
read VICIP
echo ' '
#After this, there should no longer be a need for manual input from the
#user. The remote scanning and local saving of results will be automated.

#This portion provides the user with the remote server IP address, country and uptime.
echo 'Connecting to Remote Server...'
REMIPTRUE=$(sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" 'curl -s ifconfig.co')
echo "IP Address: $REMIPTRUE"
whois "$REMIPTRUE" | grep -i country | sort | uniq
REMUPT=$(sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" 'uptime')
echo "Uptime: $REMUPT"
echo ' '

#This portion is where the remote server will scan the target for the user.
#whois is scanned first, followed by nmap.
#The respective scan outputs will be saved into a file for further analysis.
echo ' '
echo 'Scanning Target...'
echo "Saving whois data into $VICIP-whois"
sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" "whois $VICIP >> $VICIP-whois"

echo "Saving nmap data into $VICIP-nmap"
sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" "nmap $VICIP -Pn -sV -oN $VICIP-nmap"
echo ' '

#The next few lines will copy the scan results from the remote system to
#the local sytem and delete the files from the former.
echo 'Transferring scan results from remote system...'
sshpass -p "$REMPW" scp "$REMLOG@$REMIP":~/"$VICIP-whois" .
sshpass -p "$REMPW" scp "$REMLOG@$REMIP":~/"$VICIP-nmap" .
sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" "rm $VICIP*"
echo 'Done. The world is blind (mostly) to your passing.'
echo ' '
```

```
#This is the last part of the script. It notifies the user that the
#scan is complete and where to find the files for further work.
#A new directory will be made and the files moved there.
echo -e "${BGRN}Scan complete.${CLR}"
echo 'Target whois and nmap scans have been saved here:'
DTSMP=$(date +%F-%H%M)
mkdir ~/"$VICIP-$DTSMP"
mv "$VICIP"* ~/"$VICIP-$DTSMP"
REPWHO=$(find ~ -name "$VICIP"-whois)
REPNMP=$(find ~ -name "$VICIP"-nmap)
echo -e "${BCYN}$REPWHO${CLR}"
echo -e "${BCYN}$REPNMP${CLR}"
echo ' '
echo 'Farewell, User. Have a nice day.'

#Finish
```

Script Breakdown – Local - Start

```
1   #!/bin/bash
2
3   #This is the start of the script, a standard greeting followed by an
4   #output telling the user what the script is doing.
5   echo 'Greetings, User.
6   Now checking for installed programs before proceeding.'
7   echo ' '
8
9   #These colour codes are for some quality of life enhancements to
10  #highlight some outputs from the script.
11  RED='\033[0;31m'
12  GRN='\033[0;32m'
13  YLW='\033[0;33m'
14  BGRN='\033[1;32m'
15  BCYN='\033[1;36m'
16  CLR='\033[0m'
17
```

This is the start of the script. It consists of a simple greeting and letting the user know what steps the script is taking.
It also loads some codes for colour output that will be used to highlight some details to the user later on.

Script Breakdown – Local – Program Check

```
18  #This portion will contain the part of the script that handles the
19  #checking and preparation of the local host prior to attacking
20  #the remote system.
21  #This will check for and if need be, advise how to the following programs
22  #for the attack: nipe, sshpass, geoip-bin.
23  #The script will also be stopped from executing further steps, until the
24  #required programs have been installed.
25
26  #This checks for nipe.
27  echo ' '
28  CHKNPL=$(find ~ -name nipe.pl)
29  if [ -z $CHKNPL ]
30  then
31      echo -e "Nipe NOT found!
32  Please refer to the instructions at ${RED}https://github.com/htrgouvea/nipe${CLR} and run this script again"
33      exit
34  else
35      echo -e "Nipe ${GRN}detected${CLR}."
36  fi
37
```

This part of the script will check if the local system has the required programs to run the script, namely nipe, sshpass and geoip-bin.

If it does not, it will instruct the user how to do so and the script will terminate.

The check is done with a search / list for the required program name, the result of which is saved into a variable.

The variable is then checked to see if it is null.

If null, it means the program is not installed and the script will terminate.

If not null, the script will check for the next required program.

This portion shows the check for nipe. The checks for sshpass and geoipbin are shown on the next page.

Script Breakdown – Local – Program Check

```
38  #This checks for sshpass.
39  CHKSSHP=$(ls /usr/bin | grep sshpass)
40  if [ -z $CHKSSHP ]
41  then
42      echo -e "sshpass NOT found!
43  Please install using ${RED}sudo apt-get install sshpass${CLR} and run this script again"
44      exit
45  else
46      echo -e "sshpass ${GRN}detected${CLR}."
47  fi
48
49  #This checks for geoip-bin.
50  CHKGEOIP=$(ls /usr/share | grep GeoIP)
51  if [ -z $CHKGEOIP ]
52  then
53      echo -e "geoip-bin NOT found!
54  Please install using ${RED}sudo apt-get install geoip-bin${CLR} and run this script again."
55      exit
56  else
57      echo -e "geoip-bin ${GRN}detected${CLR}."
58  fi
59  echo ' '
60
```

Here are the checks for sshpass and geoip-bin.

As per the check for nipe in the previous page, these checks here will also terminate the script if the programs are not found, while prompting the user to install them before running the script again.

The next page will show the results of a positive and negative search/list hit for program installation, which will be what is matched against the null condition for the check.

There is also a screenshot of the script terminating if it is not able to detect one of the required programs.

Script Breakdown – Program Check Fail

```
┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─$ find ~ -name nipe.pl
/home/kali/nipe/nipe.pl

┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─$ find ~ -name nipez.pl
```

```
┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─$ ls /usr/bin | grep sshpass
sshpass

┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─$ ls /usr/bin | grep sshpiss
```

Results of positive and negative find/list to be checked against null

```
Greetings, User.
Now checking for installed programs before proceeding.
Nipe NOT found!
        Please refer to the instructions at https://github.com/htrgouvea/nipe and run this script again

┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─
```

Null result would terminate the script and instruct the user to install and run the script again.
This check is done another two times for sshpass and geoip-bin

Script Breakdown – Local – Activating Spoof

```
61  #Once the required programs are confirmed/installed on the host system,
62  #the script will then begin the next phase which is to spoof the IP and
63  #confirm that the spoof is active.
64  echo ' '
65  echo 'Proceeding to spoof IP...'
66  cd ~/nipe
67  #nipe can only be executed from the nipe directory so the script must go
68  #there first before the following can be executed
69  sudo perl nipe.pl restart
70  #restart is used since it stops and restarts the service regardless of current status.
71
72  PNPSTAT=$(sudo perl nipe.pl status | grep -i true)
73  SPFIP=$(sudo perl nipe.pl status | grep -i ip | awk '{print$3}')
74  #This portion checks if the spoof is active or not.
75  if [ -z "$PNPSTAT" ]
76  #Double quotation marks were used to prevent the "[:too many arguments"
77  #error arising from spaces in the output
78  then
79      echo -e "Spoof ${RED}NOT${CLR} active!
80  Please check nipe installation and/or network connections and run this script again."
81      exit
82
83  else
84      echo -e "Spoof ${GRN}ACTIVE${CLR}. The world is blind (mostly) to your origin."
85      echo "Spoofed IP Address: $SPFIP"
86      geoiplookup "$SPFIP"
87
88  fi
89  echo ' '
90
```

Once all the programs are confirmed present, the script will then activate the spoofing of the local system IP to prevent any investigation on the remote server tracing the log in back to the local system.

The script will go to the ~/nipe directory since the spoof can only be executed from there and commands will be run as root, so sudo is used as part of the command string.

The restart command is used as it will start or stop/start regardless of current state and there is no harm in refreshing a current spoof.

To check if the spoof is running properly, the script will grep the output of the status command for “true” and save into a variable.

If the variable is null, the script will exit and instruct the user to check installation and network connections.

If the variable is not null, it means the spoof is successful. The spoofed IP address is stored as a variable by grep-ing ‘ip’ and displaying the third column of a true output. This variable is then echoed to display the spoofed IP address, as well as run on geoiplookup to determine the country of the spoofed IP.

The next page will show the results of an active/inactive spoof to show the presence/absence of the word true, as well as a failed spoof and script termination.

Script Breakdown – Spoof Fail

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

[+] Status: true
[+] Ip: 185.129.62.63


┌──(kali㉿kali)-[~/nipe]
└─$ sudo perl nipe.pl status | grep -i true
[+] Status: true
```

```
┌──(kali㉿kali)-[~/nipe]
└─$ sudo perl nipe.pl stop
[sudo] password for kali:

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

[+] Status: false
[+] Ip: 115.[redacted]


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

┌──(kali㉿kali)-[~/nipe]
```

Results of grep for ‘true’ to be checked against null

```
Proceeding to spoof IP
Spoof NOT active!
Please check nipe installation and/or network connections and run this script again.

┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─
```

Null result would terminate the script and instruct the user to make some checks and run the script again

Script Breakdown – Spoof Pass – Local Steps Pass

```
└─$ bash chameleon.sh
Greetings, User.
Now checking for installed programs before proceeding.


Nipe detected.
sshpass detected.
geoip-bin detected.


Proceeding to spoof IP
Spoof ACTIVE. The world is blind (mostly) to your origin.
Spoofed IP Address: 23.129.64.216
GeoIP Country Edition: US, United States
```

If all earlier program checks are successful and the spoof is successfully activated, the result will look like this.

The script will then proceed on to the remote portion.

Script Breakdown – Remote – Accessing Remote Server

```
91  #This portion will contain the part of the script that handles communication with the remote server.
92  #For the purposes of this script we shall assume that the remote server already has whois and nmap.
93
94  #The below will prompt the user to input remote login details and the target details for scanning.
95  echo -e "Please enter ${YLW}remote${CLR} User login"
96  read REMLOG
97  echo -e "Please enter ${YLW}remote${CLR} IP"
98  read REMIP
99  echo -e "Please enter ${YLW}remote${CLR} User password"
100 read REMPW
101 echo -e "Please provide ${RED}TARGET${CLR} IP/Domain to scan"
102 read VICIP
103 echo ' '
104 #After this, there should no longer be a need for manual input from the
105 #user. The remote scanning and local saving of results will be automated.
106
107 #This portion provides the user with the remote server IP address, country and uptime.
108 echo 'Connecting to Remote Server...'
109 REMIPTRUE=$(sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" 'curl -s ifconfig.co')
110 echo "IP Address: $REMIPTRUE"
111 whois "$REMIPTRUE" | grep -i country | sort | uniq
112 REMUPT=$(sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" 'uptime')
113 echo "Uptime: $REMUPT"
114 echo ' '
115
```

This part of the script handles the connection to the remote server.

As the actions on the remote server need to be fully automated, the script will now prompt the user for
i) remote server IP and login credentials
ii) target IP/Domain to scan
These will be stored as variables for the script to execute further commands later.

After this, there should be no further input required from the user.

Commands executed from the remote server are done using sshpass. This allows ssh commands to be easily run from a script. In a single line, sshpass allows the script to log in with the user credentials and execute commands.

To display server IP, country and uptime, the script will first run curl ifconfig.co to confirm IP address.

The result is stored as a variable and run against whois (with some refinements to clean the result for displaying country) and uptime, to display country of the remote server and its uptime respectively.

Script Breakdown – Remote – Remote Server Accessed

```
Please enter remote User login
[redacted]
Please enter remote IP
164.92.189.42
Please enter remote User password
[redacted]
Please provide TARGET IP/Domain to scan
192.168.137.128

Connecting to Remote Server ...
IP Address: 164.92.189.42
Country:        US
Uptime:  17:09:12 up 31 days,  7:51,  1 user,  load average: 0.10, 0.05, 0.01
```

Screenshots showing successful login into the remote server.

The top one shows the results of a user accessing into an external IP address.

The bottom one shows the results of a user accessing into a private IP address. Note that the curl ifconfig will show that network's external IP.

Script Breakdown – Remote – Scanning & Housekeeping

```
116  #This portion is where the remote server will scan the target for the user.
117  #whois is scanned first, followed by nmap.
118  #The respective scan outputs will be saved into a file for further analysis.
119  echo ' '
120  echo 'Scanning Target...'
121  echo "Saving whois data into $VICIP-whois"
122  sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" "whois $VICIP >> $VICIP-whois"
123
124  echo "Saving nmap data into $VICIP-nmap"
125  sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" "nmap $VICIP -Pn -sV -oN $VICIP-nmap"
126  echo ' '
127
128  #The next few lines will copy the scan results from the remote system to
129  #the local sytem and delete the files from the former.
130  echo 'Transferring scan results from remote system...'
131  sshpass -p "$REMPW" scp "$REMLOG@$REMIP":~/"$VICIP-whois" .
132  sshpass -p "$REMPW" scp "$REMLOG@$REMIP":~/"$VICIP-nmap" .
133  sshpass -p "$REMPW" ssh "$REMLOG@$REMIP" "rm $VICIP*"
134  echo 'Done. The world is blind (mostly) to your passing.'
135  echo ' '
136
137  #This is the last part of the script. It notifies the user that the
138  #scan is complete and where to find the files for further work.
139  #A new directory will be made and the files moved there.
140  echo -e "${BGRN}Scan complete.${CLR}"
141  echo 'Target whois and nmap scans have been saved here:'
142  DTSMP=$(date +%F-%H%M)
143  mkdir ~/"$VICIP-$DTSMP"
144  mv "$VICIP"* ~/"$VICIP-$DTSMP"
145  REPWHO=$(find ~ -name "$VICIP"-whois)
146  REPNMP=$(find ~ -name "$VICIP"-nmap)
147  echo -e "${BCYN}$REPWHO${CLR}"
148  echo -e "${BCYN}$REPNMP${CLR}"
149  echo ' '
150  echo 'Farewell, User. Have a nice day.'
151
152  #Finish
```

This part of the script handles the remote scanning of the target from the remote server.

The whois and nmap scans must be done remotely to maintain anonymity, so again, sshpass will be used to execute the commands on the remote server, with the script calling the target IP or Domain from an earlier stored variable for the scans.

The nmap scan settings are -Pn to avoid getting dropped, -sV to pick up any switched ports and –oN for normal output. The whois is a standard command with output saved in a file.

The results of the scan are stored on the remote system, since that is where the scans were run from. The results are named them after the target IP or Domain, followed by nmap or whois. This indicates what kind of scan results and who was scanned.

To bring the results back from the remote server into the local machine for further analysis, scp is used. To bypass password prompts, sshpass is used in conjunction with scp for a seamless transfer.

Once the results are copied over, the originals on the remote server are deleted using sshpass, removing traces of the earlier scans.

Back in the local machine, the script will perform some housekeeping. Using the date command, it saves a timestamp as a variable and uses it, together with the target IP / Domain, to make a directory. The directory name will reflect the who and the when, allowing the user to keep track of multiple scans if they need to.

Lastly, the script will move the scan results into this directory and tell the user where to find the scan results.

chameleon.sh

```
└─$ bash chameleon.sh
Greetings, User.
Now checking for installed programs before proceeding.


Nipe detected.
sshpass detected.
geoip-bin detected.


Proceeding to spoof IP ...
[sudo] password for kali:
Spoof ACTIVE. The world is blind (mostly) to your origin.
Spoofed IP Address: 185.220.103.115
GeoIP Country Edition: US, United States

Please enter remote User login
[redacted]
Please enter remote IP
164.92.189.42
Please enter remote User password
[redacted]
Please provide TARGET IP/Domain to scan
8.8.8.8

Connecting to Remote Server ...
IP Address: 164.92.189.42
Country:        US
Uptime:  15:41:27 up 32 days,  6:23,  0 users,  load average: 0.00, 0.00, 0.00
```

Running script to scan 8.8.8.8

```
Scanning Target ...
Saving whois data into 8.8.8.8-whois
Saving nmap data into 8.8.8.8-nmap
Starting Nmap 7.92 ( https://nmap.org ) at 2023-04-13 15:41 UTC
Nmap scan report for dns.google (8.8.8.8)
Host is up (0.0035s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE    VERSION
53/tcp  open  tcpwrapped
443/tcp open  ssl/https  scaffolding on HTTPServer2
1 service unrecognized despite returning data. If you know the service/version, please submit t
cgi-bin/submit.cgi?new-service :
SF-Port443-TCP:V=7.92%T=SSL%I=7%D=4/13%Time=643822CE%P=x86_64-pc-linux-gnu
SF:%r(Help,6B0,"HTTP/1\.0\x20400\x20Bad\x20Request\r\nContent-Type:\x20tex
SF:t/html;\x20charset=UTF-8\r\nReferrer-Policy:\x20no-referrer\r\nContent-
SF:Length:\x201555\r\nDate:\x20Thu,\x2013\x20Apr\x202023\x2015:42:06\x20GM
SF:T\r\n\r\n<!DOCTYPE\x20html>\n<html\x20lang=en>\n\x20\x20<meta\x20charse
SF:t=utf-8>\n\x20\x20<meta\x20name=viewport\x20content=\"initial-scale=1,\
SF:x20minimum-scale=1,\x20width=device-width\">\n\x20\x20<title>Error\x204
SF:00\x20\(Bad\x20Request\)!!1</title>\n\x20\x20<style>\n\x20\x20\x20\x20\
SF:*{margin:0;padding:0}html,code{font:15px/22px\x20arial,sans-serif}html{
SF:background:#fff;color:#222;padding:15px}body{margin:7%\x20auto\x200;max
SF:-width:390px;min-height:180px;padding:30px\x200\x2015px}\*\x20>\x20body
SF:{background:url\(//www\.google\.com/images/errors/robot\.png\)\x20100%\
SF:x205px\x20no-repeat;padding-right:205px}p{margin:11px\x200\x2022px;over
SF:flow:hidden}ins{color:#777;text-decoration:none}a\x20img{border:0}@medi
SF:a\x20screen\x20and\x20\(max-width:772px\){body{background:none;margin-t
SF:op:0;max-width:none;padding-right:0}}#logo{background:url\(//www\.googl
SF:e\.com/images/branding/googlelogo/1x");

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 69.12 seconds

Transferring scan results from remote system ...
Done. The world is blind (mostly) to your passing.

Scan complete.
Target whois and nmap scans have been saved here:
/home/kali/8.8.8.8-2023-04-13-1143/8.8.8.8-whois
/home/kali/8.8.8.8-2023-04-13-1143/8.8.8.8-nmap

Farewell, User. Have a nice day.

┌──(kali㉿kali)-[~/cfc0202/NWRProj]
```

chameleon.sh

```
┌──(kali㉿kali)-[~/8.8.8.8-2023-04-13-1143]
└─$ ls
8.8.8.8-nmap  8.8.8.8-whois

┌──(kali㉿kali)-[~/8.8.8.8-2023-04-13-1143]
└─$ cat 8.8.8.8-nmap
# Nmap 7.92 scan initiated Thu Apr 13 15:41:38 2023 as: nmap -Pn -sV -oN 8.8.8.8-nmap 8.8.8.8
Nmap scan report for dns.google (8.8.8.8)
Host is up (0.0035s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE    VERSION
53/tcp  open  tcpwrapped
443/tcp open  ssl/https  scaffolding on HTTPServer2
1 service unrecognized despite returning data. If you know the service/version, please submit th
cgi-bin/submit.cgi?new-service :
SF-Port443-TCP:V=7.92%T=SSL%I=7%D=4/13%Time=643822CE%P=x86_64-pc-linux-gnu
SF:%r(Help,6B0,"HTTP/1\.0\x20400\x20Bad\x20Request\r\nContent-Type:\x20tex
SF:t/html;\x20charset=UTF-8\r\nReferrer-Policy:\x20no-referrer\r\nContent-
SF:Length:\x201555\r\nDate:\x20Thu,\x2013\x20Apr\x202023\x2015:42:06\x20GM
SF:T\r\n\r\n<!DOCTYPE\x20html>\n<html\x20lang=en>\n\x20\x20<meta\x20charse
SF:t=utf-8>\n\x20\x20<meta\x20name=viewport\x20content=\"initial-scale=1,\
SF:x20minimum-scale=1,\x20width=device-width\">\n\x20\x20<title>Error\x204
SF:00\x20\(Bad\x20Request\)!!1</title>\n\x20\x20<style>\n\x20\x20\x20\x20\
SF:*{margin:0;padding:0}html,code{font:15px/22px\x20arial,sans-serif}html{
SF:background:#fff;color:#222;padding:15px}body{margin:7%\x20auto\x200;max
SF:-width:390px;min-height:180px;padding:30px\x200\x2015px}\*\x20>\x20body
SF:{background:url\(//www\.google\.com/images/errors/robot\.png\)\x20100%\
SF:x205px\x20no-repeat;padding-right:205px}p{margin:11px\x200\x2022px;over
SF:flow:hidden}ins{color:#777;text-decoration:none}a\x20img{border:0}@medi
```

```
┌──(kali㉿kali)-[~/8.8.8.8-2023-04-13-1143]
└─$ cat 8.8.8.8-whois

#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/resources/registry/whois/tou/
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2023, American Registry for Internet Numbers, Ltd.
#



# start

NetRange:       8.0.0.0 - 8.127.255.255
CIDR:           8.0.0.0/9
NetName:        LVLT-ORG-8-8
NetHandle:      NET-8-0-0-0-1
Parent:         NET8 (NET-8-0-0-0-0)
NetType:        Direct Allocation
OriginAS:
Organization:   Level 3 Parent, LLC (LPL-141)
RegDate:        1992-12-01
Updated:        2018-04-23
Ref:            https://rdap.arin.net/registry/ip/8.0.0.0
```

Location and contents of saved scan results

chameleon.sh

```
┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─$ bash chameleon.sh
Greetings, User.
Now checking for installed programs before proceeding.


Nipe detected.
sshpass detected.
geoip-bin detected.


Proceeding to spoof IP ...
Spoof ACTIVE. The world is blind (mostly) to your origin.
Spoofed IP Address: 185.220.102.246
GeoIP Country Edition: DE, Germany

Please enter remote User login
[redacted]
Please enter remote IP
192.168.137.129
Please enter remote User password
[redacted]
Please provide TARGET IP/Domain to scan
192.168.137.128

Connecting to Remote Server ...
IP Address: 115.[redacted]
country:        SG
country:        ZZ
Uptime:  16:53:10 up 19:21,  1 user,  load average: 0.00, 0.00, 0.00
```

```
Scanning Target ...
Saving whois data into 192.168.137.128-whois
Saving nmap data into 192.168.137.128-nmap
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-12 16:53 UTC
Nmap scan report for 192.168.137.128
Host is up (0.0021s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
22/tcp open  ssh     OpenSSH 9.2p1 Debian 2 (protocol 2.0)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.41 seconds

Transferring scan results from remote system ...
Done. The world is blind (mostly) to your passing.

Scan complete.
Target whois and nmap scans have been saved here:
/home/kali/192.168.137.128-2023-04-12-1253/192.168.137.128-whois
/home/kali/192.168.137.128-2023-04-12-1253/192.168.137.128-nmap

Farewell, User. Have a nice day.
```

Running script to scan a private IP on my network
(to avoid potentially painful legal consequences for the use of nmap)

chameleon.sh

```
┌──(kali㉿kali)-[~/192.168.137.128-2023-04-12-1253]
└─$ ls
192.168.137.128-nmap  192.168.137.128-whois

┌──(kali㉿kali)-[~/192.168.137.128-2023-04-12-1253]
└─$ cat 192.168.137.128-nmap
# Nmap 7.80 scan initiated Wed Apr 12 16:53:12 2023 as: nmap -Pn -sV -oN 192.168.137.128-nmap 192.168.137.128
Nmap scan report for 192.168.137.128
Host is up (0.0021s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
22/tcp open  ssh     OpenSSH 9.2p1 Debian 2 (protocol 2.0)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Wed Apr 12 16:53:12 2023 -- 1 IP address (1 host up) scanned in 0.41 seconds
```

```
┌──(kali㉿kali)-[~/192.168.137.128-2023-04-12-1253]
└─$ cat 192.168.137.128-whois

#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/resources/registry/whois/tou/
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2023, American Registry for Internet Numbers, Ltd.
#


NetRange:       192.168.0.0 - 192.168.255.255
CIDR:           192.168.0.0/16
NetName:        PRIVATE-ADDRESS-CBLK-RFC1918-IANA-RESERVED
NetHandle:      NET-192-168-0-0-1
Parent:         NET192 (NET-192-0-0-0-0)
NetType:        IANA Special Use
OriginAS:
Organization:   Internet Assigned Numbers Authority (IANA)
RegDate:        1994-03-15
Updated:        2013-08-30
Comment:        These addresses are in use by many millions of independer
 connected to a home gateway, and are automatically configured in hundrec
private context  and traffic that needs to cross the Internet will need t
```

Location and contents of saved scan results

chameleon.sh

```
┌──(kali㉿kali)-[~/cfc0202/NWRProj]
└─$ bash chameleon.sh
Greetings, User.
Now checking for installed programs before proceeding.


Nipe detected.
sshpass detected.
geoip-bin detected.


Proceeding to spoof IP ...
Spoof ACTIVE. The world is blind (mostly) to your origin.
Spoofed IP Address: 185.220.102.251
GeoIP Country Edition: DE, Germany

Please enter remote User login
[redacted]
Please enter remote IP
192.168.137.129
Please enter remote User password
[redacted]
Please provide TARGET IP/Domain to scan
nmap.scanme.org

Connecting to Remote Server ...
IP Address: 115.[redacted]
country:        SG
country:        ZZ
Uptime:  16:41:12 up 19:09,  1 user,  load average: 0.11, 0.03, 0.01
```

```
Scanning Target ...
Saving whois data into nmap.scanme.org-whois
Saving nmap data into nmap.scanme.org-nmap
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-12 16:41 UTC
Nmap scan report for nmap.scanme.org (45.33.32.156)
Host is up (0.21s latency).
Other addresses for nmap.scanme.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
rDNS record for 45.33.32.156: scanme.nmap.org
Not shown: 996 filtered ports
PORT      STATE SERVICE    VERSION
22/tcp    open  ssh        OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http       Apache httpd 2.4.7 ((Ubuntu))
9929/tcp  open  nping-echo Nping echo
31337/tcp open  tcpwrapped
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 65.20 seconds

Transferring scan results from remote system ...
Done. The world is blind (mostly) to your passing.

Scan complete.
Target whois and nmap scans have been saved here:
/home/kali/nmap.scanme.org-2023-04-12-1242/nmap.scanme.org-whois
/home/kali/nmap.scanme.org-2023-04-12-1242/nmap.scanme.org-nmap

Farewell, User. Have a nice day.
```

Running script to scan nmap.scanme.org
Since the requirement is to scan IP and Domain, the script must work with both numbers and letters keyed in when the user provides the target details

chameleon.sh

```
┌──(kali㉿kali)-[~/nmap.scanme.org-2023-04-12-1242]
└─$ ls
nmap.scanme.org-nmap  nmap.scanme.org-whois

┌──(kali㉿kali)-[~/nmap.scanme.org-2023-04-12-1242]
└─$ cat nmap.scanme.org-nmap
# Nmap 7.80 scan initiated Wed Apr 12 16:41:15 2023 as: nmap -Pn -sV -oN nmap.scanme.org-nmap nmap.scanme.org
Nmap scan report for nmap.scanme.org (45.33.32.156)
Host is up (0.21s latency).
Other addresses for nmap.scanme.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
rDNS record for 45.33.32.156: scanme.nmap.org
Not shown: 996 filtered ports
PORT      STATE SERVICE    VERSION
22/tcp    open  ssh        OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http       Apache httpd 2.4.7 ((Ubuntu))
9929/tcp  open  nping-echo Nping echo
31337/tcp open  tcpwrapped
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Wed Apr 12 16:42:20 2023 -- 1 IP address (1 host up) scanned in 65.20 seconds
```

Location and contents of saved scan results

```
┌──(kali㉿kali)-[~/nmap.scanme.org-2023-04-12-1242]
└─$ cat nmap.scanme.org-whois
Malformed request.
>>> Last update of WHOIS database: 2023-04-12T16:41:14Z <<<

Terms of Use: Access to Public Interest Registry WHOIS information is provided to assist persons in determining the contents of a domain nam
e registration record in the Public Interest Registry registry database. The data in this record is provided by Public Interest Registry for
 informational purposes only, and Public Interest Registry does not guarantee its accuracy. This service is intended only for query-based ac
cess. You agree that you will use this data only for lawful purposes and that, under no circumstances will you use this data to (a) allow, e
nable, or otherwise support the transmission by e-mail, telephone, or facsimile of mass unsolicited, commercial advertising or solicitations
 to entities other than the data recipient's own existing customers; or (b) enable high volume, automated, electronic processes that send qu
eries or data to the systems of Registry Operator, a Registrar, or Identity Digital except as reasonably necessary to register domain names
or modify existing registrations. All rights reserved. Public Interest Registry reserves the right to modify these terms at any time. By sub
mitting this query, you agree to abide by this policy.  The Registrar of Record identified in this output may have an RDDS service that can
be queried for additional information on how to contact the Registrant, Admin, or Tech contact of the queried domain name.
```

Areas for Improvement

Saw in a video that the presenter used ipcalc to get the network range of a particular IP address.

Since we are already scanning a target, could we not use ipcalc to get more details about the target, such as the details of the target network?

In the video, the presenter mentioned pulling the figures for the network range and scanning it.

However, for the purposes of this script, this might not be so practical. Scanning a range of addresses as well as the ports associated with each address will take up a lot of time.

Given that we should remain anonymous, a high level of network activity could draw unwanted attention from the target or remote server.

```
┌──(kali㉿kali)-[~/nipe]
└─$ ipcalc 192.168.137.128
Address:   192.168.137.128      11000000.10101000.10001001. 10000000
Netmask:   255.255.255.0 = 24   11111111.11111111.11111111. 00000000
Wildcard:  0.0.0.255            00000000.00000000.00000000. 11111111
⇒
Network:   192.168.137.0/24     11000000.10101000.10001001. 00000000
HostMin:   192.168.137.1        11000000.10101000.10001001. 00000001
HostMax:   192.168.137.254      11000000.10101000.10001001. 11111110
Broadcast: 192.168.137.255      11000000.10101000.10001001. 11111111
Hosts/Net: 254                   Class C, Private Internet


┌──(kali㉿kali)-[~/nipe]
└─$ ipcalc 192.168.137.128 | grep Network: | awk '{print$2}'
192.168.137.0/24
```

grep-ed, awk-ed and ready to be assigned to a variable for a script

References

- nipe
  https://github.com/htrgouvea/nipe
- Ipcalc (at 3.27)
  https://youtu.be/FKVsz_2IWJs
- sshpass
  https://www.youtube.com/watch?v=_TP-YnX8f50
  https://www.vpsserver.com/scp-password/#table-of-contents-6
- Coloured text in script
  https://stackoverflow.com/questions/5947742/how-to-change-the-output-color-of-echo-in-linux

THANK YOU