Bash, round 2...Fight!

Lets pick up were we left off in the previous exercise.

You should have a .txt of your live IP’s that you scanned for. Mine were stored in TargetList.txt.

```
┌──(chris㉿kali)-[~]
└─$ cat TargetList.txt

10.0.0.101:
10.0.0.102:
10.0.0.103:
10.0.0.106:
10.0.0.115:
10.0.0.123:
10.0.0.136:
```

Go ahead and do one final clean up and remove the “:”. Once done just do a quick check using “cat”.

```
┌──(chris㉿kali)-[~]
└─$ vim TargetList.txt

┌──(chris㉿kali)-[~]
└─$ cat TargetList.txt

10.0.0.101
10.0.0.102
10.0.0.103
10.0.0.106
10.0.0.115
10.0.0.123
10.0.0.136
```

We are going to this time incorporate encryption in our bash script. First though lets use nmap to check out our network a bit.

Go ahead and use one of the active IP addressess and simply “nmap address” and see what we get.

```
┌──(chris㉿kali)-[~]
└─$ nmap 10.0.0.220
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-28 08:53 MDT
Nmap scan report for 10.0.0.220
Host is up (0.011s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT      STATE SERVICE
49152/tcp open  unknown
49155/tcp open  unknown
62078/tcp open  iphone-sync

Nmap done: 1 IP address (1 host up) scanned in 11.54 seconds
```

OK, good news. One of the TargetList.txt IP’s returned results as we expected. Lets now keep this going.

Let’s now target the entire list in one shot. To do this we use “-iL” <file name>, so we will do “nmap -iL TargetList.txt

```
┌──(chris㉿kali)-[~]
└─$ nmap -iL TargetList.txt
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-28 09:14 MDT
Nmap scan report for 10.0.0.101
Host is up (0.0100s latency).
All 1000 scanned ports on 10.0.0.101 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)

Nmap scan report for 10.0.0.102
Host is up (0.013s latency).
All 1000 scanned ports on 10.0.0.102 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)

Nmap scan report for 10.0.0.103
Host is up (0.017s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE
49153/tcp open  unknown
```

Excellent, ok lets keep this going and now repeat the command but this time send it to a .txt file.

```
┌──(chris㉿kali)-[~]
└─$ nmap -iL TargetList.txt -p 1-1024 > home-lan-nmap-scan.txt
```

Ok, so now we scanned our home network and saved it in a .txt. Lets now make sure nobody can see it by encrypting using pgp.

Also, lets do this using a bash script to exercise that skill set a bit more.

Lets start with using VIM or Nano (dealers choice) for the file recon2.sh

Lets get a little fancy this time and we will have the script tell us its ready to encrypt and ask the file name.

We then take the user input and name it “file” to then tell pgp encrypt the “file”.

Lastly, once done, it tells us the file has been encrypted.

```
GNU nano 7.2                                    recon2.sh
#!/usr/bin/env bash
#apt-get install pgp
#
echo "Welcome, I am ready to encrypt a file/folder"
echo "Enter the Exact File Name with extension"
read -r file
pgp -c $file
echo "I have encrypted the file!"
##decrypt.command "gpg -d filename.gpg > filename"
#gpg --decrypt filename.txt.gpg (command line example) clear
```

Lets go ahead then and install pgp using “sudo apt-get install pgp”.

```
└─$ sudo apt-get install pgp
[sudo] password for chris:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
Note, selecting 'pgpgpg' instead of 'pgp'
The following NEW packages will be installed:
  pgpgpg
0 upgraded, 1 newly installed, 0 to remove and 509 not upgraded.
```

Next, execute our bash script “./recon2.sh” and follow along with our script and we now have encrypted our home network scan file.

```
┌──(chris㉿kali)-[~]
└─$ ./recon2.sh
Welcome, I am ready to encrypt a file/folder
Enter the Exact File Name with extension
home-lan-nmap-scan.txt
gpg: keybox '/home/chris/.gnupg/pubring.kbx' created
I have encrypted the file!
```

Check using “ls -l” and we see the encrypted file ending in .txt.pgp

```
┌──(chris㉿kali)-[~]
└─$ ls -l
total 72
drwxr-xr-x 2 chris   chris   4096 Mar 21 16:09 Desktop
drwxr-xr-x 2 chris   chris   4096 Mar 21 16:09 Documents
drwxr-xr-x 2 chris   chris   4096 Mar 21 16:09 Downloads
drwxr-xr-x 2 chris   chris   4096 Mar 21 16:09 Music
drwxr-xr-x 2 chris   chris   4096 Mar 28 14:25 Pictures
drwxr-xr-x 2 chris   chris   4096 Mar 21 16:09 Public
-rw-r--r-- 1 chris   chris    255 Mar 28 08:32 TargetList.txt
drwxr-xr-x 2 chris   chris   4096 Mar 21 16:09 Templates
drwxr-xr-x 2 chris   chris   4096 Mar 21 16:09 Videos
-rw-r--r-- 1 chris   chris   4272 Mar 28 09:18 home-lan-nmap-scan.txt
-rw-r--r-- 1 chris   chris    753 Mar 28 11:53 home-lan-nmap-scan.txt.pgp
```

Now, to decrypt all we need to do is use the command “gpg –decrypt <file name>”.
Enter the password when prompted and if correct it will then decrypt and output the contents of the file.

```
┌──(chris㉿kali)-[~]
└─$ gpg --decrypt home-lan-nmap-scan.txt.pgp
gpg: AES256.CFB encrypted data
gpg: encrypted with 1 passphrase
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-28 09:16 MDT
Nmap scan report for 10.0.0.101
Host is up (0.037s latency).
All 1024 scanned ports on 10.0.0.101 are in ignored states.
Not shown: 1024 closed tcp ports (conn-refused)

Nmap scan report for 10.0.0.102
Host is up (0.019s latency).
All 1024 scanned ports on 10.0.0.102 are in ignored states.
Not shown: 982 closed tcp ports (conn-refused), 42 filtered tcp ports (no-res
ponse)
```

Ok, so that worked well and easy enough to create another bash script for decryption. Though, lets kick it up a notch and just have one script to rule them all…

```
GNU nano 7.2                                recon3.sh
#!/usr/bin/env bash
#apt-get install pgp
#
read -p "Welcome, would you like to decrypt(d) or encrypt(e) a file/folder (d/e)?" crypt

if [ "$crypt" = "e" ]; then
        echo "Enter the Exact File Name with extension"
        read -r file
        pgp -c $file
        echo "I have encpryted the file!"
elif [ "$crypt" = "d" ]; then
        echo "Enter the Exact File Name with extension"
        read -r file
        gpg -d $file
        echo "I Have decrypted the file!"
else
        echo "Error: Invalid input."
        exit 1
fi
```

The key is will be understanding if/then as well as else if (elif)/then statements.

We start with asking decrypt or encrypt. After that its fairly straight forward. We also added a portion to the script in case of a typo/mistake which will then terminate the script.

Ok, lets see how we did…

Decrypt – Check

```
┌──(chris㉿kali)-[~]
└─$ ./recon3.sh
Welcome, would you like to decrypt(d) or encrypt(e) a file/folder (d/e)?d
Enter the Exact File Name with extension
home-lan-nmap-scan.txt.pgp
gpg: AES256.CFB encrypted data
gpg: encrypted with 1 passphrase
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-28 09:16 MDT
Nmap scan report for 10.0.0.101
Host is up (0.037s latency).
All 1024 scanned ports on 10.0.0.101 are in ignored states.
Not shown: 1024 closed tcp ports (conn-refused)
```

Encrypt – Check

```
┌──(chris㉿kali)-[~]
└─$ ./recon3.sh
Welcome, would you like to decrypt(d) or encrypt(e) a file/folder (d/e)?e
Enter the Exact File Name with extension
home-lan-nmap-scan.txt
File 'home-lan-nmap-scan.txt.pgp' exists. Overwrite? (y/N) y
I have encpryted the file!
```

Typo – Check

```
┌──(chris㉿kali)-[~]
└─$ ./recon3.sh
Welcome, would you like to decrypt(d) or encrypt(e) a file/folder (d/e)?f
Error: Invalid input.
```

Great Success!