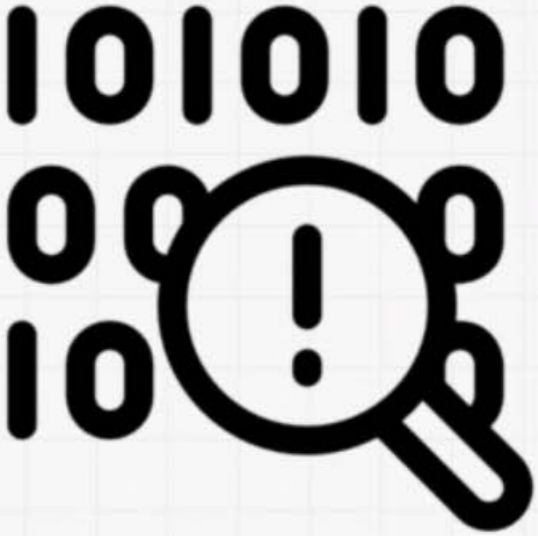
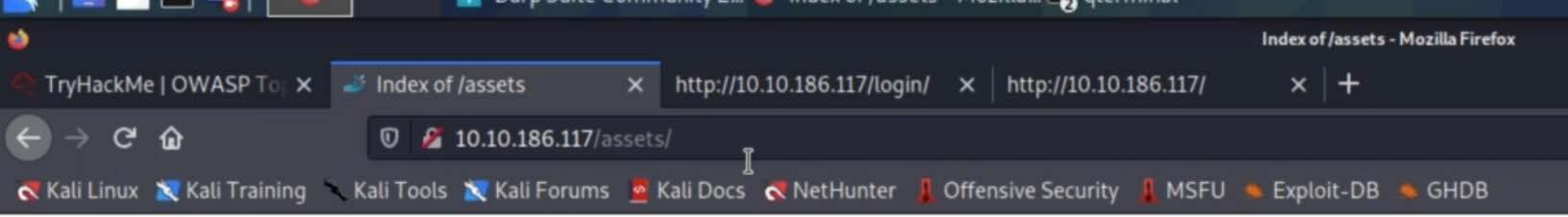


Sensitive Data Exposure

```
<div class="white-box">
  <div class="quote">
    <i class="fa fa-quote-left fa-3x"></i>
    <p class="random-quote"> <span id="text"></span></p>
  </div>
  <div class="random-author">- <span id="author"></span>
  </div>
  <div class= "buttons">
 <!-- Hey, I put the passwords file in /data directory. -->
<a id="tweet" href=""https://twitter.com/intent/tweet"" title="Tweet
this!"><button class= "button"><i class= "fa
fa-twitter"></i></button></a>
```



Index of /assets

Name	Last modified	Size	Description
Parent Directory		-	
css/	2020-07-14 17:52	-	
fonts/	2020-07-14 15:42	-	
images/	2020-07-14 15:42	-	
js/	2020-07-14 15:52	-	
php/	2020-07-14 15:42	-	
webapp.db	2020-07-14 17:52	28K	

Apache/2.4.29 (Ubuntu) Server at 10.10.186.117 Port 80

File Actions Edit View Help

```
┌──(mrhacker㉿kali)-[~]
└─$ cd Downloads

┌──(mrhacker㉿kali)-[~/Downloads]
└─$ ls
cacert.der  printmrhacker.ovpn  webapp.db

┌──(mrhacker㉿kali)-[~/Downloads]
└─$ sqlite3 webapp.db
```

```
┌──(mrhacker㉿kali)-[~]
└─$ cd Downloads

┌──(mrhacker㉿kali)-[~/Downloads]
└─$ ls
cacert.der  printmrhacker.ovpn  webapp.db

┌──(mrhacker㉿kali)-[~/Downloads]
└─$ sqlite3 webapp.db
SQLite version 3.34.1 2021-01-20 14:10:07
Enter ".help" for usage hints.
sqlite> .tables
sessions  users
sqlite> SELECT * FROM users;
4413096d9c933359b898b6202288a650|admin|6eea9b7ef19179a06954edd0f6c05ceb|1
23023b67a32488588db1e28579ced7ec|Bob|ad0234829205b9033196ba818f7a872b|1
4e8423b514eef575394ff78caed3254d|Alice|268b38ca7b84f44fa0a6cdc86e6301e0|0
sqlite>
```

Title	IP Address	Expires
Sense and Sensitivity	10.10.186.117	17m 19s

? Add 1 hour Terminate

Task 11 [Severity 3] Sensitive Data Exposure (Challenge)

Woop woop! Your answer is correct.

It's now time to put what you've learnt into practice!

Have a look around the webapp. The developer has left themselves a note indicating that there is sensitive data in a specific directory.

What is the name of the mentioned directory?

/assets — Correct Answer — Hint

Navigate to the directory you found in question one. What file stands out as being likely to contain sensitive data?

webapp.db — Correct Answer

Use the supporting material to access the sensitive data. What is the password hash of the admin user?

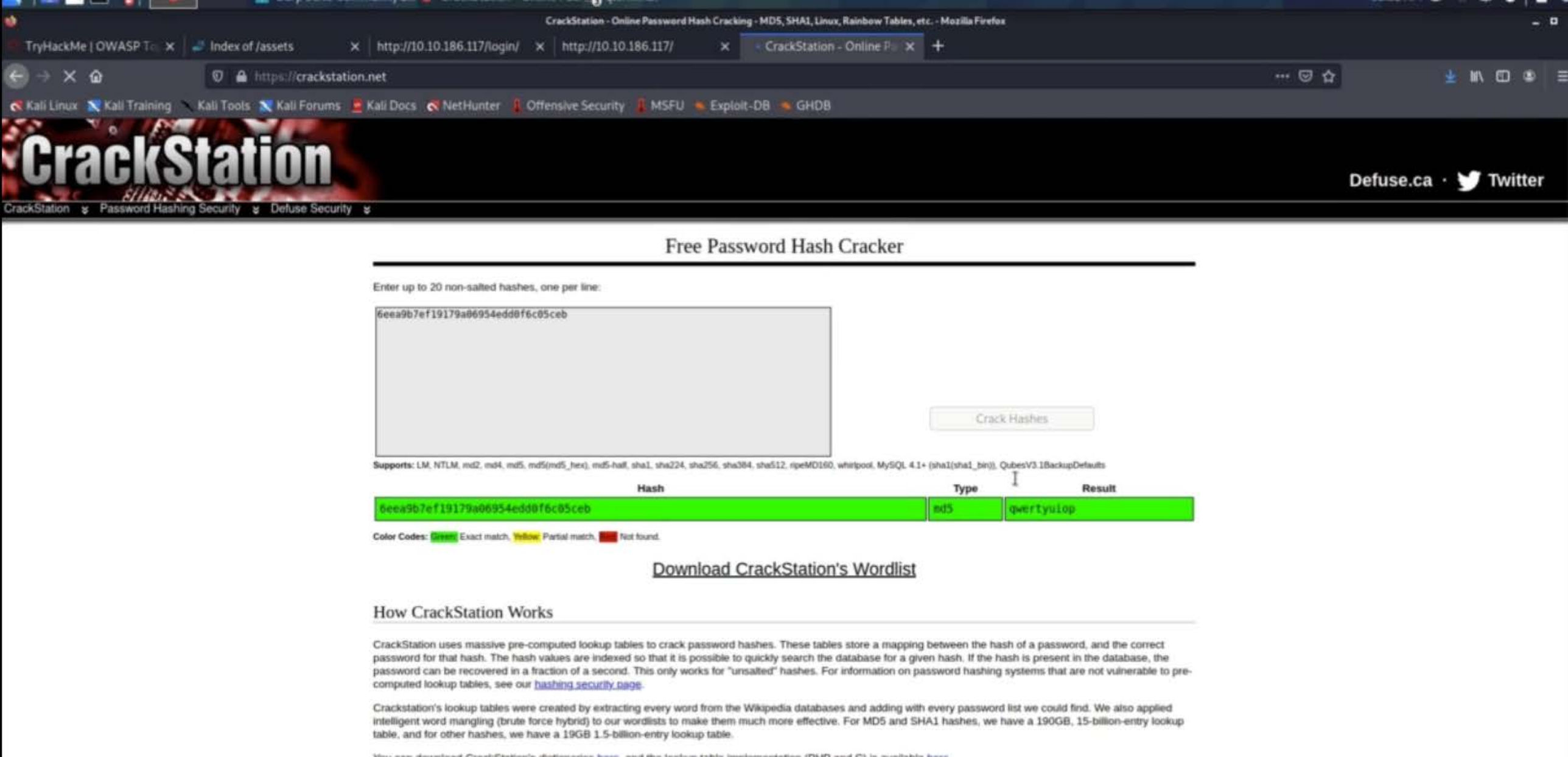
6eea9b7ef19179a06954edd0f6c05ceb — Correct Answer

Crack the hash.
What is the admin's plaintext password?

Answer format: ********** — Submit — Hint

Login as the admin. What is the flag?

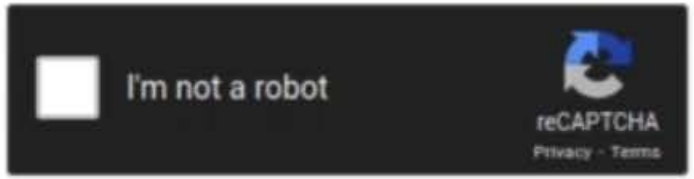
Answer format: ***{**********} — Submit


CrackStation - Online Password Hash Cracking - MD5, SHA1, Linux, Rainbow Tables, etc. - Mozilla Firefox
TryHackMe | OWASP T
Index of /assets
http://10.10.186.117/login/
http://10.10.186.117/
CrackStation - Online P
https://crackstation.net
Kali Linux
Kali Training
Kali Tools
Kali Forums
Kali Docs
NetHunter
Offensive Security
MSFU
Exploit-DB
GHDB
CrackStation
Defuse.ca · Twitter
CrackStation
Password Hashing Security
Defuse Security
Free Password Hash Cracker
Enter up to 20 non-salted hashes, one per line:
6eea9b7ef19179a06954edd0f6c05ceb
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
6eea9b7ef19179a06954edd0f6c05ceb
md5
qwertyuiop
Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.
Download CrackStation's Wordlist
How CrackStation Works
CrackStation uses massive pre-computed lookup tables to crack password hashes. These tables store a mapping between the hash of a password, and the correct password for that hash. The hash values are indexed so that it is possible to quickly search the database for a given hash. If the hash is present in the database, the password can be recovered in a fraction of a second. This only works for "unsalted" hashes. For information on password hashing systems that are not vulnerable to pre-computed lookup tables, see our hashing security page.
Crackstation's lookup tables were created by extracting every word from the Wikipedia databases and adding with every password list we could find. We also applied intelligent word mangling (brute force hybrid) to our wordlists to make them much more effective. For MD5 and SHA1 hashes, we have a 190GB, 15-billion-entry lookup table, and for other hashes, we have a 19GB 1.5-billion-entry lookup table.

Free Password Hash Cracker

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

```
6eea9b7ef19179a06954edd0f6c05ceb
```



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
6eea9b7ef19179a06954edd0f6c05ceb	md5	qwertyuiop

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Download CrackStation's Wordlist

How CrackStation Works

CrackStation uses massive pre-computed lookup tables to crack password hashes. These tables store a mapping between the hash of a password, and the correct password for that hash. The hash values are indexed so that it is possible to quickly search the database for a given hash. If the hash is present in the database, the password can be recovered in a fraction of a second. This only works for "unsalted" hashes. For information on password hashing systems that are not vulnerable to pre-computed lookup tables, see our hashing security page.

Crackstation's lookup tables were created by extracting every word from the Wikipedia databases and adding with every password list we could find. We also applied intelligent word mangling (brute force hybrid) to our wordlists to make them much more effective. For MD5 and SHA1 hashes, we have a 190GB, 15-billion-entry lookup