crackthehash

Resolviendo la máquina Crack the hash

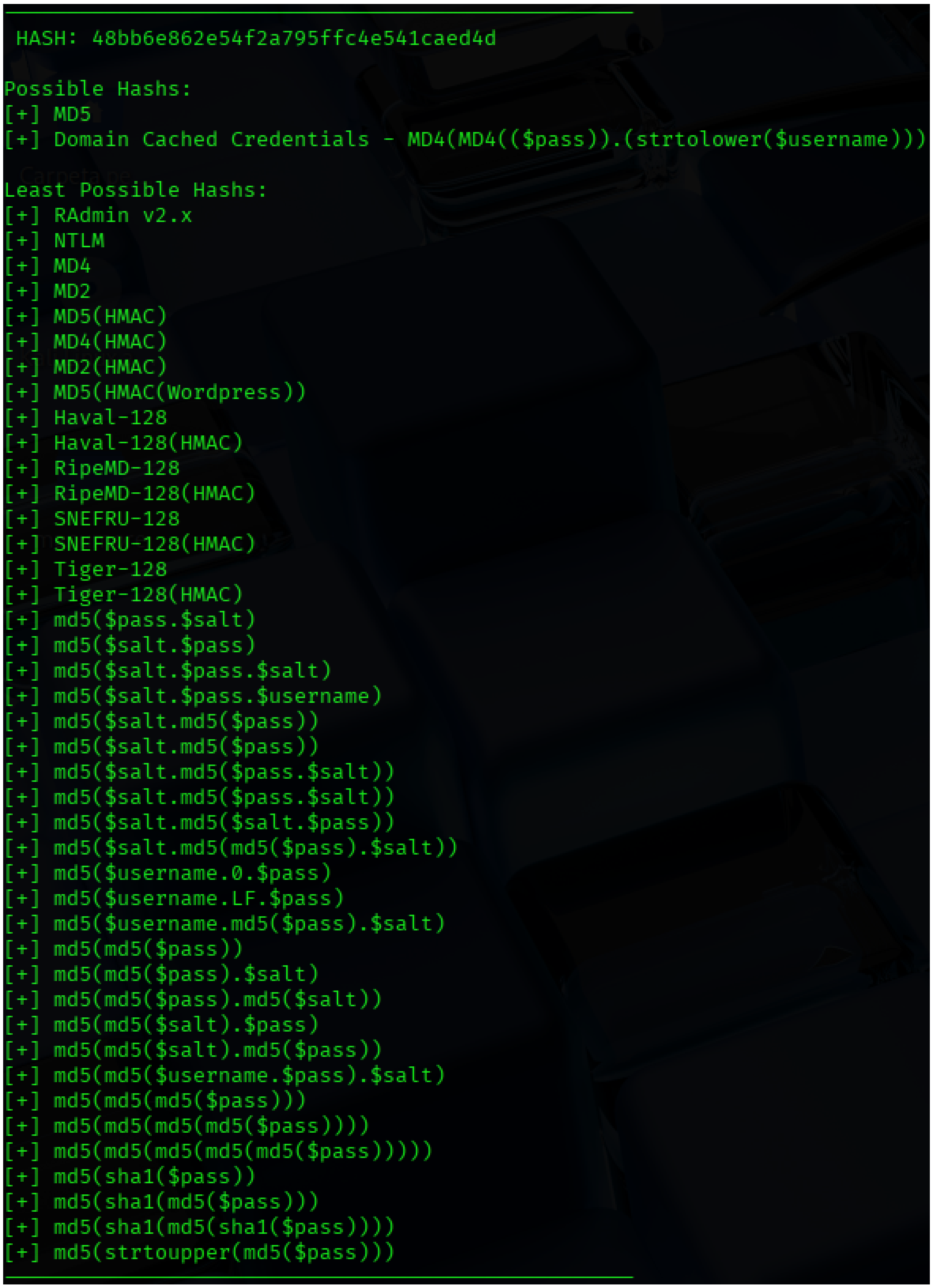
> En esta publicación, comparto cómo resolví la máquina **Crack the hash** de TryHackMe.

Level 1

Hash 1 - MD5

Hash: `48bb6e862e54f2a795ffc4e541caed4d`.

Se usa la herramienta `hash-identifier` para averiguar el *hash*.

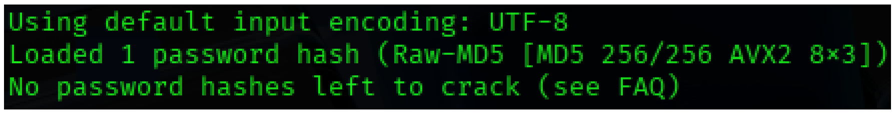


```
 HASH: 48bb6e862e54f2a795ffc4e541caed4d

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
[+] MD5(HMAC)
[+] MD4(HMAC)
[+] MD2(HMAC)
[+] MD5(HMAC(Wordpress))
[+] Haval-128
[+] Haval-128(HMAC)
[+] RipeMD-128
[+] RipeMD-128(HMAC)
[+] SNEFRU-128
[+] SNEFRU-128(HMAC)
[+] Tiger-128
[+] Tiger-128(HMAC)
[+] md5($pass.$salt)
[+] md5($salt.$pass)
[+] md5($salt.$pass.$salt)
[+] md5($salt.$pass.$username)
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($salt.$pass))
[+] md5($salt.md5(md5($pass).$salt))
[+] md5($username.0.$pass)
[+] md5($username.LF.$pass)
[+] md5($username.md5($pass).$salt)
[+] md5(md5($pass))
[+] md5(md5($pass).$salt)
[+] md5(md5($pass).md5($salt))
[+] md5(md5($salt).$pass)
[+] md5(md5($salt).md5($pass))
[+] md5(md5($username.$pass).$salt)
[+] md5(md5(md5($pass)))
[+] md5(md5(md5(md5($pass))))
[+] md5(md5(md5(md5(md5($pass)))))
[+] md5(sha1($pass))
[+] md5(sha1(md5($pass)))
[+] md5(sha1(md5(sha1($pass))))
[+] md5(strtoupper(md5($pass)))
```

Se usa `john` para realizar fuerza bruta.

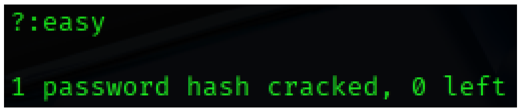
`vim md5_hash.txt`

`john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt md5_hash.txt`



```
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8×3])
No password hashes left to crack (see FAQ)
```

`john --show --format=raw-md5 md5_hash.txt`

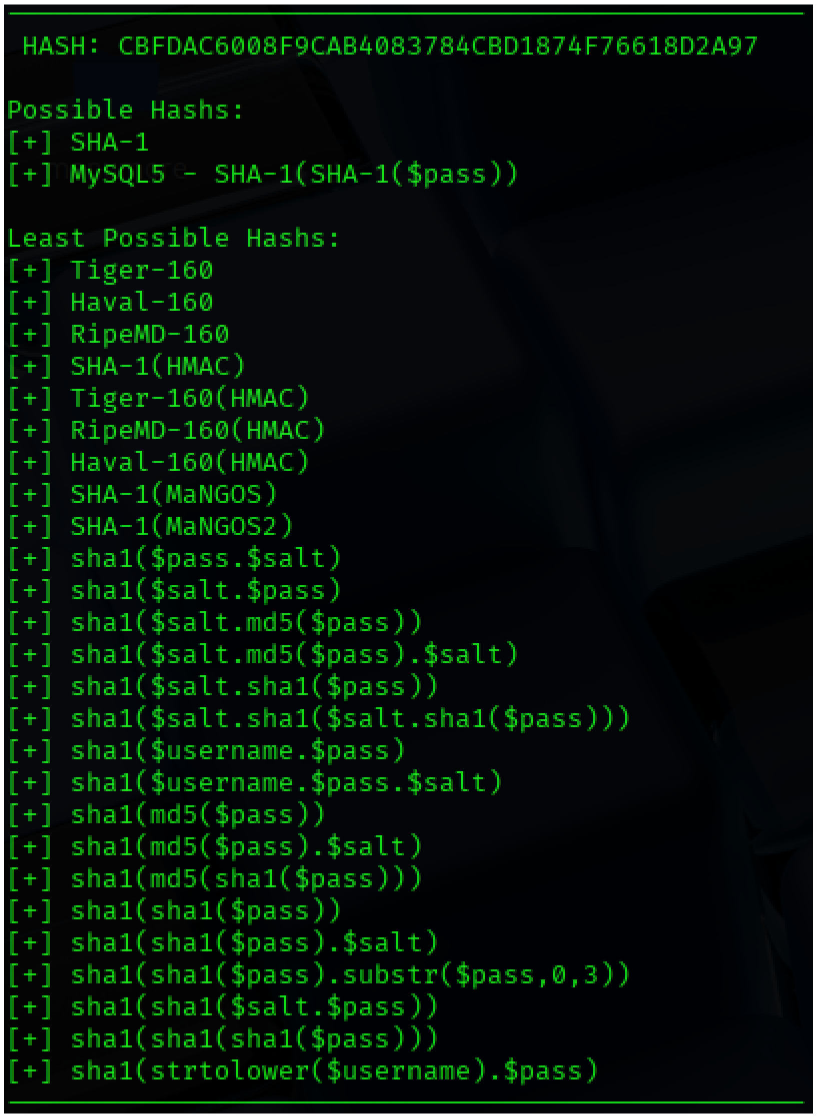


```
?:easy

1 password hash cracked, 0 left
```

Hash 2 - SHA-1

Hash: `CBFDAC6008F9CAB4083784CBD1874F76618D2A97`.

Se usa la herramienta `hash-identifier` para averiguar el *hash*.

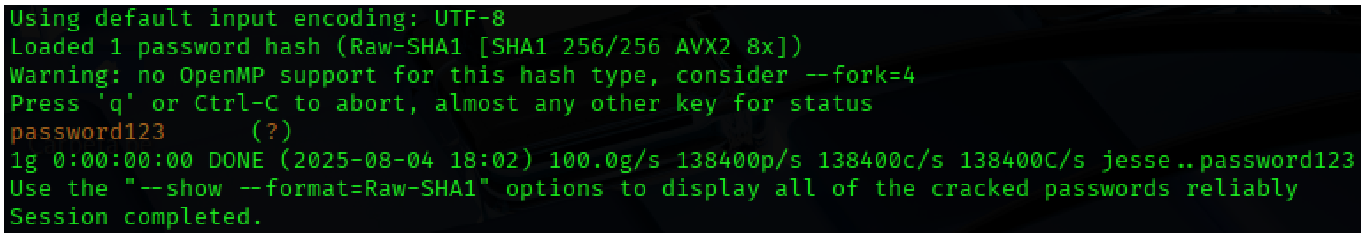


```
 HASH: CBFDAC6008F9CAB4083784CBD1874F76618D2A97

Possible Hashs:
[+] SHA-1
[+] MySQL5 - SHA-1(SHA-1($pass))

Least Possible Hashs:
[+] Tiger-160
[+] Haval-160
[+] RipeMD-160
[+] SHA-1(HMAC)
[+] Tiger-160(HMAC)
[+] RipeMD-160(HMAC)
[+] Haval-160(HMAC)
[+] SHA-1(MaNGOS)
[+] SHA-1(MaNGOS2)
[+] sha1($pass.$salt)
[+] sha1($salt.$pass)
[+] sha1($salt.md5($pass))
[+] sha1($salt.md5($pass).$salt)
[+] sha1($salt.sha1($pass))
[+] sha1($salt.sha1($salt.sha1($pass)))
[+] sha1($username.$pass)
[+] sha1($username.$pass.$salt)
[+] sha1(md5($pass))
[+] sha1(md5($pass).$salt)
[+] sha1(md5(sha1($pass)))
[+] sha1(sha1($pass))
[+] sha1(sha1($pass).$salt)
[+] sha1(sha1($pass).substr($pass,0,3))
[+] sha1(sha1($salt.$pass))
[+] sha1(sha1(sha1($pass)))
[+] sha1(strtolower($username).$pass)
```

Se usa `john` para realizar fuerza bruta.

`vim sha-1_hash.txt`

`john --format=raw-sha1 --wordlist=/usr/share/wordlists/rockyou.txt sha-1_hash.txt`

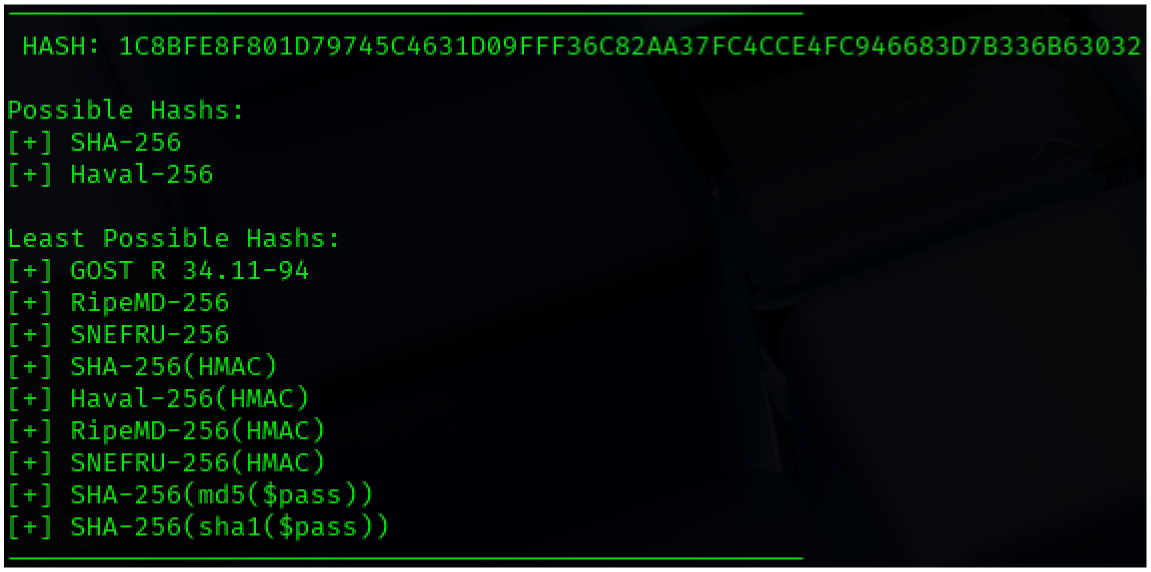


```
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
password123      (?)
1g 0:00:00:00 DONE (2025-08-04 18:02) 100.0g/s 138400p/s 138400c/s 138400C/s jesse..password123
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed.
```

Hash 3 - SHA-256

Hash: `1C8BFE8F801D79745C4631D09FFF36C82AA37FC4CCE4FC946683D7B336B63032`.

Se usa la herramienta `hash-identifier` para averiguar el *hash*.

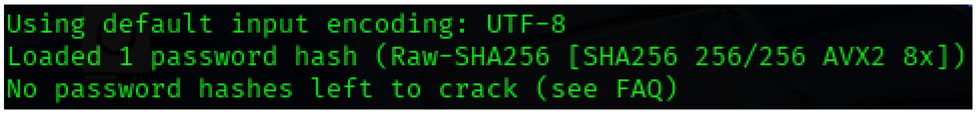


```
--------------------------------------------------
 HASH: 1C8BFE8F801D79745C4631D09FFF36C82AA37FC4CCE4FC946683D7B336B63032

Possible Hashs:
[+] SHA-256
[+] Haval-256

Least Possible Hashs:
[+] GOST R 34.11-94
[+] RipeMD-256
[+] SNEFRU-256
[+] SHA-256(HMAC)
[+] Haval-256(HMAC)
[+] RipeMD-256(HMAC)
[+] SNEFRU-256(HMAC)
[+] SHA-256(md5($pass))
[+] SHA-256(sha1($pass))
--------------------------------------------------
```

Se usa `john` para realizar fuerza bruta.

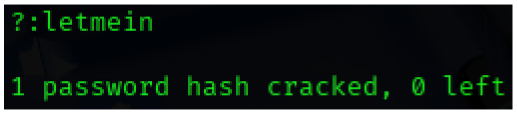
`vim sha256_hash.txt`

`john --format=raw-sha256 --wordlist=/usr/share/wordlists/rockyou.txt sha256_hash.txt`



```
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 256/256 AVX2 8x])
No password hashes left to crack (see FAQ)
```

`john --show --format=raw-sha256 sha256_hash.txt`

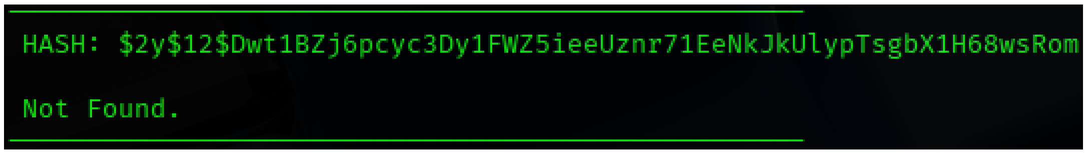


```
?:letmein

1 password hash cracked, 0 left
```

Hash 4 - Bcrypt

Hash: `$2y$12$Dwt1BZj6pcyc3Dy1FWZ5ieeUznr71EeNkJkUlypTsgbX1H68wsRom`.

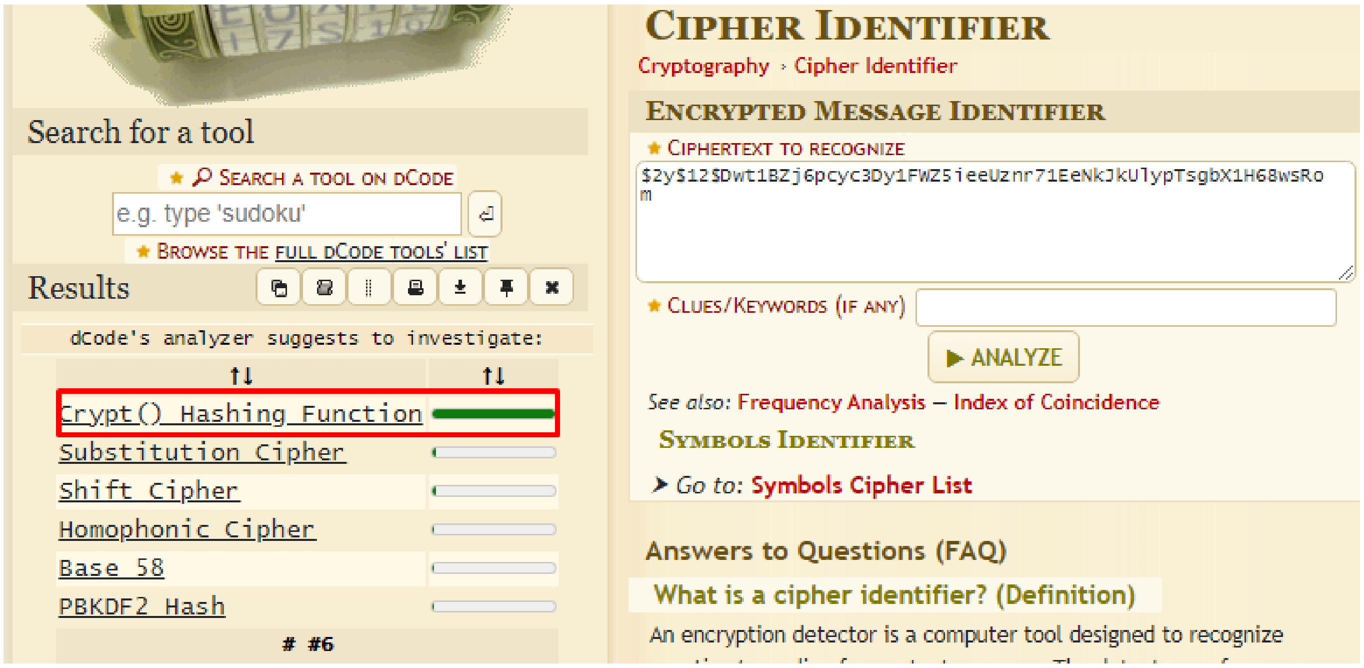
Se usa la herramienta `hash-identifier` para averiguar el *hash*.



```
HASH: $2y$12$Dwt1BZj6pcyc3Dy1FWZ5ieeUznr71EeNkJkUlypTsgbX1H68wsRom

Not Found.
```

No se identifica ningún *hash*

Se identifica el *hash* con Cipher Identifier.



Search for a tool

★ SEARCH A TOOL ON DCODE

e.g. type 'sudoku'

★ BROWSE THE FULL DCODE TOOLS' LIST

Results

dCode's analyzer suggests to investigate:

Crypt() Hashing Function
Substitution Cipher
Shift Cipher
Homophonic Cipher
Base 58
PBKDF2 Hash

CIPHER IDENTIFIER

Cryptography › Cipher Identifier

ENCRYPTED MESSAGE IDENTIFIER

★ CIPHERTEXT TO RECOGNIZE

```
$2y$12$Dwt1BZj6pcyc3Dy1FWZ5ieeUznr71EeNkJkUlypTsgbX1H68wsRom
```

★ CLUES/KEYWORDS (IF ANY)

▶ ANALYZE

See also: Frequency Analysis – Index of Coincidence

SYMBOLS IDENTIFIER

➤ Go to: Symbols Cipher List

Answers to Questions (FAQ)

What is a cipher identifier? (Definition)

An encryption detector is a computer tool designed to recognize

Se usa `john` para realizar fuerza bruta.

`vim bcrypt_hash.txt`

`john --format=bcrypt --wordlist=/usr/share/wordlists/rockyou.txt --fork=4 bcrypt_hash.txt`

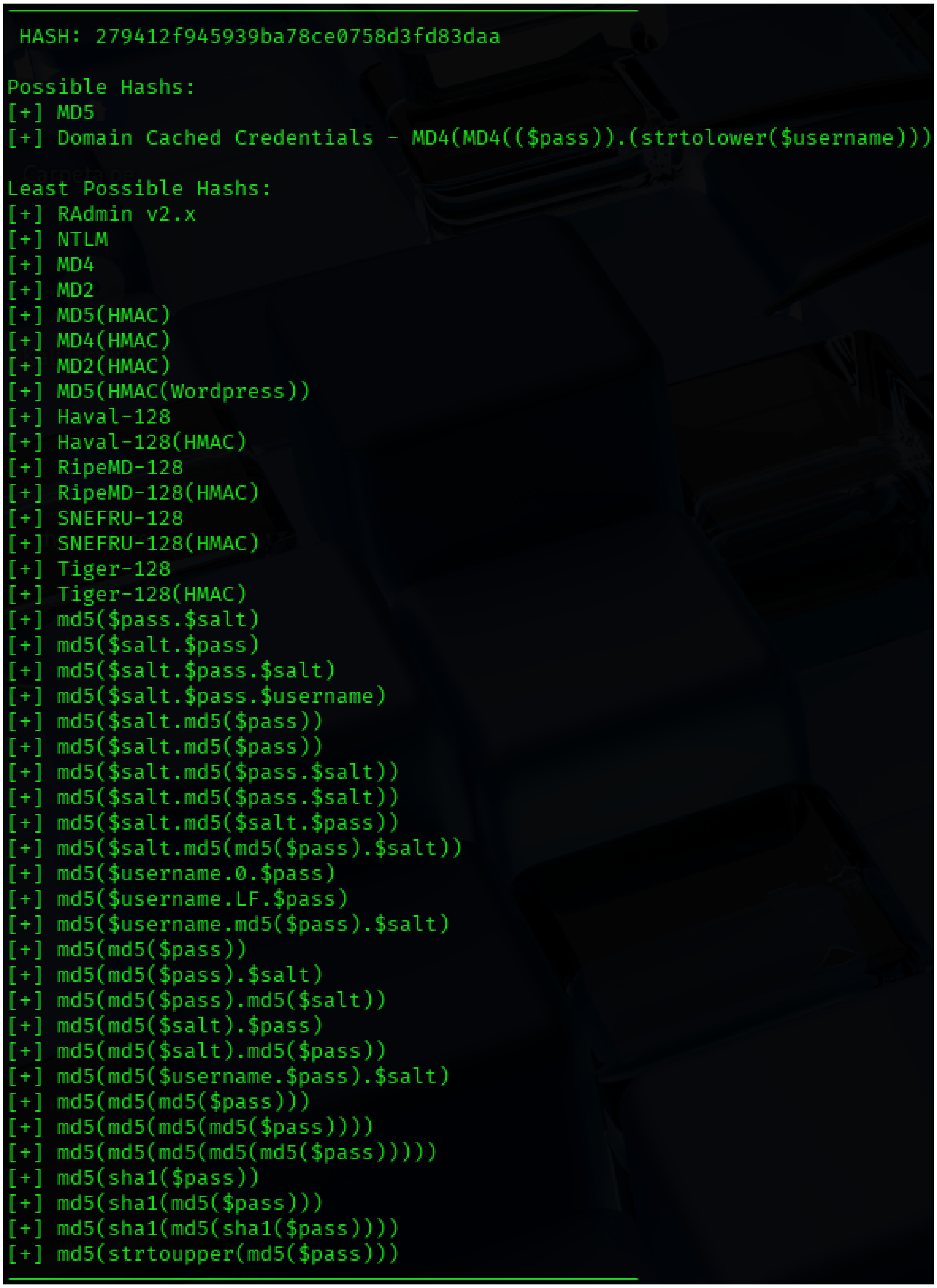
Este *hash* tarda bastante.

`bleh`

Hash 5 - MD4

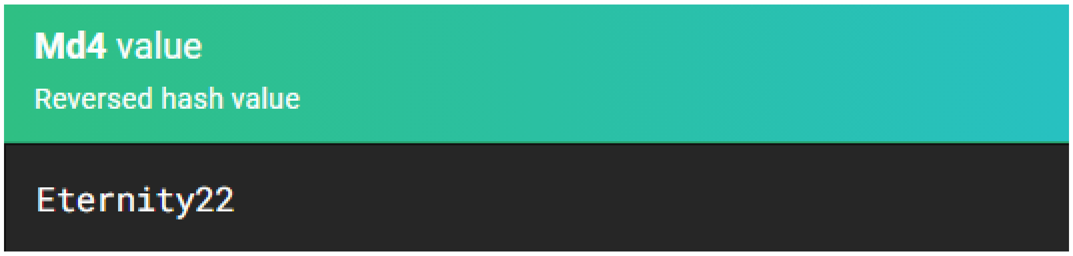
Hash: `279412f945939ba78ce0758d3fd83daa`.

Se usa la herramienta `hash-identifier` para averiguar el *hash*.



```
 HASH: 279412f945939ba78ce0758d3fd83daa

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
[+] MD5(HMAC)
[+] MD4(HMAC)
[+] MD2(HMAC)
[+] MD5(HMAC(Wordpress))
[+] Haval-128
[+] Haval-128(HMAC)
[+] RipeMD-128
[+] RipeMD-128(HMAC)
[+] SNEFRU-128
[+] SNEFRU-128(HMAC)
[+] Tiger-128
[+] Tiger-128(HMAC)
[+] md5($pass.$salt)
[+] md5($salt.$pass)
[+] md5($salt.$pass.$salt)
[+] md5($salt.$pass.$username)
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($salt.$pass))
[+] md5($salt.md5(md5($pass).$salt))
[+] md5($username.0.$pass)
[+] md5($username.LF.$pass)
[+] md5($username.md5($pass).$salt)
[+] md5(md5($pass))
[+] md5(md5($pass).$salt)
[+] md5(md5($pass).md5($salt))
[+] md5(md5($salt).$pass)
[+] md5(md5($salt).md5($pass))
[+] md5(md5($username.$pass).$salt)
[+] md5(md5(md5($pass)))
[+] md5(md5(md5(md5($pass))))
[+] md5(md5(md5(md5(md5($pass)))))
[+] md5(sha1($pass))
[+] md5(sha1(md5($pass)))
[+] md5(sha1(md5(sha1($pass))))
[+] md5(strtoupper(md5($pass)))
```

Se utiliza MD5 Hashing para desencriptarlo.



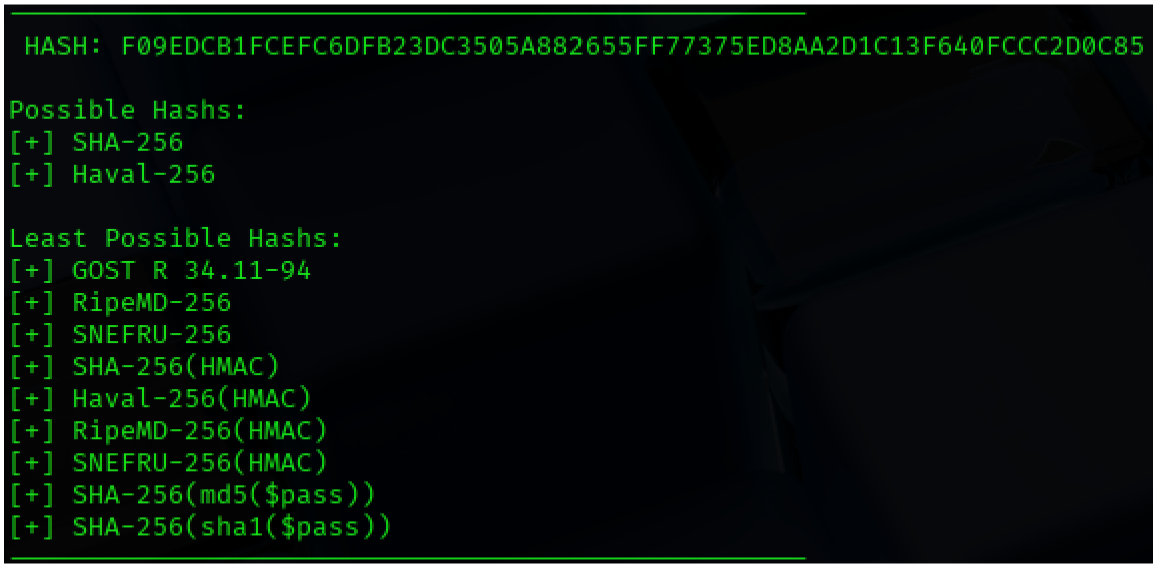
```
Eternity22
```

Level 2

Hash 1 - SHA-256

Hash: `F09EDCB1FCEFC6DFB23DC3505A882655FF77375ED8AA2D1C13F640FCCC2D0C85`.

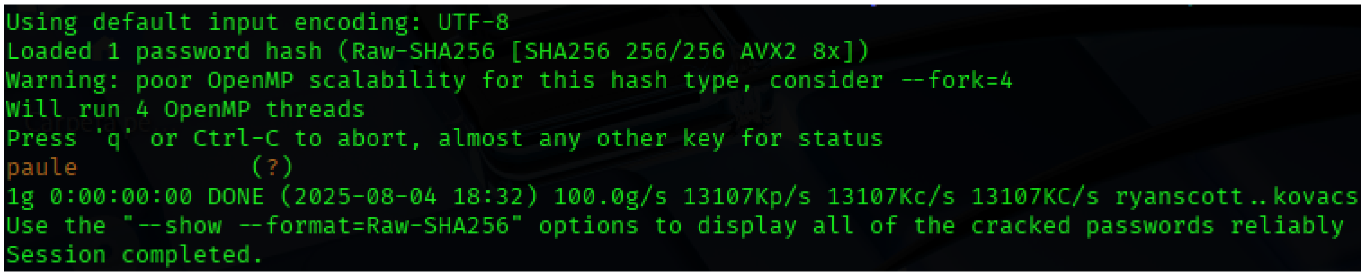
Se usa la herramienta `hash-identifier` para averiguar el *hash*.



```
 HASH: F09EDCB1FCEFC6DFB23DC3505A882655FF77375ED8AA2D1C13F640FCCC2D0C85

Possible Hashs:
[+] SHA-256
[+] Haval-256

Least Possible Hashs:
[+] GOST R 34.11-94
[+] RipeMD-256
[+] SNEFRU-256
[+] SHA-256(HMAC)
[+] Haval-256(HMAC)
[+] RipeMD-256(HMAC)
[+] SNEFRU-256(HMAC)
[+] SHA-256(md5($pass))
[+] SHA-256(sha1($pass))
```

`vim sha256_hash.txt`

`john --format=raw-sha256 --wordlist=/usr/share/wordlists/rockyou.txt sha256_hash.txt`

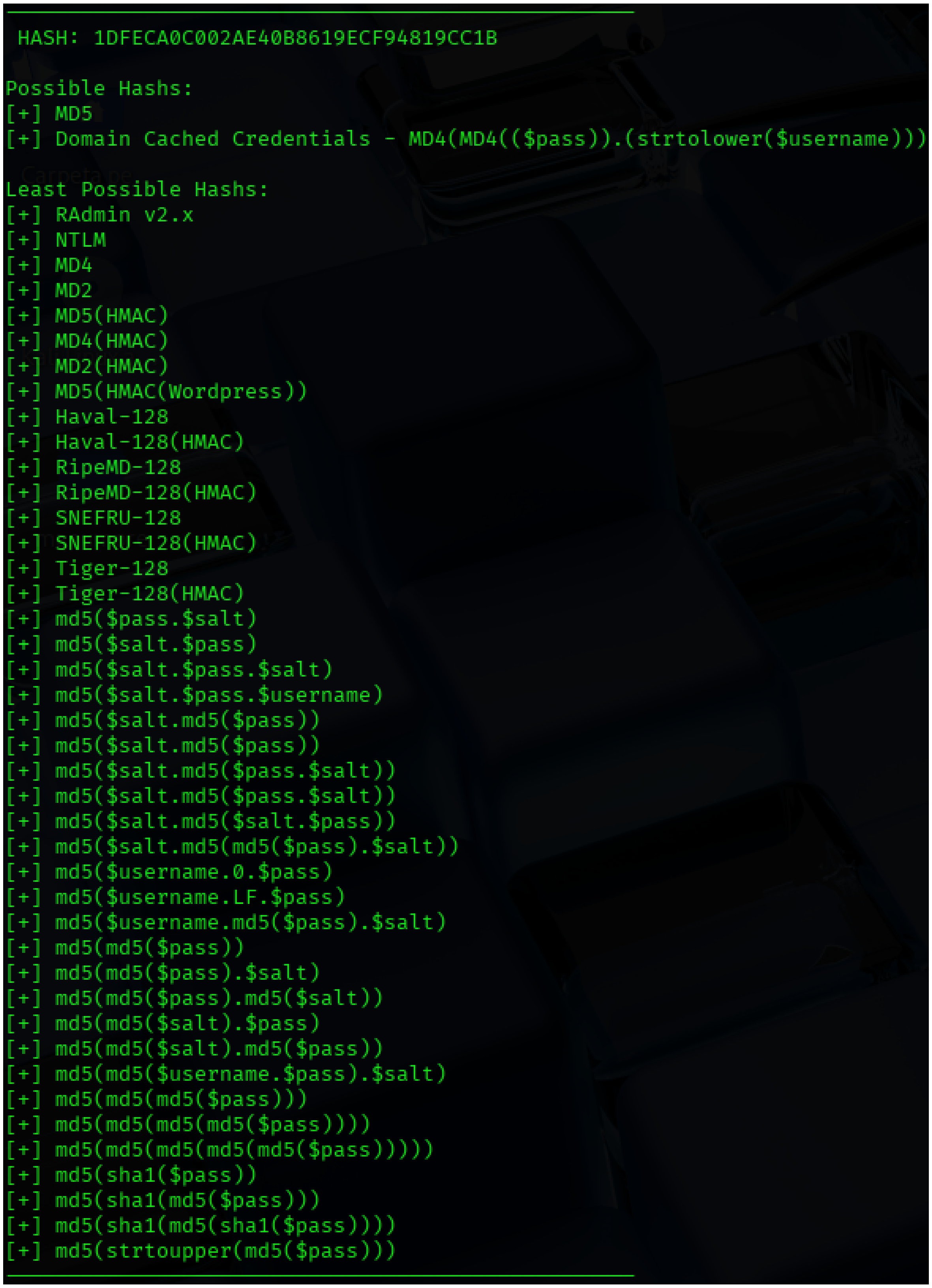


```
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=4
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
paule            (?)
1g 0:00:00:00 DONE (2025-08-04 18:32) 100.0g/s 13107Kp/s 13107Kc/s 13107KC/s ryanscott..kovacs
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.
```

Hash 2 - NTLM

Hash: `1DFECA0C002AE40B8619ECF94819CC1B`.

Se usa la herramienta `hash-identifier` para averiguar el *hash*.

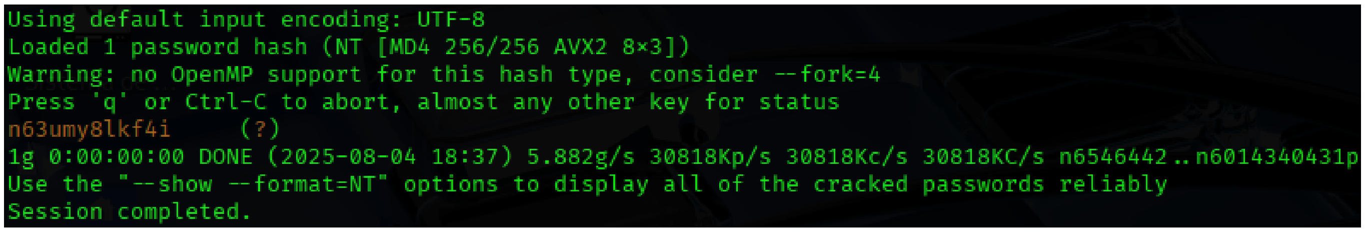


```
 HASH: 1DFECA0C002AE40B8619ECF94819CC1B

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
[+] MD5(HMAC)
[+] MD4(HMAC)
[+] MD2(HMAC)
[+] MD5(HMAC(Wordpress))
[+] Haval-128
[+] Haval-128(HMAC)
[+] RipeMD-128
[+] RipeMD-128(HMAC)
[+] SNEFRU-128
[+] SNEFRU-128(HMAC)
[+] Tiger-128
[+] Tiger-128(HMAC)
[+] md5($pass.$salt)
[+] md5($salt.$pass)
[+] md5($salt.$pass.$salt)
[+] md5($salt.$pass.$username)
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($pass.$salt))
[+] md5($salt.md5($salt.$pass))
[+] md5($salt.md5(md5($pass).$salt))
[+] md5($username.0.$pass)
[+] md5($username.LF.$pass)
[+] md5($username.md5($pass).$salt)
[+] md5(md5($pass))
[+] md5(md5($pass).$salt)
[+] md5(md5($pass).md5($salt))
[+] md5(md5($salt).$pass)
[+] md5(md5($salt).md5($pass))
[+] md5(md5($username.$pass).$salt)
[+] md5(md5(md5($pass)))
[+] md5(md5(md5(md5($pass))))
[+] md5(md5(md5(md5(md5($pass)))))
[+] md5(sha1($pass))
[+] md5(sha1(md5($pass)))
[+] md5(sha1(md5(sha1($pass))))
[+] md5(strtoupper(md5($pass)))
```

Se usa `john` para realizar fuerza bruta.

`vim ntlm_hash.txt`

`john --format=nt --wordlist=/usr/share/wordlists/rockyou.txt ntlm_hash.txt`

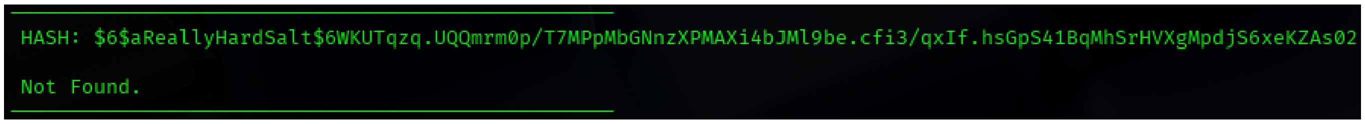


```
Using default input encoding: UTF-8
Loaded 1 password hash (NT [MD4 256/256 AVX2 8×3])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
n63umy8lkf4i     (?)
1g 0:00:00:00 DONE (2025-08-04 18:37) 5.882g/s 30818Kp/s 30818Kc/s 30818KC/s n6546442..n6014340431p
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed.
```

Hash 3 - SHA512

Hash: `$6$aReallyHardSalt$6WKUTqzq.UQQmrm0p/T7MPpMbGNnzXPMAXi4bJMl9be.cfi3/qxIf.hsGpS41BqMhSrHVXgMpdjS6xeKZAs02`.

Se usa la herramienta `hash-identifier` para averiguar el *hash*.



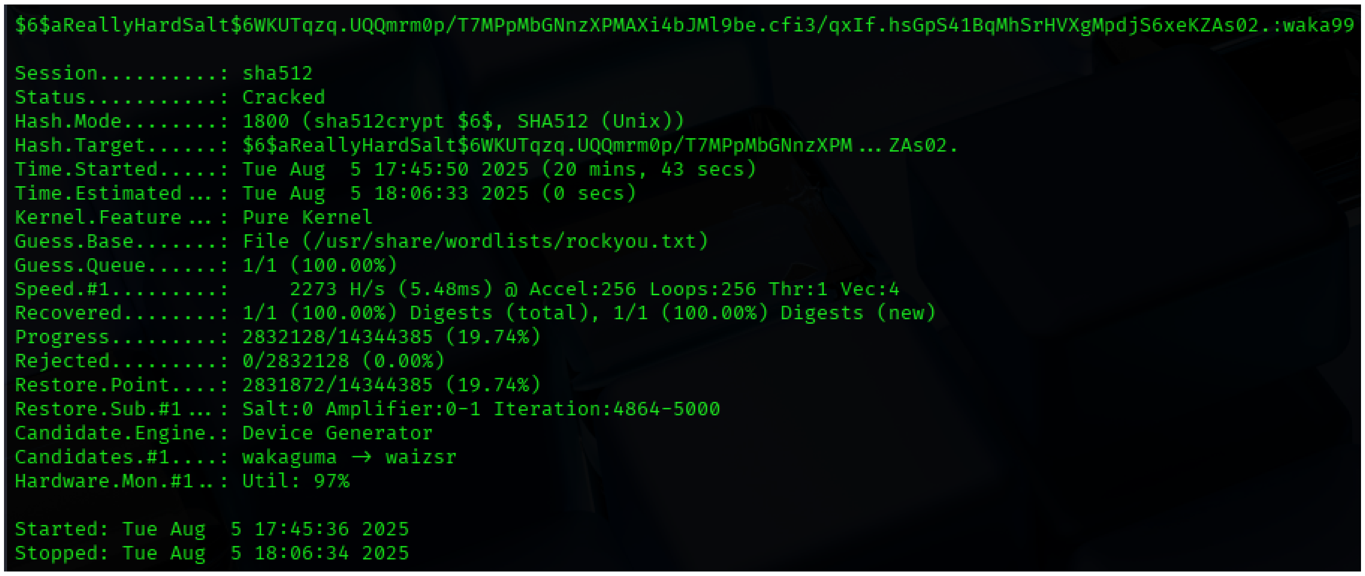
```
 HASH: $6$aReallyHardSalt$6WKUTqzq.UQQmrm0p/T7MPpMbGNnzXPMAXi4bJMl9be.cfi3/qxIf.hsGpS41BqMhSrHVXgMpdjS6xeKZAs02

 Not Found.
```

Se identifica el *hash* es **SHA512**.

Este *hash* tarda bastante.

`vim SHA-512_hash.txt`

`hashcat -m 1800 SHA-512_hash.txt /usr/share/wordlists/rockyou.txt --session sha512`

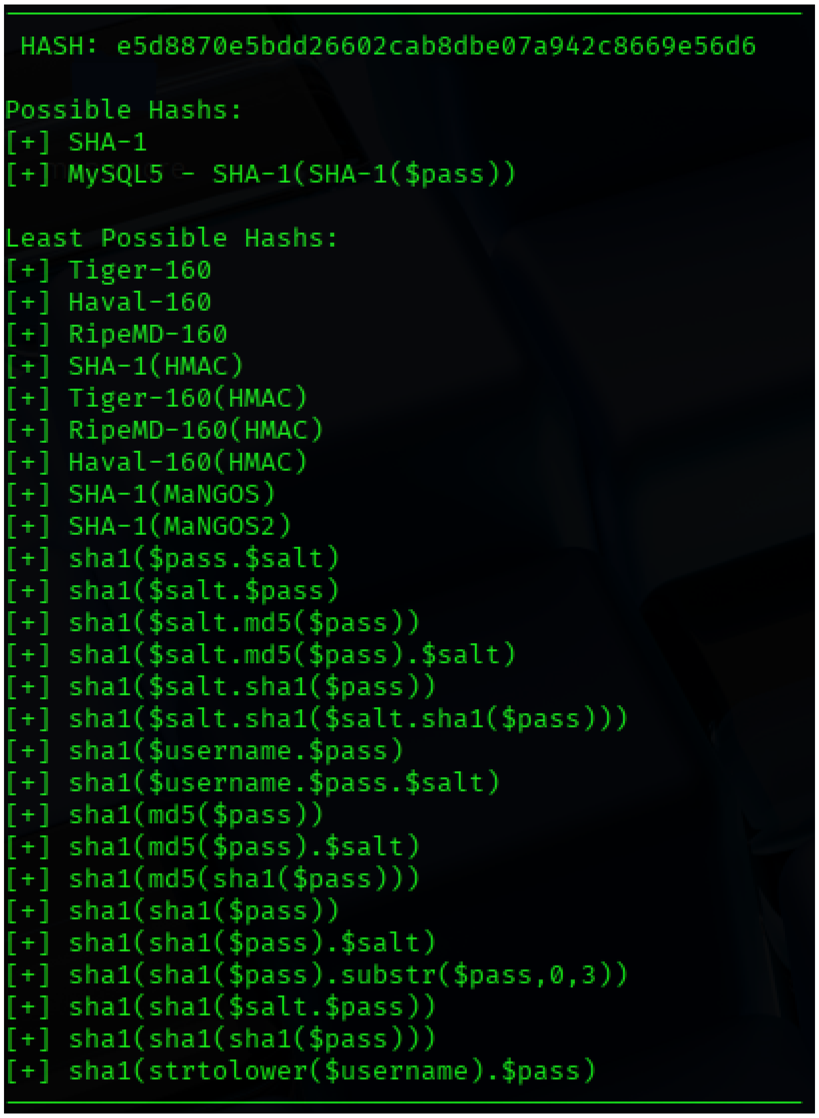


```
$6$aReallyHardSalt$6WKUTqzq.UQQmrm0p/T7MPpMbGNnzXPMAXi4bJMl9be.cfi3/qxIf.hsGpS41BqMhSrHVXgMpdjS6xeKZAs02.:waka99

Session..........: sha512
Status...........: Cracked
Hash.Mode........: 1800 (sha512crypt $6$, SHA512 (Unix))
Hash.Target......: $6$aReallyHardSalt$6WKUTqzq.UQQmrm0p/T7MPpMbGNnzXPM...ZAs02.
Time.Started.....: Tue Aug  5 17:45:50 2025 (20 mins, 43 secs)
Time.Estimated...: Tue Aug  5 18:06:33 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     2273 H/s (5.48ms) @ Accel:256 Loops:256 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 2832128/14344385 (19.74%)
Rejected.........: 0/2832128 (0.00%)
Restore.Point....: 2831872/14344385 (19.74%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4864-5000
Candidate.Engine.: Device Generator
Candidates.#1....: wakaguma -> waizsr
Hardware.Mon.#1..: Util: 97%

Started: Tue Aug  5 17:45:36 2025
Stopped: Tue Aug  5 18:06:34 2025
```

Hash 4 - SHA-1 (HMAC)

Hash: `e5d8870e5bdd26602cab8dbe07a942c8669e56d6`.

Se usa la herramienta `hash-identifier` para averiguar el *hash*.



```
 HASH: e5d8870e5bdd26602cab8dbe07a942c8669e56d6

Possible Hashs:
[+] SHA-1
[+] MySQL5 - SHA-1(SHA-1($pass))

Least Possible Hashs:
[+] Tiger-160
[+] Haval-160
[+] RipeMD-160
[+] SHA-1(HMAC)
[+] Tiger-160(HMAC)
[+] RipeMD-160(HMAC)
[+] Haval-160(HMAC)
[+] SHA-1(MaNGOS)
[+] SHA-1(MaNGOS2)
[+] sha1($pass.$salt)
[+] sha1($salt.$pass)
[+] sha1($salt.md5($pass))
[+] sha1($salt.md5($pass).$salt)
[+] sha1($salt.sha1($pass))
[+] sha1($salt.sha1($salt.sha1($pass)))
[+] sha1($username.$pass)
[+] sha1($username.$pass.$salt)
[+] sha1(md5($pass))
[+] sha1(md5($pass).$salt)
[+] sha1(md5(sha1($pass)))
[+] sha1(sha1($pass))
[+] sha1(sha1($pass).$salt)
[+] sha1(sha1($pass).substr($pass,0,3))
[+] sha1(sha1($salt.$pass))
[+] sha1(sha1(sha1($pass)))
[+] sha1(strtolower($username).$pass)
```

`hashcat -m 1100 SHA-1_hash.txt /usr/share/wordlists/rockyou.txt`

Este *hash* tarda bastante.

`481616481616`