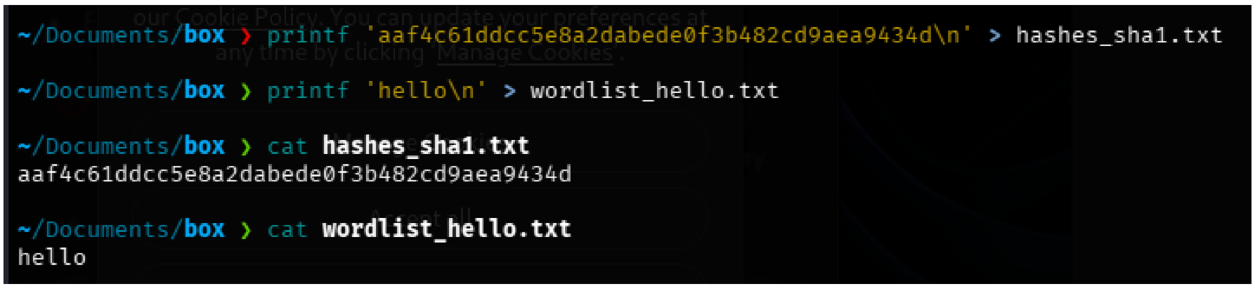
Ejercicios básicos con John the Ripper (Kali Linux)

Ejercicio 1: Crear hash SHA1 y ficheros de prueba

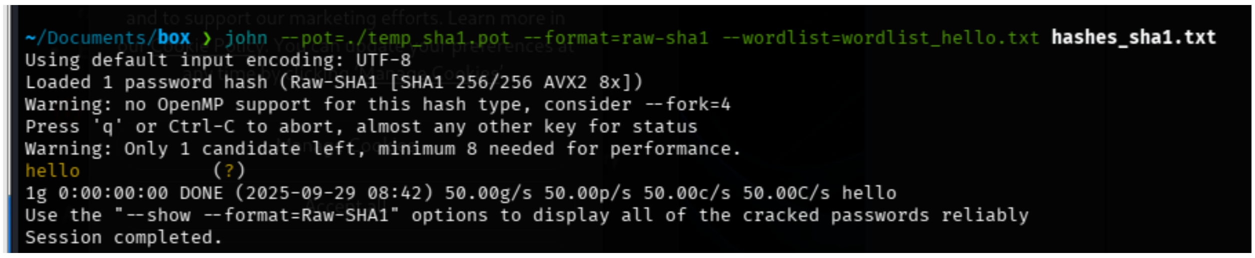
```
# hash SHA1 de "hello" es: aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d

printf 'aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d\n' > hashes_sha1.txt

printf 'hello\n' > wordlist_hello.txt

# comprobar contenido

echo "HASHES:"

cat hashes_sha1.txt

echo

echo "WORDLIST:"

cat wordlist_hello.txt
```



```
~/Documents/box ❯ printf 'aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d\n' > hashes_sha1.txt

~/Documents/box ❯ printf 'hello\n' > wordlist_hello.txt

~/Documents/box ❯ cat hashes_sha1.txt
aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d

~/Documents/box ❯ cat wordlist_hello.txt
hello
```

Intento con John (usa potfile temporal para no tocar ~/.john/john.pot)

```
john --pot=./temp_sha1.pot --format=raw-sha1 --wordlist=wordlist_hello.txt hashes_sha1.txt
```



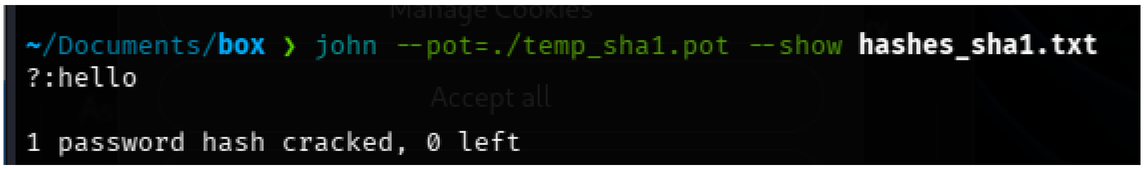
```
~/Documents/box ❯ john --pot=./temp_sha1.pot --format=raw-sha1 --wordlist=wordlist_hello.txt hashes_sha1.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 1 candidate left, minimum 8 needed for performance.
hello            (?)
1g 0:00:00:00 DONE (2025-09-29 08:42) 50.00g/s 50.00p/s 50.00c/s 50.00C/s hello
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed.
```

john --pot=./temp_sha1.pot --show hashes_sha1.txt

Salida esperada en --show (ejemplo):

aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d:hello

1 password hash cracked, 0 left

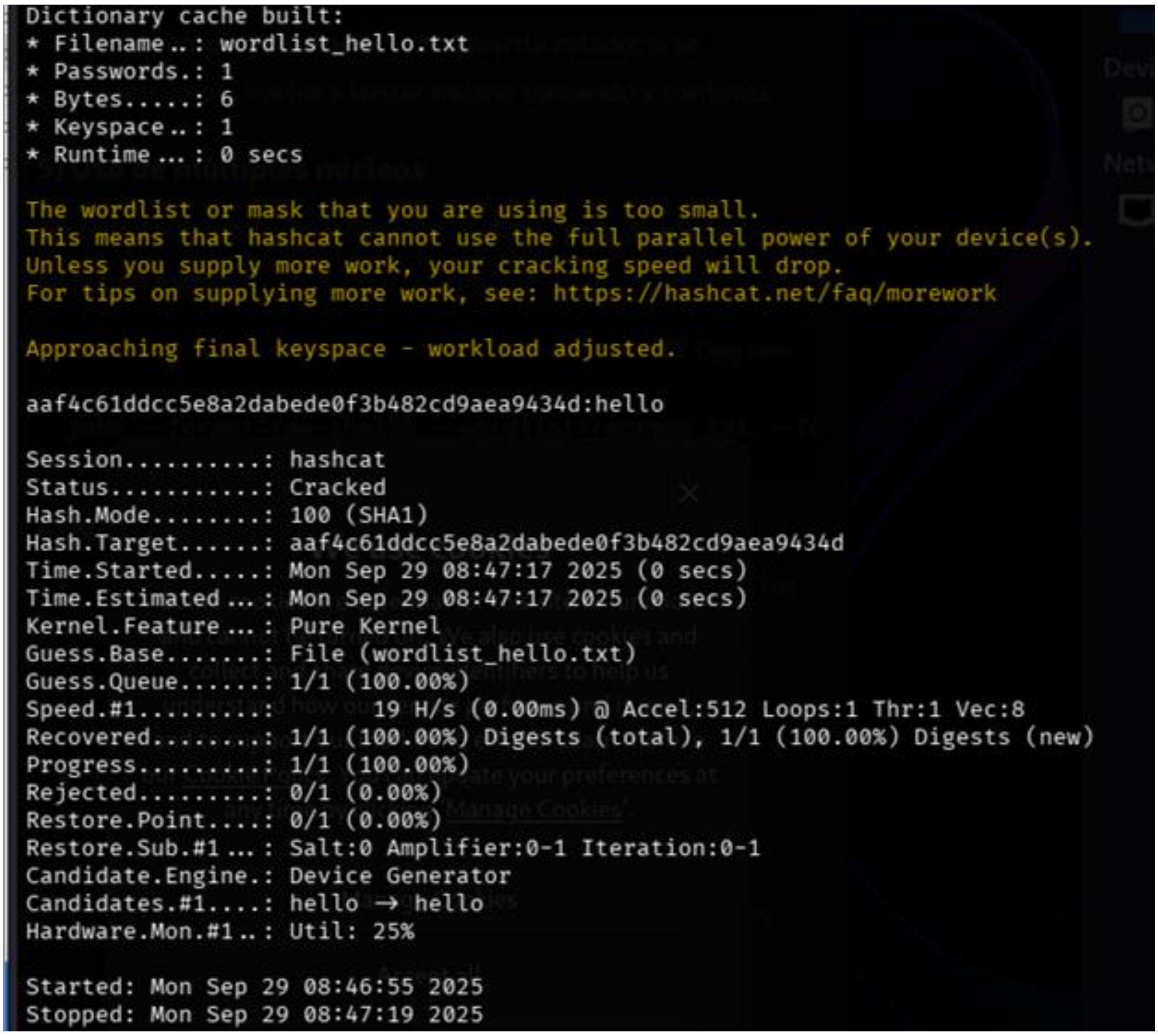


```
~/Documents/box ❯ john --pot=./temp_sha1.pot --show hashes_sha1.txt
?:hello

1 password hash cracked, 0 left
```

Si prefieres hashcat (modo 100 = SHA1)

# ejecutar cracking con hashcat (modo 100 = SHA1)

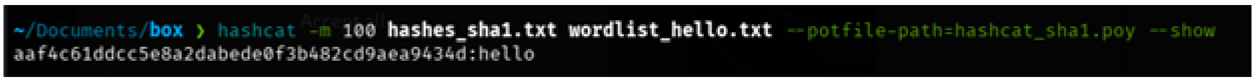
hashcat -m 100 hashes_sha1.txt wordlist_hello.txt --potfile-path=./hashcat_sha1.pot --show



```
Dictionary cache built:
* Filename..: wordlist_hello.txt
* Passwords.: 1
* Bytes.....: 6
* Keyspace..: 1
* Runtime...: 0 secs

The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.

aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d:hello

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 100 (SHA1)
Hash.Target......: aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d
Time.Started.....: Mon Sep 29 08:47:17 2025 (0 secs)
Time.Estimated...: Mon Sep 29 08:47:17 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (wordlist_hello.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:       19 H/s (0.00ms) @ Accel:512 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 1/1 (100.00%)
Rejected.........: 0/1 (0.00%)
Restore.Point....: 0/1 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: hello -> hello
Hardware.Mon.#1..: Util: 25%

Started: Mon Sep 29 08:46:55 2025
Stopped: Mon Sep 29 08:47:19 2025
```

Salida esperada:

aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d:hello

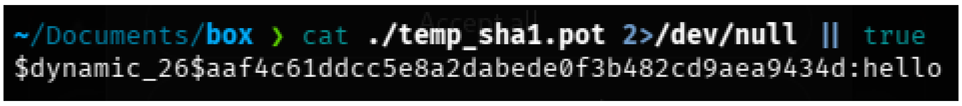


```
~/Documents/box ❯ hashcat -m 100 hashes_sha1.txt wordlist_hello.txt --potfile-path=hashcat_sha1.poy --show
aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d:hello
```

Diagnóstico rápido si John no muestra nada

# ver contenido pot temporal (si usaste --pot=./temp_sha1.pot)

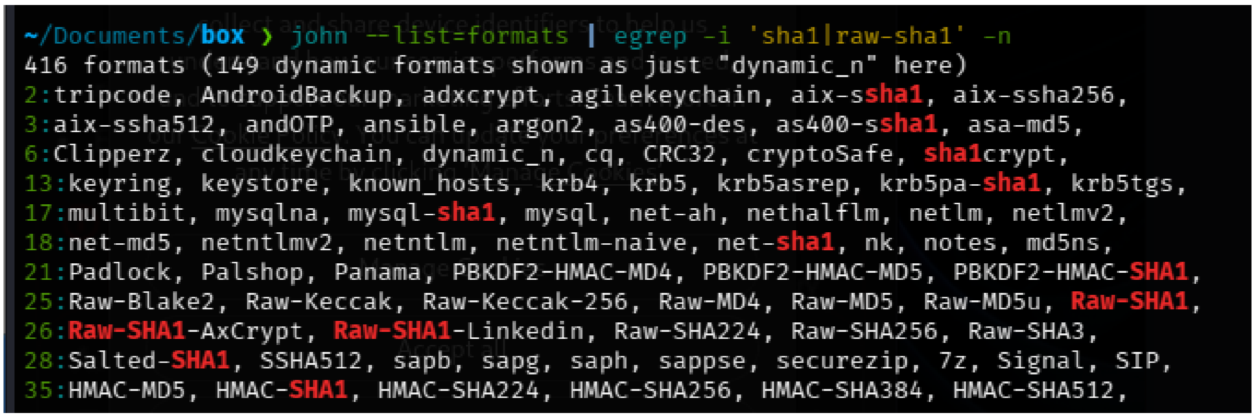
cat ./temp_sha1.pot 2>/dev/null || true



```
~/Documents/box ❯ cat ./temp_sha1.pot 2>/dev/null || true
$dynamic_26$aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d:hello
```

# comprobar formato detectado por john (lista formatos)

john --list=formats | egrep -i 'sha1|raw-sha1' -n

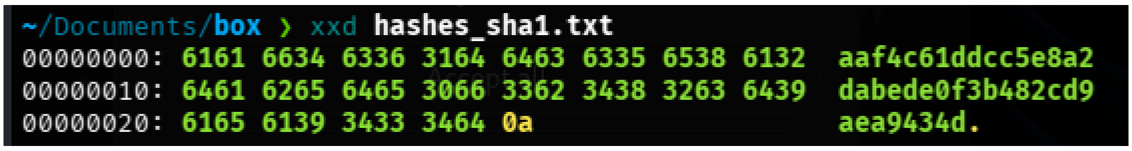


```
~/Documents/box ❯ john --list=formats | egrep -i 'sha1|raw-sha1' -n
416 formats (149 dynamic formats shown as just "dynamic_n" here)
2:tripcode, AndroidBackup, adxcrypt, agilekeychain, aix-ssha1, aix-ssha256,
3:aix-ssha512, andOTP, ansible, argon2, as400-des, as400-ssha1, asa-md5,
6:Clipperz, cloudkeychain, dynamic_n, cq, CRC32, cryptoSafe, sha1crypt,
13:keyring, keystore, known_hosts, krb4, krb5, krb5asrep, krb5pa-sha1, krb5tgs,
17:multibit, mysqlna, mysql-sha1, mysql, net-ah, nethalflm, netlm, netlmv2,
18:net-md5, netntlmv2, netntlm, netntlm-naive, net-sha1, nk, notes, md5ns,
21:Padlock, Palshop, Panama, PBKDF2-HMAC-MD4, PBKDF2-HMAC-MD5, PBKDF2-HMAC-SHA1,
25:Raw-Blake2, Raw-Keccak, Raw-Keccak-256, Raw-MD4, Raw-MD5, Raw-MD5u, Raw-SHA1,
26:Raw-SHA1-AxCrypt, Raw-SHA1-Linkedin, Raw-SHA224, Raw-SHA256, Raw-SHA3,
28:Salted-SHA1, SSHA512, sapb, sapg, saph, sappse, securezip, 7z, Signal, SIP,
35:HMAC-MD5, HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512,
```

Si ves mensajes tipo "No password hashes left to crack" o "0 password hashes cracked, 1 left"

prueba usando un potfile nuevo (como en los ejemplos) para forzar prueba limpia.

asegúrate de que hashes_sha1.txt solo tenga el hash (una línea, sin espacios ni CRLF extras). Usa xxd hashes_sha1.txt para verificar



```
~/Documents/box ❯ xxd hashes_sha1.txt
00000000: 6161 6634 6336 3164 6463 6335 6538 6132  aaf4c61ddcc5e8a2
00000010: 6461 6265 6465 3066 3362 3438 3263 6439  dabede0f3b482cd9
00000020: 6165 6139 3433 3464 0a                   aea9434d.
```

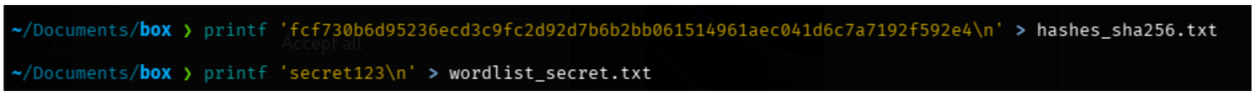
Ejercicio 2: Crear ficheros de prueba

# hash SHA256 de "secret123"

printf 'fcf730b6d95236ecd3c9fc2d92d7b6b2bb061514961aec041d6c7a7192f592e4\n' > hashes_sha256.txt

# wordlist con la contraseña en claro

printf 'secret123\n' > wordlist_secret.txt



```
~/Documents/box ❯ printf 'fcf730b6d95236ecd3c9fc2d92d7b6b2bb061514961aec041d6c7a7192f592e4\n' > hashes_sha256.txt
~/Documents/box ❯ printf 'secret123\n' > wordlist_secret.txt
```

# comprobar contenido

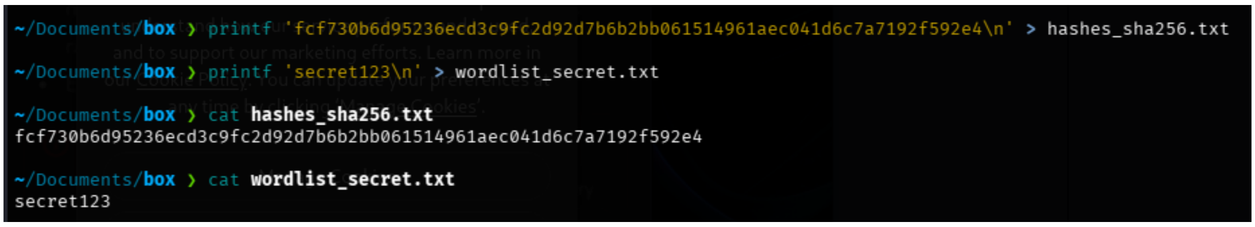
echo "HASH:"

cat hashes_sha256.txt

echo

echo "WORDLIST:"

cat wordlist_secret.txt

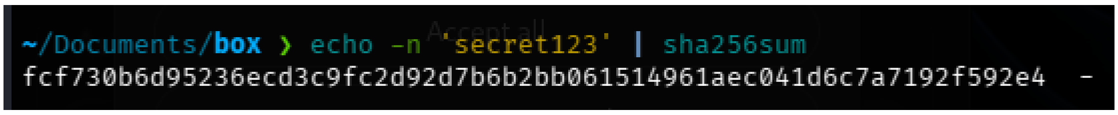


```
~/Documents/box ❯ printf 'fcf730b6d95236ecd3c9fc2d92d7b6b2bb061514961aec041d6c7a7192f592e4\n' > hashes_sha256.txt
~/Documents/box ❯ printf 'secret123\n' > wordlist_secret.txt
~/Documents/box ❯ cat hashes_sha256.txt
fcf730b6d95236ecd3c9fc2d92d7b6b2bb061514961aec041d6c7a7192f592e4
~/Documents/box ❯ cat wordlist_secret.txt
secret123
```

2. Verificar localmente el hash (opcional)

echo -n 'secret123' | sha256sum

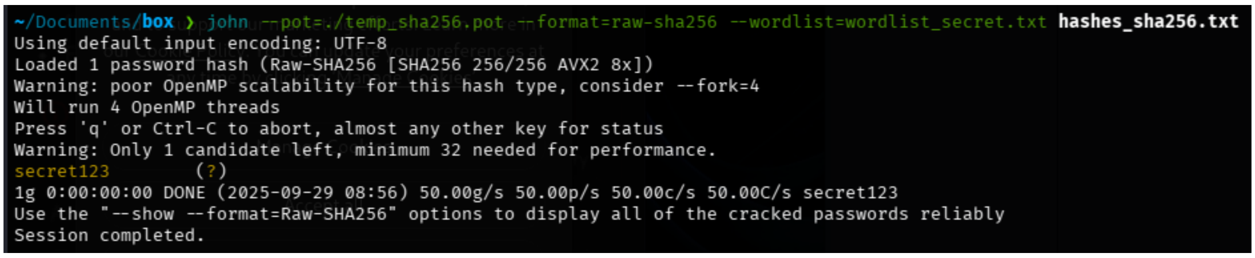
# salida esperada: fcf730b6d95236ecd3c9fc2d92d7b6b2bb061514961aec041d6c7a7192f592e4 -



```
~/Documents/box ❯ echo -n 'secret123' | sha256sum
fcf730b6d95236ecd3c9fc2d92d7b6b2bb061514961aec041d6c7a7192f592e4  -
```

3. Ejecutar John (potfile temporal para no tocar ~/.john/john.pot)

john --pot=./temp_sha256.pot --format=raw-sha256 --wordlist=wordlist_secret.txt hashes_sha256.txt



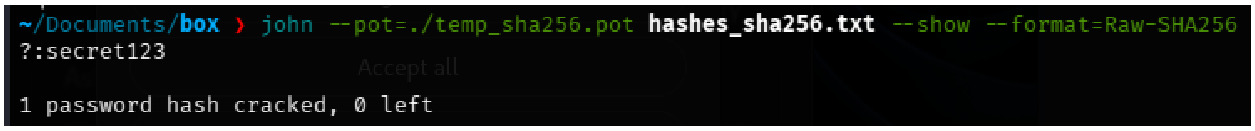
```
~/Documents/box ❯ john --pot=./temp_sha256.pot --format=raw-sha256 --wordlist=wordlist_secret.txt hashes_sha256.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=4
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 1 candidate left, minimum 32 needed for performance.
secret123        (?)
1g 0:00:00:00 DONE (2025-09-29 08:56) 50.00g/s 50.00p/s 50.00c/s 50.00C/s secret123
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.
```

john --pot=./temp_sha256.pot --show hashes_sha256.txt

Salida esperada en --show:

fcf730b6d95236ecd3c9fc2d92d7b6b2bb061514961aec041d6c7a7192f592e4:secret123

1 password hash cracked, 0 left



```
~/Documents/box ❯ john --pot=./temp_sha256.pot hashes_sha256.txt --show --format=Raw-SHA256
?:secret123

1 password hash cracked, 0 left
```

Ejercicio 3: Ejemplo SHA-256

Perfecto — otro ejemplo SHA-256, paso a paso. Usaré la contraseña MiClave!2025. Su SHA-256 es:

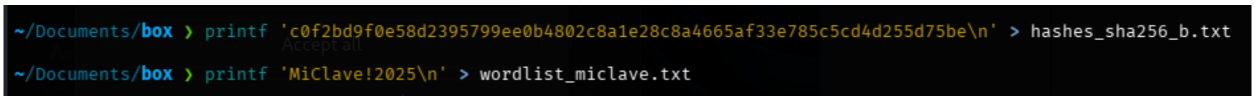
c0f2bd9f0e58d2395799ee0b4802c8a1e28c8a4665af33e785c5cd4d255d75be

Copia/pega estos comandos en tu terminal (Kali/Parrot):

1. Crear ficheros de prueba

printf 'c0f2bd9f0e58d2395799ee0b4802c8a1e28c8a4665af33e785c5cd4d255d75be\n' > hashes_sha256_b.txt

printf 'MiClave!2025\n' > wordlist_miclave.txt



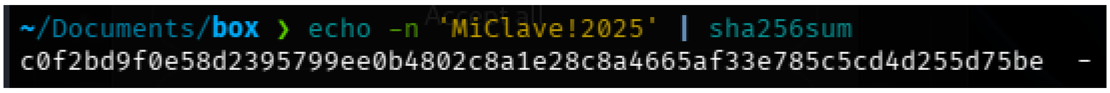
```
~/Documents/box ❯ printf 'c0f2bd9f0e58d2395799ee0b4802c8a1e28c8a4665af33e785c5cd4d255d75be\n' > hashes_sha256_b.txt
~/Documents/box ❯ printf 'MiClave!2025\n' > wordlist_miclave.txt
```

2. Verificar localmente el hash (opcional)

echo -n 'MiClave!2025' | sha256sum

# salida esperada:

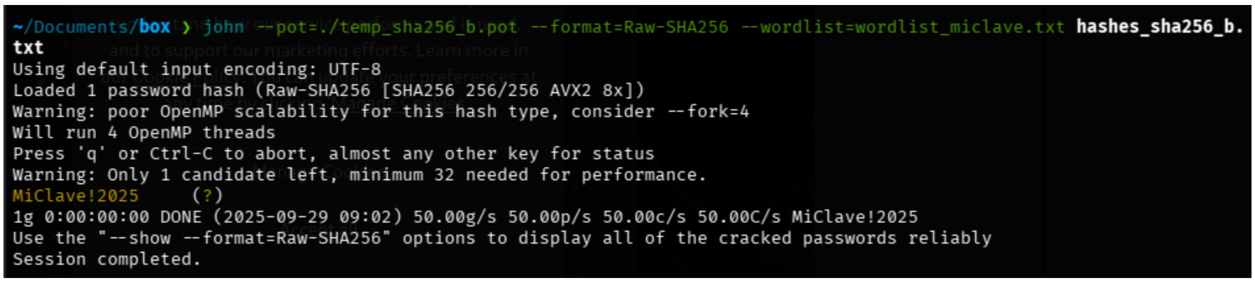
# c0f2bd9f0e58d2395799ee0b4802c8a1e28c8a4665af33e785c5cd4d255d75be -



```
~/Documents/box ❯ echo -n 'MiClave!2025' | sha256sum
c0f2bd9f0e58d2395799ee0b4802c8a1e28c8a4665af33e785c5cd4d255d75be  -
```

3. Probar con John (potfile temporal para no tocar ~/.john/john.pot)

john --pot=./temp_sha256_b.pot --format=raw-sha256 --wordlist=wordlist_miclave.txt hashes_sha256_b.txt



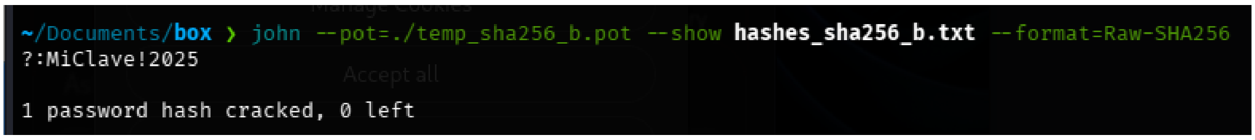
```
~/Documents/box ❯ john --pot=./temp_sha256_b.pot --format=Raw-SHA256 --wordlist=wordlist_miclave.txt hashes_sha256_b.
txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=4
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 1 candidate left, minimum 32 needed for performance.
MiClave!2025     (?)
1g 0:00:00:00 DONE (2025-09-29 09:02) 50.00g/s 50.00p/s 50.00c/s 50.00C/s MiClave!2025
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.
```

john --pot=./temp_sha256_b.pot --show hashes_sha256_b.txt

Salida esperada en --show:

c0f2bd9f0e58d2395799ee0b4802c8a1e28c8a4665af33e785c5cd4d255d75be:MiClave!2025

1 password hash cracked, 0 left



```
~/Documents/box ❯ john --pot=./temp_sha256_b.pot --show hashes_sha256_b.txt --format=Raw-SHA256
?:MiClave!2025

1 password hash cracked, 0 left
```