Hashcat

https://github.com/hashcat/hashcat

Written by Justin Wang || linkedin.com/in/hsiaoan-wang-4514ab45/
Revised by Kent Ickler || 𝕏 @krelkci

Hashcat is a powerful tool for recovering lost passwords, and, thanks to GPU acceleration, it's one of the fastest. It works by rapidly trying different password guesses to determine the original password from its scrambled (hashed) version. Hashcat uses various clever techniques, like dictionary attacks (testing common passwords), leetspeak tricks (e.g., replacing "e" with "3"), pattern-based guessing, and combining different words or phrases. This helps expose weak passwords and poor security habits, which many people rely on when configuring and registering accounts online. Because of its effectiveness, Hashcat is widely used in cybersecurity training, ethical hacking, and penetration testing to improve password security and help organizations strengthen their defenses.

```
hashcat -m <mode> -a <attack> <file storing your hash> <path to wordlist/mask>
```

Commonly Used Modes (-m)

0	MD5
900	MD4
1700	SHA2-512
10	MD5 ($pass.$salt)
20	MD5 ($salt.$pass)
110	SHA1:salt
120	SHA1:pass
2600	md5(md5($pass))
4500	sha1(sha1($pass))
400	phpass
8900	scrypt
2500	WPA/WPA2
2501	WPA/WPA2 PMK
4800	iSCSI CHAP authentication, MD5(CHAP)
5500	NetNTLMv1 / NetNTLMv1+ESS
5600	NetNTLMv2
7500	Kerberos 5, etype 23, AS-REQ Pre-Auth
7300	IPMI 2 RAKP HMAC-SHA1
7350	IPMI2 RAKP HMAC-MD5
13100	Kerberos 5, etype 23, TGS-REP
18200	Kerberos 5, etype 23, AS-REP
19600	Kerberos 5, etype 17, TGS-REP
19700	Kerberos 5, etype 18, TGS-REP
19800	Kerberos 5, etype 17, Pre-Auth
19900	Kerberos 5, etype 18, Pre-Auth
27000	NetNTLMv1 / NetNTLMv1+ESS (NT)
27100	NetNTLMv2 (NT)
27300	SNMPv3 HMAC-SHA512-384
28900	Kerberos 5, etype 18, DB
1000	NTLM
1100	Domain Cached Credentials (DCC), MS Cache
1800	sha512crypt 6, SHA512 (Unix)
3000	LM
5700	Cisco-IOS type 4 (SHA256)
7400	sha256crypt 5, SHA256 (Unix)
8100	Citrix NetScaler (SHA1)
12800	MS-AzureSync PBKDF2-HMAC-SHA256
131	MSSQL (2000)
132	MSSQL (2005)
200	MySQL323
300	MySQL4.1/MySQL5
1731	MSSQL (2012, 2014)
1600	Apache $apr1$ MD5, md5apr1, MD5 (APR)
8300	DNSSEC (NSEC3)
15000	FileZilla Server > 0.9.55
22100	Bitlocker
22400	AES Crypt (SHA256)
29521	LUKS v1 SHA-256 + AES
9500	MS Office 2010
9600	MSOffice 2013
5200	Password Safe v3
6800	LastPass + LastPass sniffed
13400	KeePass 1 (AES/Twofish) and KeePass 2 (AES)
29700	KeePass 1 (AES/Twofish) and KeePass 2 (AES) - keyfile only mode
11600	7Zip
13600	WinZip

Attack Modes (-a)

0 = Straight Dictionary Attack
Example: `hashcat -m 500 -a 0 hash.txt dict.txt`

1 = Combination Attack
Example: `hashcat -m 500 -a 1 hash.txt dict1.txt dict2.txt`

3 = Brute Force Attack
Example: `hashcat -m 500 -a 3 hash.txt ?l?d?u`

6 = Hybrid Wordlist + Mask
Example: `hashcat -m 500 -a 6 hash.txt wordlist.txt ?d?s`

7 = Mask + Wordlist
Example: `hashcat -m 500 -a 7 hash.txt ?d?s wordlist.txt`

Useful Command Arguments

`--runtime=X`	Abort session after X seconds of runtime.
`--session=X`	Define session name to be string X.
`--restore`	Restore session from --session.
`-o`	Define output file for recovered hash.
`--show`	Show the cracked hashes.
`--left`	Show the uncracked hashes.
`--username`	Enable ignoring of usernames in hashfile.
`--remove`	Enable removal of hashes once they are cracked.
`-b`	Run benchmark of selected hash modes.

Mask Character Sets (?)

`?l`	`abcdefghijklmnopqrstuvwxyz`
`?u`	`ABCDEFGHIJKLMNOPQRSTUVWXYZ`
`?d`	`123456789`
`?h`	`0123456789abcdef`
`?H`	`0123456789ABCDEF`
`?s`	`!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~`
`?a`	`?l?u?d?s`
`?b`	`0x00 - 0xff`

Example: `hashcat -m500 -a 3 hash.txt ?l?l?a?a?a?a?d?d`
Brute force cracking using the masks to check for passwords that have 2 lowercase letters, 4 characters of all possibilities, and 2 numbers.

For a more expansive cheatsheet, check this out:

https://www.blackhillsinfosec.com/wp-content/uploads/2020/09/HashcatCheatSheet.v2018.1b.pdf