

Forum:[Hack Forums](#)

Thread: [How to crack an MD5 hash with hashcat](#)

webdev77 - May 13, 2020, 9:45 am

A hash function is an irreversible cryptographic algorithm used to generate a unique string. A hash is the resulting string from a hash function. Passwords are typically hashed and then stored in some form of database for authentication purposes.

When a user enters their credentials, the entered password is run through a hash function, and then it is compared to the stored hash value in a database. If there is a match of hashes, authorization is granted.

Although hashes cannot be reversed, the original value can still be obtained through a variety of methods. Wordlists, rule-based attacks, and brute force methods are all very effective means of recovering a password when a hash value is available.

Today we will focus on bruteforcing a hash. It is the easiest method but a complex password will not be able to be bruteforced in any reasonable amount of time, which is when rule-based attacks or even social engineering become desirable.

First download [hashcat](#). It's open source so you can dig through it if you so desire.

Assuming you already have an MD5 hash to crack (note this process works for any hash type if you read the documentation for the proper parameters to use), place it in a file called hashes.txt in the same directory as hashcat.

I will use the following example MD5 hash.

Code 098f6bcd4621d373cade4e832627b4f6

In a command line tool navigated to the directory of hashcat, enter the following to launch a bruteforce attack on the hash. In this case, -m is the hash type where 0 corresponds to MD5. -a is the attack mode where 3 is brute force. hashes.txt is the file name where hashes are stored.

Code hashcat64 -m 0 -a 3 hashes.txt

Once the attack is complete, you will be notified of a success or failure. To view the hashes cracked, enter the following.

Code hashcat64 hashes.txt --show

That's it. For additional documentation, look at [Hashcat's wiki pages](#). It is an important skill to learn to read for yourself.

Crypt - September 25, 2020, 7:49 pm

Nice tutorial, Webdev.

I am confused a little bit though. You are telling it to go to "hashes.txt" which is where the hash is stored but don't you need to give it a word list to use?

webdev77 - September 26, 2020, 7:57 am

(Sep 25, 2020 07:49 PM)Crypt Wrote: [\[CITE\]](#) Nice tutorial, Webdev.

I am confused a little bit though. You are telling it to go to "hashes.txt" which is where the hash is stored but don't you need to give it a word list to use?

This is a brute force attack. You can also use a word list.

Crypt - September 26, 2020, 11:17 am

(Sep 26, 2020 07:57 AM)webdev77 Wrote: [\[CITE\]](#) This is a brute force attack. You can also use a word list.

I'm derping. For some reason I thought a dictionary attack brute-force attack was the same thing.

ThinWhiteDuke - May 31, 2021, 8:59 am

You can also use char sets to filter your attacks to get a quicker result

xZalurex - June 2, 2021, 6:30 am

the best thing is to get a request at AWS for their large server with the incredibly insanely fast computers. Tere GPU servers are crazy for hashing. Should check it out, mind you don't leave it on for a month cause its 30k