

Forum: [Hack Forums](#)

Thread: [SHA1 Hash decrypting](#)

themet315 - May 30, 2018, 4:21 pm

I'm trying to decrypt a SHA1 salted hash.

The hashing algorithm is:

Code SHA-1('--'.\$salt.'--'.\$pass'--') ((yes, it is from the 8tracks leak))

Example:

password: hello

salt: test

Code SHA-1('--test--hello--')returns: b5150f76dae9b7465e69594dd7cfa3d04702102

For those who have experience using hashcat, what would be the best way to find the pass?
Is there a way to perform a mask attack to find 'hello', while keeping '--test--' constant at the beginning and '--' at the end?

Using hashcat mask attack, characters can stay constant during the iterations like:

Code hashcat64.exe -a 3 pass?dreturns the candidates:

Code pass0

pass1

...

pass9

But when I try to prepend '--test--' it errors because it mistakes the hyphens as commands. I would post in hashcat forums but I can't seem to get my account to get verified, don't know why.

WhiteTiger. - May 30, 2018, 5:25 pm

You need to use -m 110 (pass, salt) or -m 120 (salt, pass)

However, since you don't know the length of the salt bruteforcing this will almost be impossible and I would move on if I were you.

There's a reason why these things are called hashes and salts.

~WT

themet315 - May 30, 2018, 5:36 pm

Actually, I know the salt. Sorry I forgot to mention that.

The only unknown is the password. I actually figured it out, by using the command to crack a 5-char password:

Code hashcat64.exe -a 6 -m 100 hash.txt salt.dict --custom-charset1=?u?!?d

--custom-charset2=?!?d ?1?2?2?2?2-- --force

where, in the first example, salt.dict would contain

Code --test--

I'm running this right now and the estimated time is an hour. I did a bit of math and found to crack a

7-character password this way would take about 54 days...is there any way to optimize this?

Update: I've optimized it a bit more, using a straight, rule-based attack.

First I've set up a rule file, called 'myrules.rule', which contains:

Code `^- ^- ^t ^s ^e ^t ^- ^- $- $- #Prepends '--test--' and appends '--'`

`^- ^- ^t ^s ^e ^t ^- ^- T8 $- $- #Does the same but also capitalizes the 8th character ('--test--hello--' becomes '--test--Hello--')`

Then I used Maskprocessor program to generate a list of password candidates. I iterated:

Code `mp64.exe ?l -o list.txt`

`mp64.exe ?l?l -o list.txt`

`mp64.exe ?l?l?l -o list.txt`

`mp64.exe ?l?l?l?l -o list.txt`

Which essentially creates one huge text file called 'list.txt' which has all letter combinations from 'a' down to 'zzzz'

Then in hashcat, used the following command line:

Code `hashcat64.exe -m 100 hash.txt list.txt -r myrules`

The end result is a straight dictionary attack consisting of all candidates in list.txt with the rules applied to each to artificially insert the salt and hyphens. Turned out to be much faster than the mask attack, a 6-character crack took a couple hours. However the list.txt file containing all the candidates is a 70GB text file, and that's only for 1-6 character combinations. To add 7-character candidates would probably push it over 150 or 200 GB, increasing exponentially. If there was a way to eliminate candidates for example ones with triple or more repeating characters I could probably get that number down more.