

Assignment 4: Crack the hash.

Grade: 7%

Submission:

You must Edit the lab document. Do not create a new word document.

For every cracked password, include (paste) a labeled screenshot directly below the corresponding question. Each screenshot must display your system username. This assignment consists of multiple challenges. Read the document carefully. You may encounter roadblocks, so start early to allow enough time for troubleshooting.

Submit a single PDF.

Hashcat is a popular and effective password cracker widely used by both penetration testers and sysadmins as well as criminals and spies.

Passwords are no longer stored in plaintext (or shouldn't be, anyway). Instead, passwords are encrypted using a one-way function called a hash. Calculating a password like "Password1" into a hash is lightning quick. What if all you've got is the hash? A brute-force attack to reverse the hash function and recover the password could be computationally infeasible. Like, until the heat death of the universe infeasible.

Luckily, or unluckily depending on your point of view, none of us is likely to live that long, but there are many ways to reverse a hash to recover the original password without resorting to a probably fruitless brute-force attack. It turns out humans are so predictable in their password choices that hashcat can often recover a password.

Uses:

Cracking passwords has many legitimate uses besides the obvious criminal and espionage ones. A sysadmin may wish to pre-emptively check the security of user passwords. If hashcat can crack them, so can an attacker.

Penetration testers on engagement will frequently find themselves cracking stolen password hashes to move laterally inside a network, or to escalate privileges to an admin user. Since penetration testers work to find security holes on purpose, under contract, so that their customer can improve their security, this is also a perfectly legitimate use case.

The real takeaway is that both illegal attackers and legit defenders use hashcat. The best way to prevent an attacker from using hashcat against you is to test your own defenses first to make sure any such attack can't succeed.

How does hashcat work?

At its most basic level, hashcat guesses a password, hashes it, and then compares the resulting hash to the one it's trying to crack. If the hashes match, we know the password. If not, keep guessing. There are numerous attacks sort of a full brute-force attempt, including dictionary attacks,

combinator attacks, mask attacks, and rule-based attacks. Hashcat can also harness the power of your GPU to brute force if you have the computing rig for it — and time to spare.

Hashcat examples

Hashcat dictionary attack: Since humans tend to use really bad passwords, a dictionary attack is the first and obvious place to start. The [rockyou.txt word list](#) is a popular option. Containing more than 14 million passwords sorted by frequency of use, it begins with common passwords such as “123456”, “12345”, “123456789”, “password”, “iloveyou”, “princess”, “1234567”, and “rockyou”, all the way to less common passwords such as “xCvBnM”, “ie168”, “abygurl69”, “a6_123”, and “*7¡Vamos!”.

Many other free wordlists exist on the internet, especially targeted at specific languages. Hashcat lets you specify the wordlist of your choice.

Read Hashcat [documentation](#) for more details.

Hashcat 101

Let's say you are hacking a Linux box and all you have is a shadow.log (download this file from Blackboard for sample hashes)

```
ubuntu:$6$MOCeF0du$eCgZ9.I.hS5CDST1aQHozhLbBH6rAUj97vvW/22eaCynqLv/whZKM  
1freAN3n2XQicWjDr0UFreVv0IAvI.fl0:15549:0:99999:7:::
```

No problem with a handy tool like hashcat and a trusty password list like rockyou.txt. You can possibly crack the password.

Required: Linux machine with Hashcat installed.

Tutorial: [How To Install Hashcat](#)

Commands:

```
sudo apt update  
sudo apt upgrade  
sudo apt install hashcat
```

You can for example run a benchmark to make sure everything is working properly:

```
hashcat -b
```

1. Copy the hash (let us crack, say the manager account password).

Copy the hash that you want to find the password into a new file, for example, user.hash
For the manager, I copied the following into my user.hash file.

*\$6\$MOCeF0du\$eCgZ9.I.hS5CDST1aQHozhLbBH6rAUj97vvW/22eaCynqLv/whZKM1fr
eAN3n2XQiCWjDr0UFreVv0IAvI.fl0*

2. Download rockyou.txt file and save it into the same folder as user.hash. Alternatively, you can use newrockyou.txt for faster execution.

3. Then execute.

hashcat -a 0 user.hash rockyou.txt -m 1800

Ensure all the files are in the same folder and the commands are executed from the same folder. I recommend downloading rockyou.txt from the web.

Where:

-a 0	> use dictionary attack
user.hash	> file that contains hash to be cracked
rockyou.txt	> file containing password list
-m 1800	> Unix type 6 password hashes

You can also execute the command as (if above command gives you error)

hashcat -m 1800 -a 0 user.hash rockyou.txt

```
robinchataut@ubuntu:~/Desktop/hash$ hashcat -m 1800 -a 0 user.hash rockyou.txt  
hashcat (v6.2.5) starting
```

3. It takes time. Do some stretching or drink a coffee ☺

Once it is done you can actually see the cracked password besides the hash like in below screenshot

```
Watchdog: Hardware monitoring interface not found on your system.  
Watchdog: Temperature abort trigger disabled.  
  
Host memory required for this attack: 0 MB  
  
Dictionary cache hit:  
* Filename...: rockyou.txt  
* Passwords..: 14344384  
* Bytes.....: 139921497  
* Keyspace...: 14344384  
  
$6$SaltVal2$ybuPu7Nm09LKn0p0ozhFhFw2SS2cqkLsx8c50EAWfKIjTxBEJqxUQzLh900QMgFTGiw6YuFDueNAapfLKt0f1:forgot  
  
Session.....: hashcat  
Status.....: Cracked  
Hash.Mode.....: 1800 (sha512crypt $6$, SHA512 (Unix))  
Hash.Target.....: $6$SaltVal2$ybuPu7Nm09LKn0p0ozhFhFw2SS2cqkLsx8c50EA...LKt0f1  
Time.Started....: Mon Sep  4 03:59:46 2023 (1 sec)  
Time.Estimated...: Mon Sep  4 03:59:47 2023 (0 secs)  
Kernel.Feature...: Pure Kernel  
Guess.Base.....: File (rockyou.txt)  
Guess.Queue.....: 1/1 (100.00%)  
Speed.#1.....: 2289 H/s (5.29ms) @ Accel:256 Loops:256 Thr:1 Vec:2  
Recovered.....: 1/1 (100.00%) Digests  
Progress.....: 3072/14344384 (0.02%)  
Rejected.....: 0/3072 (0.00%)  
Restore.Point....: 2816/14344384 (0.02%)  
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4864-5000  
Candidate.Engine.: Device Generator  
Candidates.#1....: pirate -> dangerous
```

Congrats on cracking your first password Mr. Hacker!


Challenge 1 (20 Points):

1. Try to find passwords for other users in the given files. You must find at least one other passwords. Attach screenshot of cracked password. Screenshot must show your username.
2. What other credentials (username:password) could have been used to gain access also have SUDO privileges? Refer to shadow.log and sudoers.log.

Challenge 2 (50 Points):

Find the hash mode and password for following hashes. Note that the hash type for the hashes might be different.

Hashcat supports hundreds of hashes, and the chosen hash mode needs to be stated for hashcat to know what to attack. The modes can be found using **hashcat --help** (note: hashcat cannot attack multiple hash types in a single session but there are other tools that can). Recent versions of hashcat also support hash auto-detection if **-m <\$mode>** isn't passed, however as several hash types are constructed similarly, false positives can occur and therefore it isn't encouraged. For example, MD5 would be **-m 0**, SHA1 would be **-m 100** and so on.



```
- [ Hash modes ] -
```

#	Name	Category
900	MD4	Raw Hash
0	MD5	Raw Hash
100	SHA1	Raw Hash
1300	SHA2-224	Raw Hash
1400	SHA2-256	Raw Hash
10800	SHA2-384	Raw Hash
1500	SHA3-224	Raw Hash
1600	SHA3-256	Raw Hash
1700	SHA3-384	Raw Hash
1800	SHA3-512	Raw Hash

Find a way/tool to identify hash type before you proceed. Each cracked password is 5 points. Attach a screenshot of each password you cracked. Screenshot must show your username. See the sample below.

- a. [eb61eead90e3b899c6bcbe27ac581660](#)
- b. [48bb6e862e54f2a795ffc4e541caed4d](#)
- c. [CBFDAC6008F9CAB4083784CBD1874F76618D2A97](#)
- d. [1C8BFE8F801D79745C4631D09FFF36C82AA37FC4CCE4FC946683D7B336B63032](#)
- e. [\\$2y\\$12\\$Dwt1BZj6pcyc3Dy1FWZ5ieeUznr71EeNkJkUlypTsgbX1H68wsRom](#)
- f. [279412f945939ba78ce0758d3fd83daa](#)

- g. F09EDCB1FCEFC6DFB23DC3505A882655FF77375ED8AA2D1C13F640FCCC2D0C85
- h. 1DFECA0C002AE40B8619ECF94819CC1B
- i. \$6\$aReallyHardSalt\$6WKUTqzq.UQQmrm0p/T7MPpMbGNnzXPMAXi4bJmI9be.cfi3/qxIf.hsGpS41BqMhSrHVXgMpdjS6xeKZAs02.
- j. e5d8870e5bdd26602cab8dbe07a942c8669e56d6

Challenge 3 (20 Points):

Now crack the first two hashes in Challenge 2 using John the Ripper. John the Ripper is a password-cracking tool used to identify weak passwords by trying different combinations. It's used for security testing, including dictionary and brute-force attacks, and can crack various password hash formats. Its primary purpose is to assess and strengthen password security.

Attach a screenshot of at first **two** passwords (from challenge 2) you cracked using John the Ripper.

Installation:

```
sudo apt install snapd  
sudo snap install john-the-ripper
```

See Sample run for (b):

Note: You can always use newrockyou.txt for faster execution.

hash: CBFDAC6008F9CAB4083784CBD1874F76618D2A97

hash type: SHA1

hashcat -m 100 user.hash rockyou.txt

john -format=raw-sha1 --wordlist=rockyou.txt user.hash

```
robinchataut@ubunturobin:~/Desktop/hash$ hashcat -m 100 user.hash rockyou.txt  
hashcat (v6.2.5) starting
```

```

Dictionary cache hit:
* Filename..: rockyou.txt
* Passwords.: 14344384
* Bytes.....: 139921497
* Keyspace..: 14344384

cbfdac6008f9cab4083784cbd1874f76618d2a97:password123

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 100 (SHA1)
Hash.Target.....: cbfdac6008f9cab4083784cbd1874f76618d2a97
Time.Started.....: Fri Sep 15 01:48:07 2023 (0 secs)
Time.Estimated....: Fri Sep 15 01:48:07 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 3292.8 kH/s (0.09ms) @ Accel:256 Loops:1 Thr:1 Vec:4
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 2048/14344384 (0.01%)
Rejected.....: 0/2048 (0.00%)
Restore.Point....: 1024/14344384 (0.01%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: kucing -> lovers1

Started: Fri Sep 15 01:47:59 2023
Stopped: Fri Sep 15 01:48:09 2023
robinchataut@ubuntu:~/Desktop/hash$

```

```

robinchataut@ubuntu:~/Desktop/hash$ john -format=raw-sha1 --wordlist=rockyou.txt user.hash
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 128/128 ASIMD 4x])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, 'h' for help, almost any other key for status
password123 (?)
1g 0:00:00:00 DONE (2023-09-15 01:54) 100.0g/s 138400p/s 138400c/s 138400C/s liberty..password123
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed.
robinchataut@ubuntu:~/Desktop/hash$ locate john.pot
/home/robinchataut/snap/john-the-ripper/584/.john/john.pot
robinchataut@ubuntu:~/Desktop/hash$ cat /home/robinchataut/snap/john-the-ripper/584/.john/john.pot
$dynamic_26$cbfdac6008f9cab4083784cbd1874f76618d2a97:password123

```

Questions (10 Points):

1. After using both tools, what key differences did you observe in their functionality, speed, and ease of use? Which tool do you prefer for password cracking tasks, and why? Provide a brief explanation to justify your choice.
2. Password cracking tools are often associated with malicious activities, but they also serve crucial legitimate purposes in cybersecurity. Identify two legitimate use cases where password cracking is ethically and legally used. Briefly explain how and why password cracking is valuable in these scenarios.

References:

- <https://www.csoononline.com/profile/j-m-porup/>, Created by JM Porup
- <https://cybersecurityfreeresource.wordpress.com/2022/03/31/hashcat-101-cracking-password-hashes/>
- https://hashcat.net/wiki/doku.php?id=combinator_attack