# Hashcat Password Testing

## Contents

Time required: 60 minutes

**How to Create Screenshots:** Please use the Windows Snip and Sketch Tool or the Snipping Tool. Paste a screenshot of just the program you are working on. If you are snipping a virtual machine, make sure your focus is outside the virtual machine before you snip.

1. Press and hold down the **Windows key** & **Shift**, then type **S.** This brings up the on-screen snipping tool.
2. Click and Drag your mouse around whatever you want to snip.
3. Release the mouse button. This places the snip into the Windows Clipboard.
4. Go into Word or wherever you want to paste the snip. Hold down **CTRL**, then type **V** to paste the snip.

## Lab Description

**Hashcat** is a well-known password cracker. It is designed to break even the most complex passwords. It tries to crack a specific password in multiple ways, combined with versatility and speed.

Password representations are primarily associated with hash keys, such as MD5, SHA, WHIRLPOOL, RipeMD, etc. They are also defined as a one-way function — this is a mathematical operation that is easy to perform, but very difficult to reverse engineer.

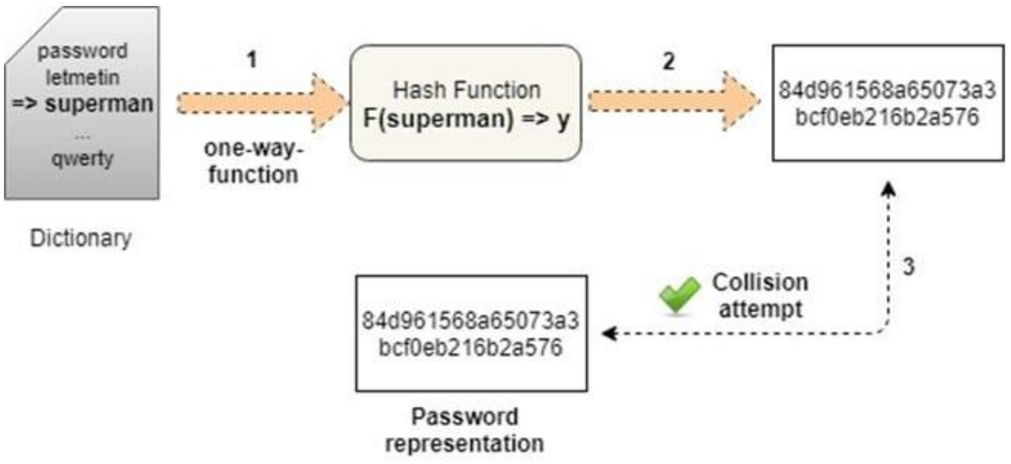
**Hashcat** turns readable data into a garbled state (this is a random string of fixed length size). Hashes do not allow someone to decrypt data with a specific key, as standard encryption protocols allow.

**Hashcat** uses precomputed dictionaries, rainbow tables, and even a brute-force approach to find an effective and efficient way crack passwords. This is an introductory tutorial for cracking passwords using the Hashcat software package.

## How to Crack Hashes

The simplest way to crack a hash is to try first to guess the password. Each attempt is hashed and then is compared to the actual hashed value to see if they are the same.

Dictionary and brute-force attacks are the most common ways of guessing passwords. These techniques make use of a file that contains words, phrases, common passwords, and other strings that are likely to be used as a viable password.



It should be noted that there is no 100% way to prevent dictionary attacks or brute force attacks.

Other approaches that are used to crack passwords are as follows:

- **Lookup Tables:** Hashes are pre-computed from a dictionary and then stored with their corresponding password into a lookup table structure.

- **Reverse Lookup Tables:** This attack allows for a cyber attacker to apply a dictionary or brute-force attack to many hashes at the same time, without having to pre-compute a lookup table.
- **Rainbow Tables:** Rainbow tables are a time-memory technique. They are similar to lookup tables, except that they sacrifice hash cracking speed to make the lookup tables smaller.
- **Hashing with Salt:** With this technique, the hashes are randomized by appending or prepending a random string, called a "salt." This is applied to the password before hashing.

## Cracking Passwords with Hashcat

Hashcat is part of on Kali Linux. It possesses the following features:

- It is multi-threaded;
- It is multi-hash and multi-OS based (Linux, Windows and OSX native binaries);
- It is multi-Algorithm based (MD4, MD5, SHA1, DCC, NTLM, MySQL, etc.);
- All attack-modes can be extended by specialized rules;
- It is possible to resume or limit sessions automatically. It recognizes recovered hashes from the outfile at startup;
- It can load the salt list from the external file. This can be used as a brute-force attack variant;
- The number of threads can be configured and executed based on the lowest priority;
- It supports both hex-charset and hex-salt files;
- The 90+ Algorithm can be implemented with performance and optimization in mind.

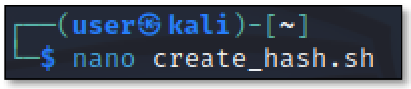
A dictionary attack will be simulated for a set of MD5 hashes initially created and stored in a target file. The "rockyou" wordlist found in Kali Linux will be used.

## Crack a Password by a Dictionary Attack

We will create a file with multiple hash entries from several passwords. They will be outputted to a file called **target_hashes.txt**

A shell script in Linux is like a batch file in Windows. It is a way of automating several commands. Linux pays no attention to file extensions. .sh is a convention amongst Linux users to indicate that the file is a shell script.

1. In Kali Linux: create a shell script with nano.
2. The following command uses the command line editor **nano** to create a file named **create_hash.sh**

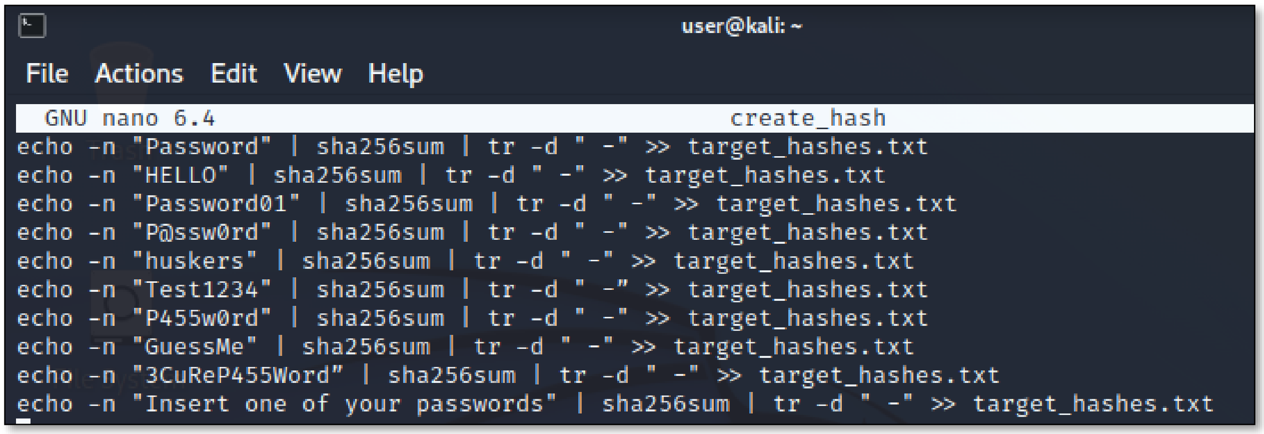


```
┌──(user㉿kali)-[~]
└─$ nano create_hash.sh
```

3. Copy the commands below.

```
echo -n "Password" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "HELLO" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "Password01" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "P@ssw0rd" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "huskers" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "Test1234" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "P455w0rd" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "GuessMe" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "3CuReP455Word" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "Insert one of your passwords" | sha256sum | tr -d " -" >> target_hashes.txt
```

4. In **nano** → use **Shift Insert** to paste the commands into the shell script.
5. Press **CTRL-S** to save the file. **CTRL-X** to exit nano.

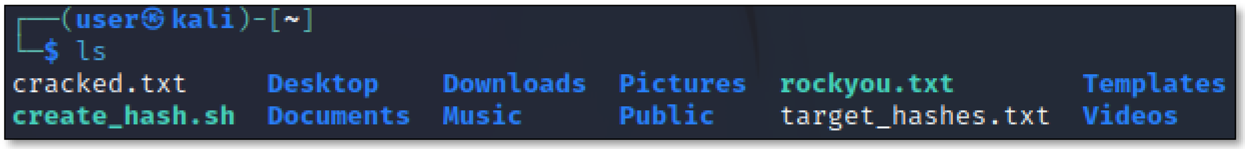


```
  GNU nano 6.4                                    create_hash
echo -n "Password" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "HELLO" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "Password01" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "P@ssw0rd" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "huskers" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "Test1234" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "P455w0rd" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "GuessMe" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "3CuReP455Word" | sha256sum | tr -d " -" >> target_hashes.txt
echo -n "Insert one of your passwords" | sha256sum | tr -d " -" >> target_hashes.txt
```

6. Type in the following command to make create_hashes.sh executable.

```
sudo chmod 744 create_hash.sh
```

7. Type **ls** to make sure the file is in your home folder. Create_hash.sh should be green as shown.



```
┌──(user㉿kali)-[~]
└─$ ls
cracked.txt     Desktop     Downloads  Pictures  rockyou.txt        Templates
create_hash.sh  Documents   Music      Public    target_hashes.txt  Videos
```

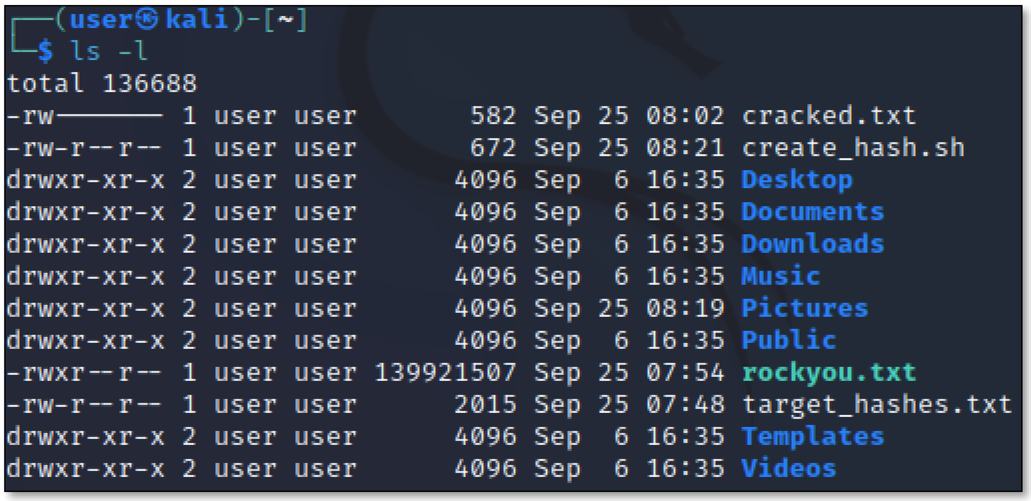
## Command Explanation

- **echo** prints the command to the command prompt.
- **-n** removes the new line added to the end of "Password." This is important as we don't want the new line characters to be hashed with our password.
- **|** The pipe symbol sends the output of the last command into the next command.
- **sha256sum** is a built-in Linux hashing command.
- **tr –d " -"** removes any characters that are a space or hyphen from the output.
- **>>** appends the output of each command to **target_hashes.txt**

## Make a Linux Shell Script Executable

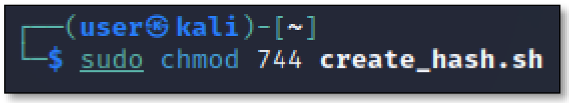
**create_hash.sh** is a text file.

1. Type **ls -l**



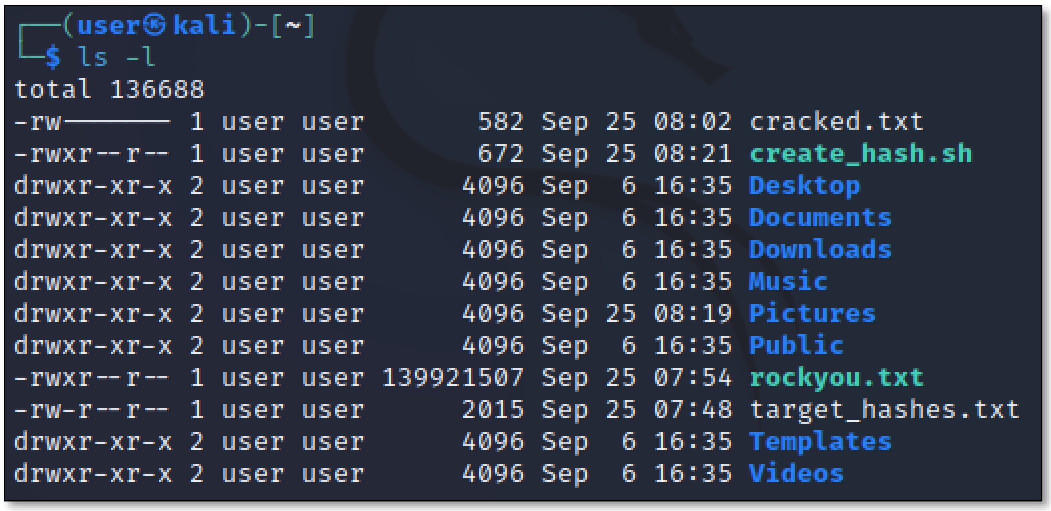
```
┌──(user㉿kali)-[~]
└─$ ls -l
total 136688
-rw------- 1 user user       582 Sep 25 08:02 cracked.txt
-rw-r--r-- 1 user user       672 Sep 25 08:21 create_hash.sh
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Desktop
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Documents
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Downloads
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Music
drwxr-xr-x 2 user user      4096 Sep 25 08:19 Pictures
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Public
-rwxr--r-- 1 user user 139921507 Sep 25 07:54 rockyou.txt
-rw-r--r-- 1 user user      2015 Sep 25 07:48 target_hashes.txt
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Templates
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Videos
```

2. **create_hash.sh** does not have the executable bit set. Enter the following command.



```
┌──(user㉿kali)-[~]
└─$ sudo chmod 744 create_hash.sh
```

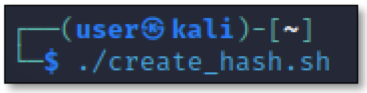
3. Type **ls -l**



```
┌──(user㉿kali)-[~]
└─$ ls -l
total 136688
-rw------- 1 user user       582 Sep 25 08:02 cracked.txt
-rwxr--r-- 1 user user       672 Sep 25 08:21 create_hash.sh
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Desktop
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Documents
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Downloads
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Music
drwxr-xr-x 2 user user      4096 Sep 25 08:19 Pictures
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Public
-rwxr--r-- 1 user user 139921507 Sep 25 07:54 rockyou.txt
-rw-r--r-- 1 user user      2015 Sep 25 07:48 target_hashes.txt
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Templates
drwxr-xr-x 2 user user      4096 Sep  6 16:35 Videos
```

4. In the file list, **rwx** means that the file owner (user) has read, write, and execute permissions.

5. Type the following to hash your passwords.



```
┌──(user㉿kali)-[~]
└─$ ./create_hash.sh
```

8. **cat** is a Linux command is a quick way to look at the contents of a text file. Check that your password hashes were created by typing the following command line in the terminal:

**cat target_hashes.txt**

You should see random numbers and letters.

9. Insert a screenshot of your hashes.

Click or tap here to enter text.

10. Kali Linux has numerous wordlists built right into it. To find them, use the following command line:

**locate rockyou**

If locate gives you an error → **sudo updatedb**

11. There will be several files. The **rockyou.txt** wordlist should show up like this:

```
/usr/share/wordlists/rockyou.txt.gz
```

12. Copy rockyou.txt.gz to your user folder.

```
cp /usr/share/wordlists/rockyou.txt.gz /home/user
```

13. **gzip** is a common file compression type in Linux. It is similar to zip. Extract the rockyou.txt file with the following command.

```
sudo gzip -d ./rockyou.txt.gz
```

14. Type the following command to confirm that rockyou.txt was extracted. **ls** is like **dir**, it gives a directory listing.

```
ls
```

15. Type the following command to look at the contents of the rockyou.txt password dictionary. Press any key to page through the file. It is 125 MB, there are a lot of passwords. These are all real passwords that people have used.

```
more rockyou.txt
```

16. Press **CTRL C** to break the command.

17. Insert a screen shot.

Click or tap here to enter text.

## Cracking the Hashes

1. Start Hashcat in the Kali Linux console with the following command line:

```
hashcat -h
```

You will see some basic help and examples. Some of the most important hashcat options are -m (the hashtype) and -a (attack mode). In general, we need to use both options in most password cracking attempts when using Hashcat.

Hashcat also has specifically designed rules to use on a wordlist file. The character list can be customized to crack the password(s).

We can now start cracking the hashes contained in the **target_hashes.txt** file. On a physical machine, Kali would use your GPU to crack the hashes.

**NOTE:** The Kali Linux virtual machine needs a minimum of 3072 MB of ram to run hashcat.

2. Use the following command line to crack the hashes. This command is all one line

```
hashcat -m 1400 -a 0 -o cracked.txt target_hashes.txt rockyou.txt --force --
deprecated-check-disable
```

### Explanation

- **-m 1400** designates the type of hash we are cracking (SHA256)
- **-a 0** designates a dictionary attack
- **-o cracked.txt** is the output file for the cracked passwords
- **target_hashes.txt** is our input file of hashes
- **rockyou.txt** is the path to the wordlist file for this dictionary attack.
- **-- force** is because we are using a virtual machine, which doesn't have a hardware graphics card.

When the program ends back at the command line, we have cracked 7 target hashes that were initially proposed. If you cracked one of your passwords, change it now.

3. Use **cat cracked.txt** to show your hashes and the results of your cracking.
4. Insert a screenshot of your results.

Click or tap here to enter text.

These passwords are weak, it does not take much effort or time to crack them. It is important to note that the simpler the password is, the easier it will be to crack.

**Lesson:** Make your password a long and complex one. 16 characters is the current recommendation. Avoid using obvious personal information; never reuse passwords and change them regularly.

## Assignment Submission

Attach this completed document to the assignment in Blackboard.