**Mastering Password Cracking with Hashcat**

**What is Hashcat?**

Hashcat is an open-source password recovery tool that uses advanced algorithms and high-speed techniques to crack passwords. It supports multiple attack modes, such as dictionary attacks, brute-force attacks, and hybrid attacks, allowing for flexibility in different scenarios.

**Features of Hashcat**

* **Supports Multiple Hash Types**: MD5, SHA1, SHA256, bcrypt, and more.
* **Multi-Platform**: Runs on Windows, Linux, and macOS.
* **High Performance**: Utilizes CPU and GPU acceleration.
* **Open-Source**: Freely available and customizable.

**Setting Up Hashcat**

**Prerequisites**

1. **Hardware**: A computer with a capable CPU or GPU.
2. **Operating System**: Windows, Linux, or macOS.
3. **Hashcat**: The software itself, which can be downloaded from the official website.

**Link:** <https://hashcat.net/hashcat/>

**Performing a Password Crack with Hashcat**

**Step 1: Identify the Hash Type**

Before you start cracking, we need to identify the type of hash we are dealing with. Common types include MD5, SHA1, SHA256, etc. Tools like hash-identifier or online services can help identify the hash type based on the hash string.

**Step 2: Prepare Your Environment**

1. **Open Terminal or Command Prompt**: Depending on your operating system, open the terminal (Linux/macOS) or command prompt (Windows).
2. **Navigate to Hashcat Directory**: Change to the directory where you extracted Hashcat.

**Step 3: Create a Hash File**

Save the hash we want to crack in a text file. For example, md5.txt. Ensure each hash is on a new line.

**Step 4: Choose an Attack Mode**

Hashcat supports several attack modes:

* **Dictionary Attack**: Uses a wordlist to guess the password.
* **Wordlist + Rule Attack**: Uses a wordlist and Rules file to guess the password.
* **Combinator Attac**k: combining words from two separate wordlists
* **Brute-Force Attack**: Tries all possible combinations.

we'll use a dictionary attack and Wordlist+rules attack in this guide.

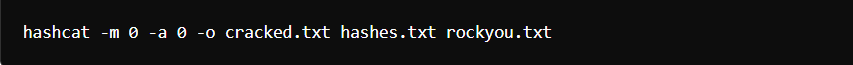
**Step 5: Download or Create a Wordlist**

You can use existing wordlists or create your own. Popular wordlists include:

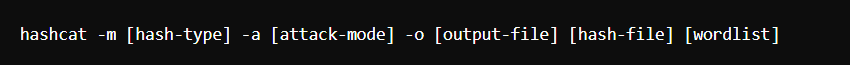
* [Rockyou.txt](https://github.com/brannondorsey/naive-hashcat/releases/download/data/rockyou.txt)
* [SecLists](https://github.com/danielmiessler/SecLists)

**Step 6: Run Hashcat**

Execute the Hashcat command with the appropriate options. The basic syntax is:



For example, to crack an MD5 hash using a dictionary attack:

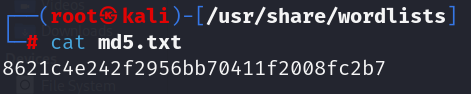


* **-m 0**: Specifies the hash type (0 for MD5).
* **-a 0**: Specifies the attack mode (0 for dictionary attack).
* **-o cracked.txt:** Output file for cracked passwords.
* **hashes.txt:** Input file containing hashes.
* **rockyou.txt:** Wordlist to use.

Further we are going to perform **Dictionary and wordlist+rule based attack.**

**Dictionary Attack:**

First we create a hash txt file.



Now we perform hashcat dictionary attack to crack hash



**Hashes Information**

* Hashes: Refers to the number of hashes being processed.
* **1 digests**: There is 1 hash (digest) in total.
* **1 unique digests**: Out of the total, there is 1 unique hash.
* **1 unique salts**: There is 1 unique salt associated with the hash. Salts are random data added to hashes to make them unique.

**Temperature abort trigger set to 90c**: Hashcat will abort the process if the device temperature reaches 90 degrees Celsius to prevent damage.

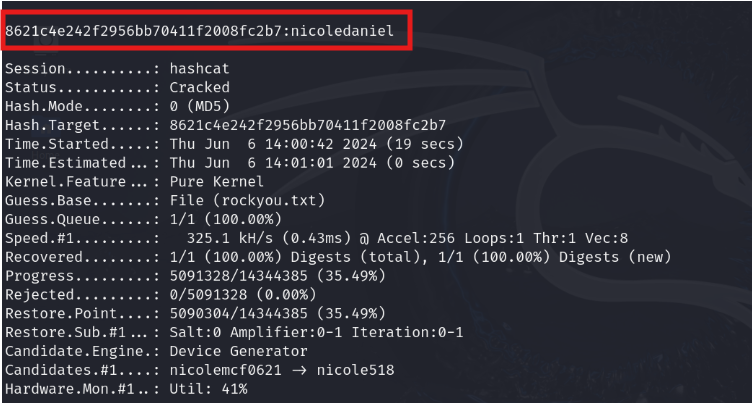
**Host memory required for this attack: 0 MB**: Indicates that no additional host memory (RAM) is required for this particular attack.



**Dictionary Cache Information**

**Dictionary cache built**: Indicates that the dictionary (wordlist) has been cached for quicker access.

* **Filename ..: wordlists/rockyou.txt**: The path to the wordlist file used.
* **Passwords .: 14344392**: The total number of passwords in the wordlist.
* **Bytes .....: 139921507**: The total size of the wordlist in bytes.



**8621c4e242f2956bb70411f2008fc2b7**: The hash being cracked.

**Hash.Mode**: The type of hash being cracked, which is MD5 (mode 0).

**Guess.Base**: The source of password candidates, which is the wordlist file rockyou.txt.

**Guess.Queue**: The number of guesses made relative to the total number of guesses. Here, 1 out of 1 (100.00%) guesses were made.

**Recovered**: The number of successfully cracked hashes:

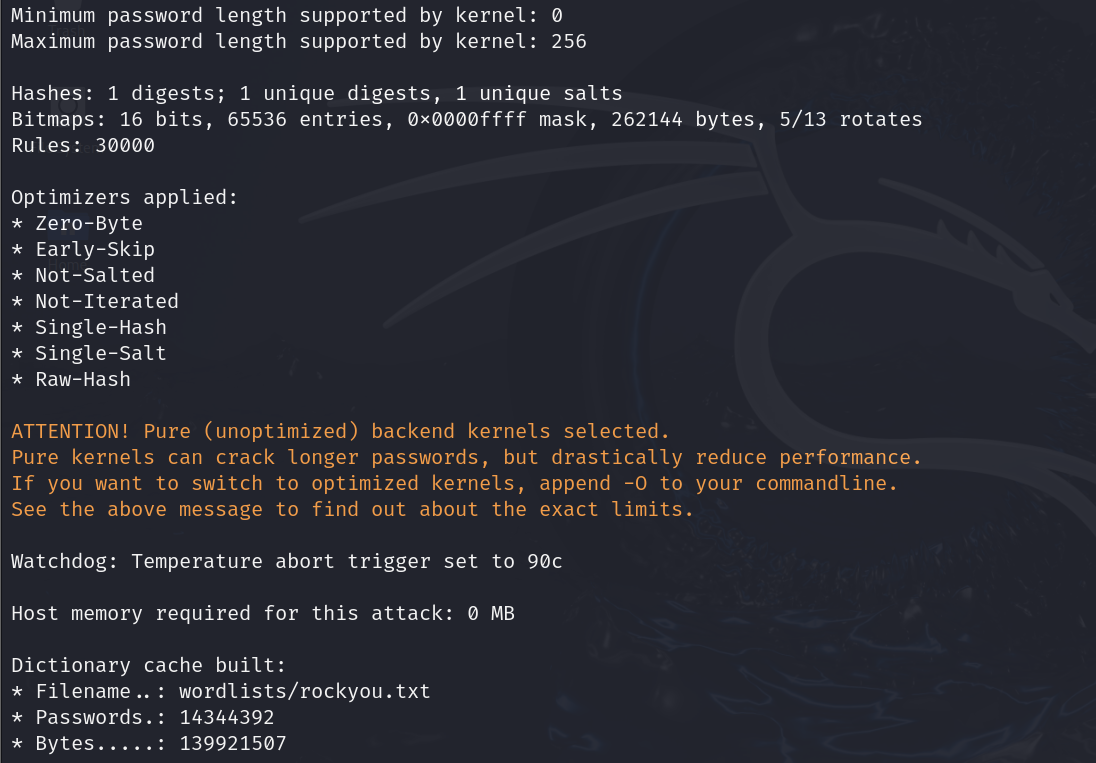
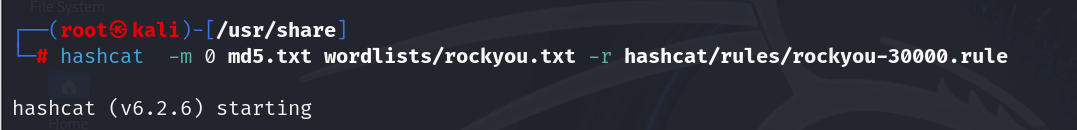
* **1/1 (100.00%) Digests (total)**: One out of one hash has been cracked.
* **1/1 (100.00%) Digests (new)**: Indicates all hashes were new and have been cracked.

**Salt:0**: Number of salts used.

**Amplifier:0-1**: Amplification range used.

**Iteration:0-1**: Iteration range used.

**Wordlist+Rule Attack:**



**Hashes Information**

* Hashes: Refers to the number of hashes being processed.
* **1 digests**: There is 1 hash (digest) in total.
* **1 unique digests**: Out of the total, there is 1 unique hash.
* **1 unique salts**: There is 1 unique salt associated with the hash. Salts are random data added to hashes to make them unique.

**Rules Information: Rules: 30000**

**Rules**: The number of rules being applied to the wordlist. In this case, 30,000 rules are used to modify the words in the wordlist.

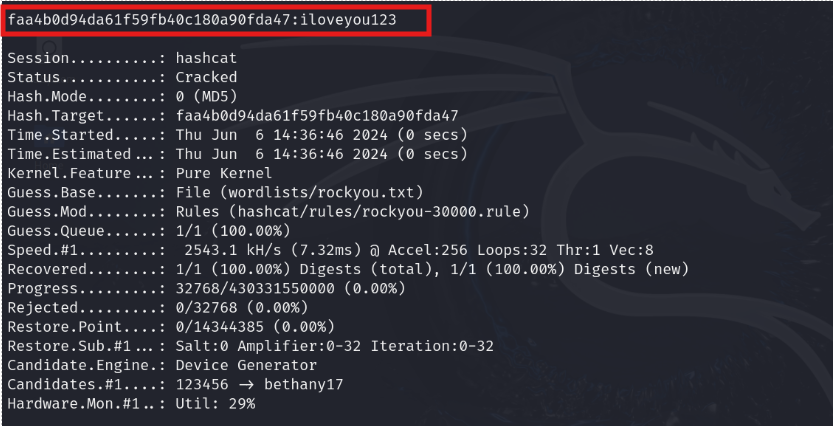
**Temperature abort trigger set to 90c**: Hashcat will abort the process if the device temperature reaches 90 degrees Celsius to prevent damage.

**Host memory required for this attack: 0 MB**: Indicates that no additional host memory (RAM) is required for this particular attack.

**Dictionary Cache Information**

**Dictionary cache built**: Indicates that the dictionary (wordlist) has been cached for quicker access.

* **Filename ..: wordlists/rockyou.txt**: The path to the wordlist file used.
* **Passwords .: 14344392**: The total number of passwords in the wordlist.
* **Bytes .....: 139921507**: The total size of the wordlist in bytes.



**faa4b0d94da61f59fb40c180a90fda47**: This is the MD5 hash that was cracked.

**Pure Kernel:** This indicates the type of kernel used by Hashcat. "Pure Kernel" refers to the pure computation kernel without any additional optimizations.

**Rules**: Indicates that a rule file was used.

**(hashcat/rules/rockyou-30000.rule)**: The path to the rule file that was applied to the wordlist to generate password candidates. Rules are used to transform words from the wordlist in various ways to increase the chances of cracking the password.

**1/1 (100.00%) Digests (total)**: Indicates that out of the total number of hashes (digests), one hash was processed, representing 100% of the total.

**1/1 (100.00%) Digests (new)**: Indicates that out of the new hashes provided, one hash was processed, representing 100% of the new hashes.

**Salt:0**: Indicates no salting was used for the hash.

**Amplifier:0-32**: The range of amplification factors used.

**Iteration:0-32**: The range of iterations used.

**Summary:**

Hashcat is a powerful password recovery tool that supports various attack modes, including dictionary attacks. A dictionary attack involves using a precompiled list of potential passwords (wordlist) to crack hashed passwords. Combining a wordlist with rules can enhance the effectiveness of the attack by modifying and expanding the list of potential passwords.

By following above steps and leveraging the capabilities of Hashcat, we can efficiently perform dictionary attacks and wordlists + rule to crack hashed passwords, enhancing our security assessment.