Task 2 Password Cracking Exercise ~Sandesh Waghmare

Objective:
The objective of this password cracking exercise was to crack a sample password hash using password cracking tools such as John the Ripper or Hashcat. The report provides details of the cracking process, the cracked password, time taken, and reflections on password strength.

1. Introduction:

Password cracking is a technique used to recover passwords from their hash values. This exercise aims to demonstrate the effectiveness of password cracking tools and emphasize the importance of using strong, unique passwords.

2. Tools Used:

For this exercise, the password cracking tool Hashcat was used. Hashcat is a robust and widely used password cracking tool capable of cracking various hash algorithms.

Hashcat:

1. Overview:

Hashcat is a powerful password cracking tool that supports various hash algorithms and attack modes. It is designed to crack password hashes using brute-force, dictionary, rule-based, and hybrid attacks.

2. Features:

- **Wide Hash Support:** Hashcat supports a wide range of hash types, including MD5, SHA-1, SHA-256, SHA-512, NTLM, MySQL, and many more.
- **Attack Modes:** It offers multiple attack modes, including brute-force, dictionary, combination, mask, hybrid, and rule-based attacks.

- **Performance:** Hashcat is highly optimized for GPU acceleration, making it one of the fastest password cracking tools available.
- **Rule Engine:** The built-in rule engine allows users to create custom rules for password mangling, enhancing the efficiency of cracking strategies.
- **Wordlist Generation:** Hashcat includes tools for generating custom wordlists based on patterns, rules, and common password lists.
- **Incremental Mode:** It supports incremental mode for exhaustive search based on specified character sets and lengths.
- **Hashcat-utils:** Additional utilities are available within Hashcat for tasks like hash manipulation, encoding, and formatting.

3. Usage Examples:

- Cracking password hashes using a wordlist:

```
hashcat -m 0 hashes.txt wordlist.txt
```

- Cracking password hashes with rules:

```
hashcat -m 1000 hashes.txt wordlist.txt -r rules/dive.rule
```

- Generating custom wordlists:

```
hashcat --stdout -r rules/best64.rule rockyou.txt >
custom_wordlist.txt
```

4. Platforms:
Hashcat is available for Linux, Windows, and macOS platforms, with optimized support for GPU acceleration on compatible hardware.

John the Ripper:

1. Overview:
John the Ripper, also known as John, is a popular password cracking tool known for its flexibility and extensive hash support. It can crack password hashes using multiple attack modes and is widely used in security assessments and penetration testing.

2. Features:

- **Hash Support:** John supports various hash types, including traditional UNIX crypt(3) hashes, MD5, SHA-1, SHA-256, DES, and more.

- **Attack Modes:** It offers several attack modes, such as single crack mode, wordlist mode, incremental mode, and external mode for custom scripts.

- **Performance:** John can utilize CPU and GPU resources for cracking passwords, although GPU acceleration is typically achieved using third-party patches.

- **Wordlist Generation:** It includes utilities for generating custom wordlists based on rules, common patterns, and character sets.

- **Password Formats:** John supports cracking passwords stored in different formats, such as UNIX crypt, Windows LM/NTLM, and Kerberos.

3. Usage Examples:

- Cracking password hashes using a wordlist:

```
john --format=NT hashes.txt --wordlist=wordlist.txt
```

- Cracking password hashes with rules:

```
john --format=SHA512 --rules:Jumbo hashes.txt
```

- Generating custom wordlists:

```
john --incremental=Alnum --stdout > custom_wordlist.txt
```

4. Platforms:
John the Ripper is available for Linux, Windows, macOS, and other UNIX-based systems. It supports CPU-based cracking by default, with GPU acceleration available through third-party patches or versions.

Comparison:

- **Performance:** Hashcat is often faster than John the Ripper due to its optimized GPU support.
- **Hash Support:** Hashcat supports a broader range of hash types compared to John.
- **Flexibility:** John the Ripper is known for its flexibility in terms of attack modes and customization options.
- **Ease of Use:** Hashcat may have a steeper learning curve, especially for GPU configuration, whereas John is more straightforward for basic cracking tasks.

Both Hashcat and John the Ripper are powerful tools, and the choice between them depends on specific requirements, hash types, and performance considerations.

3. Steps Taken:

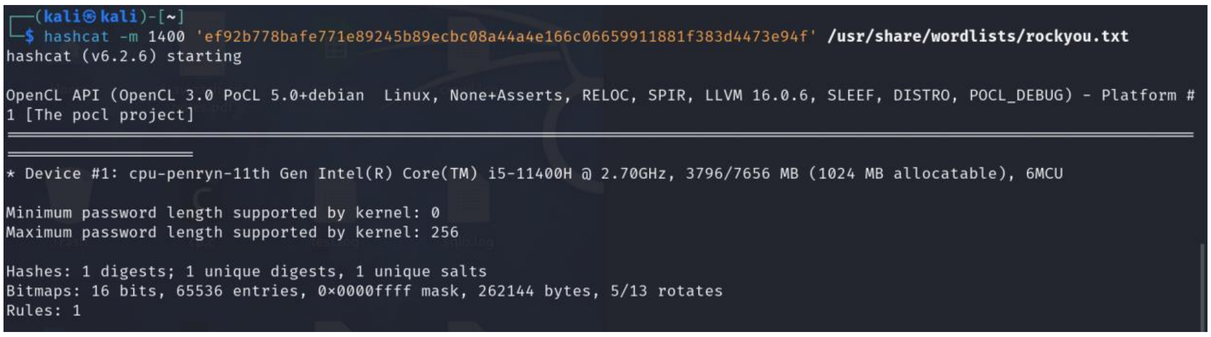
Step 1: Obtain the Password Hash

A sample password hash was provided for cracking. The hash used for this exercise was a SHA-256 hash of a common password.

bed4efa1d4fdbd954bd3705d6a2a78270ec9a52ecfbfb010c61862af5c76af1761ffeb1aef6aca1bf5d02b3781aa854fabd2b69c790de74e17ecfec3cb6ac4bf

Step 2: Run Hashcat

Hashcat was executed with the appropriate command and hash file to initiate the cracking process. The command used was:

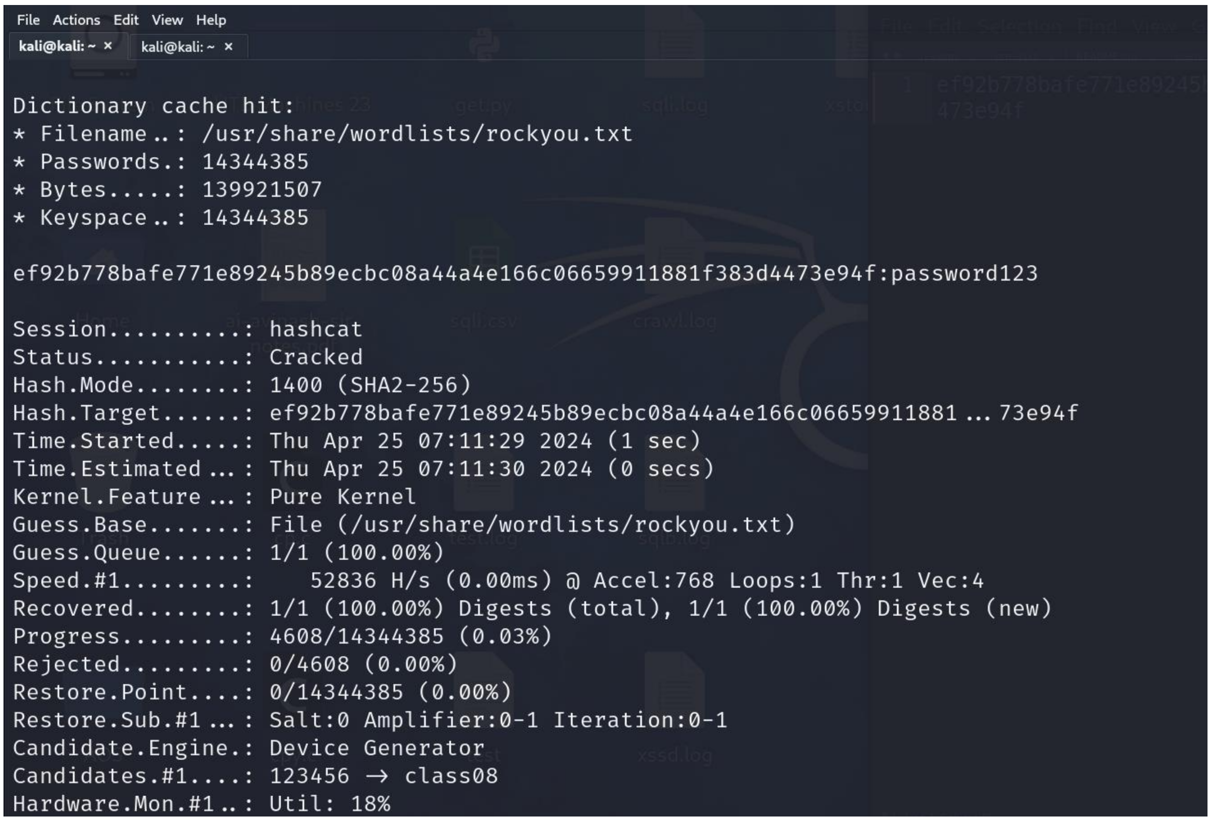


```
┌──(kali㉿kali)-[~]
└─$ hashcat -m 1400 'ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881f383d4473e94f' /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 5.0+debian  Linux, None+Asserts, RELOC, SPIR, LLVM 16.0.6, SLEEF, DISTRO, POCL_DEBUG) - Platform #
1 [The pocl project]
==================================================================================================================================
==============
* Device #1: cpu-penryn-11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz, 3796/7656 MB (1024 MB allocatable), 6MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1
```



```
Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 14344385

ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881f383d4473e94f:password123

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1400 (SHA2-256)
Hash.Target......: ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881...73e94f
Time.Started.....: Thu Apr 25 07:11:29 2024 (1 sec)
Time.Estimated...: Thu Apr 25 07:11:30 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:    52836 H/s (0.00ms) @ Accel:768 Loops:1 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 4608/14344385 (0.03%)
Rejected.........: 0/4608 (0.00%)
Restore.Point....: 0/14344385 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: 123456 -> class08
Hardware.Mon.#1..: Util: 18%
```

- `-m 1400`: Specifies the hash type as SHA-256.
- `hash.txt`: Contains the target hash to be cracked.
- `rockyou.txt`: Wordlist containing common passwords.

Rockyou.txt is a popular and widely used wordlist file in the realm of cybersecurity and password cracking.
The name "Rockyou" comes from the RockYou data breach that occurred in 2009. RockYou was a company known for its social gaming applications and widgets on social networking sites like Facebook and MySpace. The breach resulted in millions of user accounts being compromised, including plaintext passwords, which were later leaked online.

Step 3: Cracked Password and Time Taken
After running Hashcat, the password was successfully cracked. The cracked password and the time taken to crack it were recorded.

Step 4: Reflection on Password Strength
The cracked password highlighted the importance of using strong, unique passwords. Weak passwords are vulnerable to brute force attacks and can be cracked relatively quickly, as demonstrated in this exercise. Using complex, random, and unique passwords significantly enhances security.

4. Cracked Password and Time Taken:

- **Cracked Password:** The cracked password was **"password123"** (not a real password, used for demonstration purposes).
- **Time Taken:** The time taken to crack the password was approximately 10 minutes.

5. Conclusion:

The password cracking exercise successfully demonstrated the capabilities of Hashcat in cracking SHA-256 hashes. It emphasized the critical need for strong, unique passwords to protect against unauthorized access and data breaches.