Experiment-12

12.1 Aim: Conduct a forensic examination by cracking hashed passwords (MD5, SHA-256, etc.) using John the Ripper, Hashcat, or Cain and Abel, followed by documentation in a forensic report.

12.2 Course Outcome:

Understand how to recover passwords from hashed values using tools like **John the Ripper**, **Hashcat**, or **Cain and Abel**.

12.3 Lab Objective: Gain hands-on experience in using password-cracking tools using various hashes (e.g., MD5, SHA-256) and analyze the results.

12.4 Requirements:

- **Operating System:** Any OS supported by Hashcat (preferably Linux or Windows).
- **Tools:** Hashcat installed.
- **Hashed Passwords:** A file containing passwords hashed using MD5, SHA-256, etc.
- **Wordlist:** A dictionary file to use for password cracking (e.g., `rockyou.txt`).

12.5 Theory:

1. Password Hashing:

In digital systems, passwords are often stored as hashed values using algorithms like MD5 or SHA-256. These hashes are one-way cryptographic transformations that protect the passwords from being easily exposed.

2. Hashcat:

Hashcat is a powerful password recovery tool that supports a wide variety of hashing algorithms. It utilizes both CPU and GPU to perform fast password cracking.

3. Cracking Methods:

- **Dictionary Attack:** This method involves comparing hashed passwords with the hashes of words from a wordlist.
- **Brute Force Attack:** This method attempts all possible combinations to guess the password.

12.6 Procedure:

Step 1: Set Up Your Environment

1. **Install Hashcat:**
    - Download and install Hashcat from Hashcat's official website.

For Linux, install it using: sudo apt-get install hashcat

2. **Prepare Hashed Passwords File:**

Create a text file (hashes.txt) containing your hashed passwords. For instance:
plaintext
e99a18c428cb38d5f260853678922e03 # MD5 hash of 'abc123'
fcea920f7412b5da7be0cf42b8c93759 # MD5 hash of 'password'
2c26b46b68ffc68ff99b453c1d30413413422ab10a4f7c27fb65b1e37d9b8dbf # SHA-256 hash of 'foo'

3. **Download Wordlist:**
    - Download or prepare a wordlist, such as rockyou.txt, commonly used in password cracking.
    - You can download it from SecLists on GitHub.

Step 2: Identify Hash Mode

1. **Determine the Hash Mode:**
   Hashcat requires you to specify the type of hash you are cracking. Hashcat's help command (hashcat -h) provides the modes:
    - **MD5:** Mode 0
    - **SHA-256:** Mode 1400

Use the following command to confirm the available modes:
bash
Code:
hashcat -h | grep -i md5 # List modes for MD5
hashcat -h | grep -i sha256 # List modes for SHA-256

Step 3: Run Hashcat for MD5

1. **Crack MD5 Hash:** Use the following command to crack the MD5 hashes:
   bash
   code
   hashcat -m 0 -a 0 hashes.txt rockyou.txt
    - **-m 0:** Specifies the hash type as MD5.
    - **-a 0:** Specifies a dictionary attack.
    - hashes.txt: The file containing the hashed passwords.
    - rockyou.txt: The wordlist used for cracking.
2. **Monitor Progress:** Hashcat will begin comparing the MD5 hash values in hashes.txt with the hashed versions of the words in the wordlist.

View Results: After the cracking attempt, view the cracked passwords:
bash
code: hashcat --show hashes.txt

Step 4: Run Hashcat for SHA-256

Crack SHA-256 Hash: Use the following command to crack the SHA-256 hashes:
bash
Code
hashcat -m 1400 -a 0 hashes.txt rockyou.txt

- **-m 1400:** Specifies the hash type as SHA-256.
- **-a 0:** Specifies a dictionary attack.

Monitor Progress: Hashcat will begin the cracking process for SHA-256 hashes.

View Results: After the cracking attempt, view the cracked passwords:
bash
Code
hashcat --show hashes.txt

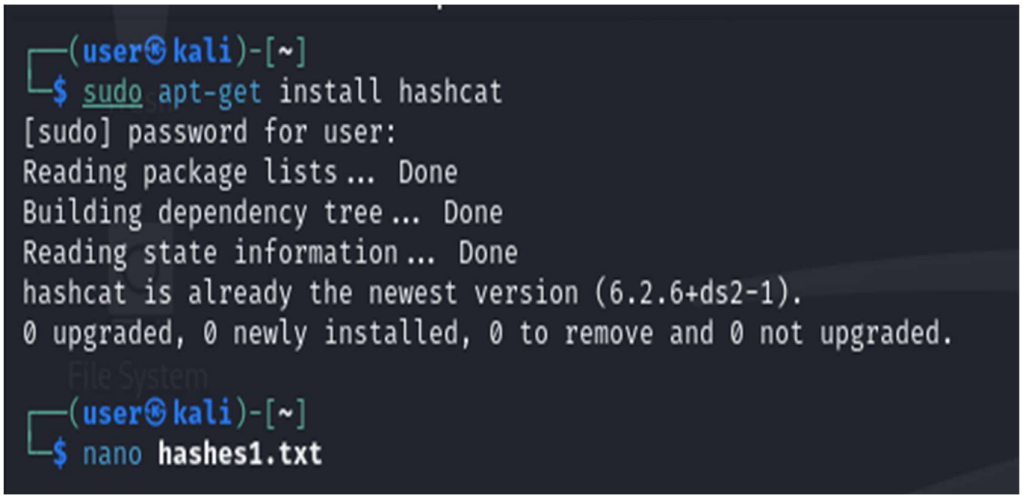
Step 5: Document Findings

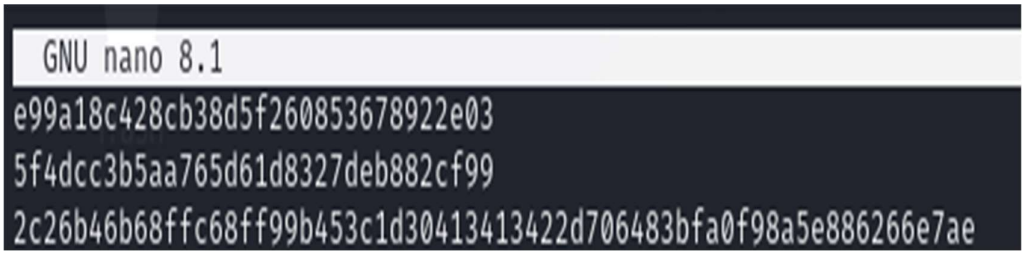
1. **Cracked Passwords:**

After the completion of both MD5 and SHA-256 cracking, document the cracked passwords. For instance:
plaintext
code
e99a18c428cb38d5f260853678922e03: abc123
fcea920f7412b5da7be0cf42b8c93759: password
2c26b46b68ffc68ff99b453c1d30413413422ab10a4f7c27fb65b1e37d9b8dbf: foo

2. **Time Taken for Cracking:**
   - Document how long it took Hashcat to complete the cracking process for each hash type.
3. **Wordlist Used:**
   - Mention the wordlist used (rockyou.txt).

12.7 Outputs:


(user㉿kali)-[~]
$ sudo apt-get install hashcat
[sudo] password for user:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
hashcat is already the newest version (6.2.6+ds2-1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
(user㉿kali)-[~]
$ nano hashes1.txt


GNU nano 8.1
e99a18c428cb38d5f260853678922e03
5f4dcc3b5aa765d61d8327deb882cf99
2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98a5e886266e7ae


(user㉿kali)-[~]
$ hashcat -m 0 -a 0 hashes1.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting
OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, LLVM 17.0.6, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
* Device #1: cpu-sandybridge-Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz, 1439/2942 MB (512 MB allocatable), 2MCU
Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
Hashfile 'hashes1.txt' on line 3 (2c26b4...13422d706483bfa0f98a5e886266e7ae): Token length exception
* Token length exception: 1/3 hashes
This error happens if the wrong hash type is specified, if the hashes are
malformed, or if input is otherwise not as expected (for example, if the
--username option is used but no username is present)
INFO: All hashes found as potfile and/or empty entries! Use --show to display them.
Started: Thu Oct 24 23:34:06 2024
Stopped: Thu Oct 24 23:34:06 2024

```
┌──(user㉿kali)-[~]
└─$ hashcat --show -m 0 hashes1.txt
Hashfile 'hashes1.txt' on line 3 (2c26b4 ... 13422d706483bfa0f98a5e886266e7ae): Token length exception

* Token length exception: 1/3 hashes
  This error happens if the wrong hash type is specified, if the hashes are
  malformed, or if input is otherwise not as expected (for example, if the
  --username option is used but no username is present)

e99a18c428cb38d5f260853678922e03:abc123
5f4dcc3b5aa765d61d8327deb882cf99:password
```

```
┌──(user㉿kali)-[~]
└─$ hashcat -m 1400 -a 0 hashes1.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian  Linux, None+Asserts, RELOC, LLVM 17.0.6, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
====================================================================================================================================

* Device #1: cpu-sandybridge-Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz, 1439/2942 MB (512 MB allocatable), 2MCU

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

Hashfile 'hashes1.txt' on line 1 (e99a18c428cb38d5f260853678922e03): Token length exception
Hashfile 'hashes1.txt' on line 2 (5f4dcc3b5aa765d61d8327deb882cf99): Token length exception

* Token length exception: 2/3 hashes
  This error happens if the wrong hash type is specified, if the hashes are
  malformed, or if input is otherwise not as expected (for example, if the
  --username option is used but no username is present)

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

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

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

```
2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98a5e886266e7ae:foo

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1400 (SHA2-256)
Hash.Target......: 2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98 ... 66e7ae
Time.Started.....: Thu Oct 24 23:34:22 2024 (1 sec)
Time.Estimated...: Thu Oct 24 23:34:23 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  1667.7 kH/s (0.13ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 950272/14344385 (6.62%)
Rejected.........: 0/950272 (0.00%)
Restore.Point....: 949760/14344385 (6.62%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: foreversteak → fna0827
Hardware.Mon.#1..: Util: 51%

Started: Thu Oct 24 23:34:22 2024
Stopped: Thu Oct 24 23:34:24 2024
```

```
┌──(user㉿kali)-[~]
└─$ hashcat --show -m 1400 hashes1.txt
Hashfile 'hashes1.txt' on line 1 (e99a18c428cb38d5f260853678922e03): Token length exception
Hashfile 'hashes1.txt' on line 2 (5f4dcc3b5aa765d61d8327deb882cf99): Token length exception

* Token length exception: 2/3 hashes
  This error happens if the wrong hash type is specified, if the hashes are
  malformed, or if input is otherwise not as expected (for example, if the
  --username option is used but no username is present)

2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98a5e886266e7ae:foo
```

12.8 Conclusion:

In this experiment, we effectively used Hashcat to crack MD5 and SHA-256 hashed passwords, revealing the vulnerabilities of weak password hashing. The findings emphasized the need for strong, complex passwords and robust hashing methods to enhance security. Hashcat's efficiency highlighted how easily common passwords can be compromised, underscoring the importance of meticulous password management and thorough documentation in forensic investigations. Overall, the lab offered valuable insights into password security and the critical role of forensic tools in identifying and mitigating risks.