Task 4: Password Security & Authentication Analysis

Name: - Ch Rohit Kumar

Gmail: -rohitkumarrrr20@gmail.com

Clg: -Aditya University

INTRODUCTION

Passwords are the most common authentication method used in computers, websites, and mobile applications. But many users still choose weak passwords like ***123456, password, admin, qwerty****, etc. Attackers take advantage of weak passwords using password cracking tools like* ***Hashcat*** *and* ***John the Ripper****.*

In cybersecurity, password security analysis helps us understand:

- *How passwords are stored*
- *How attackers crack passwords*
- *How to prevent password attacks using strong authentication methods*

This report explains password hashing, hash identification, cracking techniques, and recommendations for strong authentication.

2. OBJECTIVES

The main objectives of this task are:

1. *Understand how passwords are stored securely using hashes*
2. *Identify different types of password hashes*
3. *Generate sample password hashes*
4. *Crack weak hashes using dictionary and brute force attacks*
5. *Compare Hashcat and John the Ripper performance*
6. *Learn why weak passwords fail easily*
7. *Study MFA and its importance*
8. *Provide recommendations for secure authentication*

3. TOOLS USED

Primary Tools

➢ *Hashcat*

- *Fast password recovery tool*
- *Uses GPU/CPU for cracking*
- *Supports many hash types (MD5, SHA, NTLM, bcrypt, etc.)*

➢ *John the Ripper (JTR)*

- *Powerful password cracker*
- *Works well with CPU*
- *Supports wordlist and brute force cracking*

THEORY / BACKGROUND

➢ *What is Hashing?*

*Hashing is a process of converting input data (password) into a fixed-length string called a **hash**.*
Example:

- *Password: hello123*
- *Hash: some_long_value*

Important: *Hashing is **one-way**, meaning we cannot directly reverse it.*

Common Hash Types

Different systems use different algorithms.

➢ *MD5*

- *32 characters hex*
- *Very fast → easy to crack*
  Example: 5f4dcc3b5aa765d61d8327deb882cf99 (password)

➢ *SHA-1*

- *40 characters hex*
- *Also weak today*
  Example: 5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8

➢ *bcrypt*

- *Slow hash algorithm*
- *Much harder to crack*
  Example starts like: $2y$ or $2b$

Modern systems prefer bcrypt, scrypt, Argon2.

METHODOLOGY / PRACTICAL WORK

➢ *Step 1: Generate Password Hashes*

We created some sample passwords and generated their hashes.

Example weak passwords used:

- *123456*
- *password*
- *admin123*
- *qwerty*
- *welcome123*

Example strong passwords used:

- *Rohith@2026#Secure*
- *Mfa+Strong!Pass99*

Hashes generated (example MD5):

➢ *Identify Hash Type*

We identified hashes by:

- *Length*
- *Characters used (hex / base64)*
- *Patterns (like $2y$ for bcrypt)*

Examples:

- *32 hex → likely MD5*
- *40 hex → likely SHA-1*
- *$2y$ → bcrypt*

➢ *Step 3: Cracking Hashes using Hashcat*

We used a wordlist dictionary attack first.

Dictionary Attack

A dictionary attack tries common passwords from a file like:

- *rockyou.txt*
- *custom wordlist*

Example command format (for learning):

```
hashcat -m <hash-type> -a 0 hashes.txt wordlist.txt
```

Then results show cracked passwords if found.

✅ ***Observation:*** *Weak passwords were cracked quickly using dictionary attacks.*

➢ *Step 4: Cracking Hashes using John the Ripper*

We performed cracking using JTR.

Example format:

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

To show cracked passwords:

```
john --show hashes.txt
```

Observation: *John also cracked weak hashes easily, especially MD5 and SHA-1.*

ATTACK TYPES

➢ *Dictionary Attack*

- *Uses list of common passwords*
- *Faster than brute force*
- *Works if password is predictable*

✅ *Example passwords cracked:*

- *password*
- *admin123*
- *welcome123*

➢ *Brute Force Attack*

- *Tries every combination*
- *Very slow for long complex passwords*
- *But guaranteed if time is enough*

Example:

- *For a 6-digit number password → easy*
- *For a 12-character strong password → extremely hard*

WHY WEAK PASSWORDS FAIL

Weak passwords fail because:

1. *They are short (less characters)*
2. *They use common patterns (123, qwerty)*
3. *They contain dictionary words*
4. *They lack symbols and complexity*
5. *They are reused across multiple sites*

This makes them easy to crack by Hashcat or John.

MULTI FACTOR AUTHENTICATION (MFA)

➢ *What is MFA?*

*MFA means using **2 or more authentication steps**:*

1. *Something you know (password)*
2. *Something you have (OTP, phone)*
3. *Something you are (fingerprint)*

➢ *Why MFA is Important?*

Even if attacker cracks password, they still need:

- *OTP / Authenticator code*

MFA reduces risk of account takeover significantly.

RECOMMENDATIONS (BEST PRACTICES)

➢ *Strong Password Rules*

A strong password should:
*Be **12–16 characters** long*
Use uppercase + lowercase
Use numbers + special symbols
Avoid name, date of birth, mobile number
Avoid common words
Unique for every account

Example strong password:
Rohith@2026#Secure

➢ *Secure Storage Methods*

Organizations should:
*Use **bcrypt / Argon2** hashing*
*Add **salt** to passwords*
Enable account lockout after attempts
Use rate limiting and CAPTCHA
Use MFA for all accounts

CONCLUSION

*This password security analysis report demonstrated how attackers can crack weak password hashes using tools like **Hashcat** and **John the Ripper**. It showed that common passwords can be recovered quickly using dictionary attacks, while strong passwords combined with modern hashing algorithms and MFA are difficult to break.*

Therefore, organizations and individuals should implement strong password policies, secure hashing algorithms like bcrypt/Argon2, and enable MFA to prevent password-based attacks.