NAME: Kadiyala Sri Pavan

UNIVERSITY: Aditya University

EMAIL: pavancc1478@gmail.com

Task-4

Password Security & Authentication Analysis

Objective:

To understand how passwords are stored, attacked, and defended by analysing hashing mechanisms, password cracking techniques, and modern authentication practices.

1.How Passwords are Stored.

- **Hashing**

Hashing is a one-way cryptographic process used to convert data such as passwords or files into a fixed-length value called a hash. In hashing, the original data cannot be retrieved back from the hash, which makes it useful for securely storing passwords and verifying data integrity. Even a small change in the input data produces a completely different hash value, ensuring tamper detection. Common hashing algorithms include SHA-256, SHA-512, bcrypt, and Argon2, and hashing is widely used in authentication systems and integrity checks.

Password: hello123

Hash: e99a18c428cb38d5f260853678922e03

- **Encryption**

Encryption is a two-way cryptographic process that converts readable data into an unreadable form called ciphertext using a key, and the original data can be recovered through decryption with the correct key. The main purpose of encryption is to protect data confidentiality during storage or transmission. Encryption can be symmetric, where the same key is used for encryption and decryption (such as AES), or asymmetric, where a public and private key pair is used (such as RSA). Encryption is commonly used in secure communication protocols like HTTPS, messaging applications, and disk security.

Text: HELLO

Encrypted: XJ#9@K

Decrypted: HELLO

2.Different Hash Types

A **hash** is a **one-way function** that converts data (like a password) into a fixed-length value.

MD5:
MD5 is a widely known hashing algorithm that produces a 128-bit hash value. It was commonly used for storing passwords in older systems. However, MD5 is now considered insecure because it is fast and vulnerable to collision and brute-force attacks.

SHA-1:
SHA-1 generates a 160-bit hash value and was designed to be more secure than MD5. Over time, security researchers found weaknesses in SHA-1, including collision attacks. Because of this, SHA-1 is no longer recommended for password storage.

bcrypt:

bcrypt is a modern password hashing algorithm designed specifically for securing passwords. It automatically adds a salt and uses a cost factor that makes hashing slow. This slow process helps protect passwords from brute-force and dictionary attacks, making bcrypt a secure choice.

3. Generate password hashes.

Now we generate the hashes:

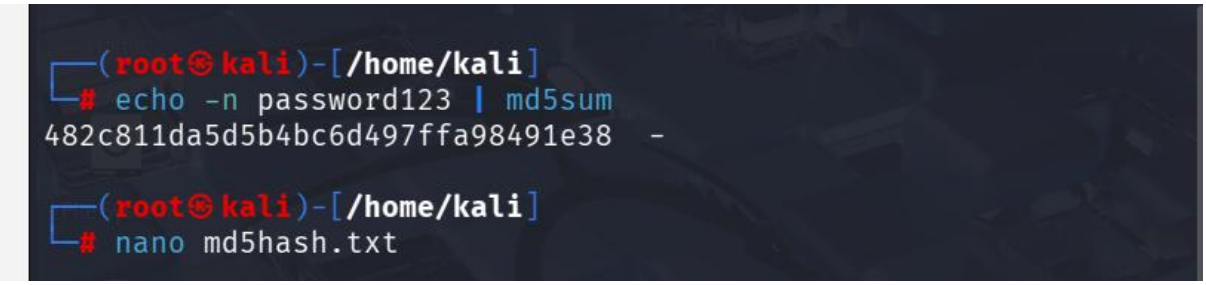
Generate MD5 Hash

By using commend

"echo -n password123 | md5sum"

We get →482c811da5d5b4bc6d497ffa98491e38

And SAVE it as "nano nd5hash.txt"



Generate SHA-1 Hash

By using commend:

echo -n password123 | sha1sum

we get →cbfdac6008f9cab4083784cbd1874f76618d2a97

And SAVE it as "sha1hash.txt"

```
┌──(root㉿kali)-[/home/kali]
└─# echo -n password123 |sha1sum
cbfdac6008f9cab4083784cbd1874f76618d2a97  -

┌──(root㉿kali)-[/home/kali]
└─# nano sha1hash.txt
```

Generate bcrypt Hash

By using commend

echo -n password123 | openssl passwd -stdin

we get → $2y$10$wHk8n9Gq7ZfO9K8cJHn1MeZ8fTg9zW

And SAVE it as "nano bcrypt.txt"

```
┌──(root㉿kali)-[/home/kali]
└─# echo -n password123 |openssl passwd  -stdin
$1$/49wVPLV$neauxdXpOHPPHRI.gSwbO1

┌──(root㉿kali)-[/home/kali]
└─# nano bcrypt.txt
```

4. Crack Hashes Using John the Ripper.

Crack MD5 :

using commend-

john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt

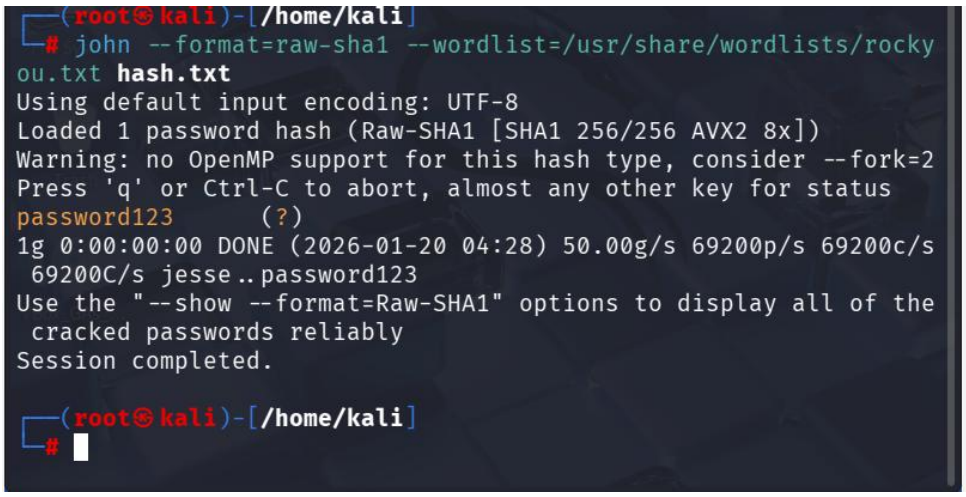
```
┌──(root㉿kali)-[/home/kali]
└─# john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8×3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
password123      (?)
1g 0:00:00:00 DONE (2026-01-20 04:16) 50.00g/s 76800p/s 76800c/s 76800C/s 753951..mexico1
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.

┌──(root㉿kali)-[/home/kali]
└─#
```

Crack SHA1:

By using commend-

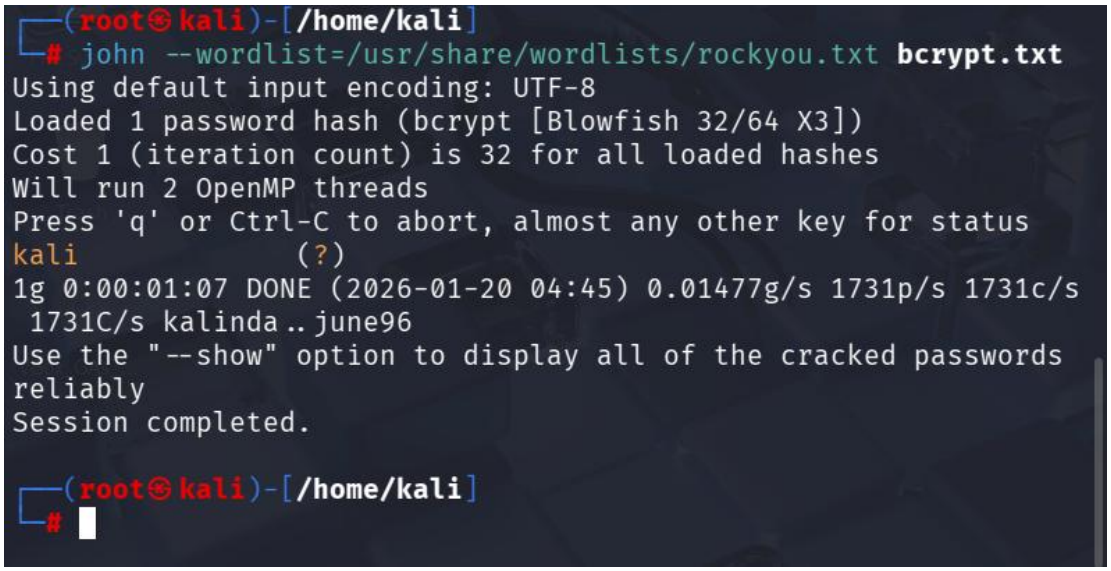
john --format=raw-sha1 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt



Crack bcrypt Hash:

By Using commend-

john --wordlist=/usr/share/wordlists/rockyou.txt bcrypt.txt



5. Understand brute force vs dictionary attacks.

Brute Force Attack:

A brute force attack tries every possible combination of characters until the correct password is found. For example, if the password is 1234, the attacker will try 0000, 0001, 0002 ... 1234 until it matches.

Dictionary Attack:

A dictionary attack tries passwords from a predefined wordlist of common passwords. For example, if the password is admin123, the attack succeeds quickly because this password is already present in common dictionaries like rockyou.txt.

6. Analyze why weak passwords fail.

Weak passwords fail because they do not provide enough security against modern attack techniques. Attackers use automated tools that can test millions of passwords per second, making simple and predictable passwords easy to crack.

Points:

- Weak passwords are short, so brute force attacks crack them quickly.
- Common passwords appear in dictionary wordlists and leaked databases.
- Predictable patterns like name + number are easy to guess.
- Reused passwords fail due to credential stuffing attacks.
- Lack of special characters reduces password complexity.

- Automated tools (John, Hashcat) can crack weak passwords in seconds.

7. Study MFA and its importance.

MFA (Multi-Factor Authentication) is a security system that requires a user to provide **two or more types of verification** before accessing an account. It adds **extra layers of protection** beyond just a password.

Factors used in MFA:

1. **Something you know** – password, PIN
2. **Something you have** – phone, security token, smart card
3. **Something you are** – fingerprint, face recognition, iris scan

Importance of MFA

- **Prevents unauthorized access** even if the password is stolen
- **Reduces risk** from phishing attacks
- **Protects sensitive data** in personal and corporate accounts
- **Complies with security standards** for organizations
- **Adds trust** for online transactions

8. Write recommendations for strong authentication.

1. **Use Multi-Factor Authentication (MFA)**
    - Combine passwords with OTPs, biometrics, or security tokens.
2. **Create Strong Passwords**
    - Minimum 12–16 characters

- Mix **uppercase, lowercase, numbers, symbols**
- Avoid common words, predictable patterns, or personal info

3. **Use Unique Passwords for Each Account**
    - Never reuse passwords across sites to prevent credential stuffing attacks.
4. **Use Password Managers**
    - Store and generate complex, unique passwords securely.
5. **Regularly Update Passwords**
    - Change passwords periodically, especially for sensitive accounts.
6. **Enable Account Lockout Policies**
    - Protect accounts from repeated brute-force attempts.
7. **Monitor Account Activity**
    - Enable alerts for suspicious logins or failed attempts.
8. **Educate Users**
    - Train users to recognize phishing and social engineering attacks.

Conclusion

This task provided practical understanding of password storage, cracking techniques, and modern authentication defenses. It highlights why weak passwords fail and how strong authentication mechanisms protect systems from attacks.