Task 4: Password Security & Authentication Analysis

1) How Passwords Are Stored (Hashing vs Encryption)

Passwords should never be stored in plain text because anyone who gains access to the database can directly read them. Instead, secure systems use **hashing** or sometimes **encryption**.

a. Encryption

Encryption converts a password into an unreadable format using a **secret key,** and it can be reversed back to the original password using the same key. If the encryption key is compromised, all passwords become exposed. Because of this risk, encryption is not recommended for storing passwords.

Example:

```
password123 → Encrypted text(encrypted using the key) → password123 (after decryption)
```

b. Hashing

Hashing is a one-way process where the password is converted into a fixed-length value called a hash. The original password cannot be recovered from the hash. During login, the entered password is hashed again and compared with the stored hash.

Example:

```
password123 → 482c811da5d5b4bc6d497ffa98491e38
```

From my research, I understood that **hashing is the safest and standard method** used to store passwords.

2) Identifying Different Hash Types (MD5, SHA-1, bcrypt)

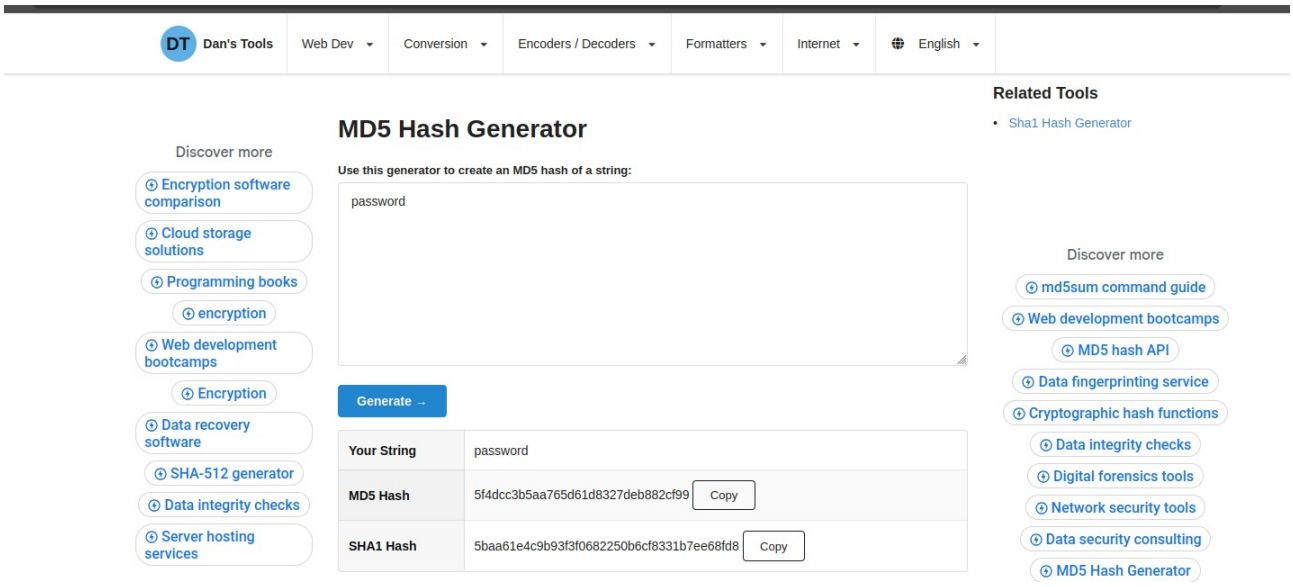
Different hashing algorithms generate different types of hashes, which can be identified using their length and structure.

a. MD5

- Produces a 32-character hexadecimal hash
- Very fast and insecure

Example:

`password → 5f4dcc3b5aa765d61d8327deb882cf99`

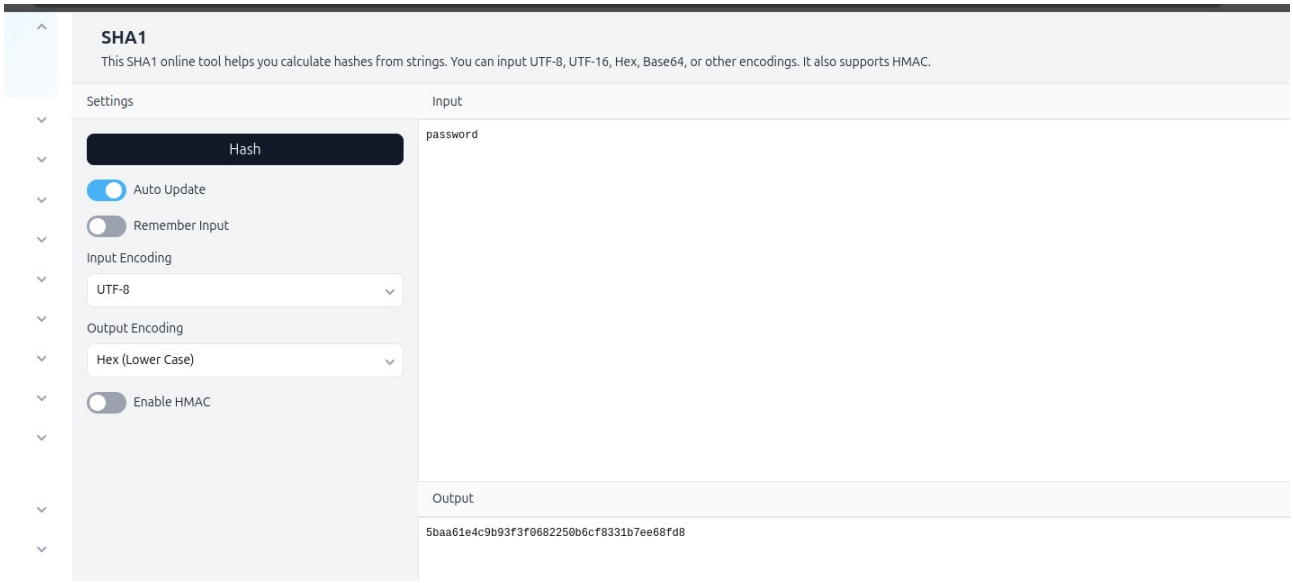


b. SHA-1

- Produces a 40-character hexadecimal hash
- More secure than MD5 but now broken

Example:

`password → 5baa61e4c9b93f3f0682250b6cf8331b`



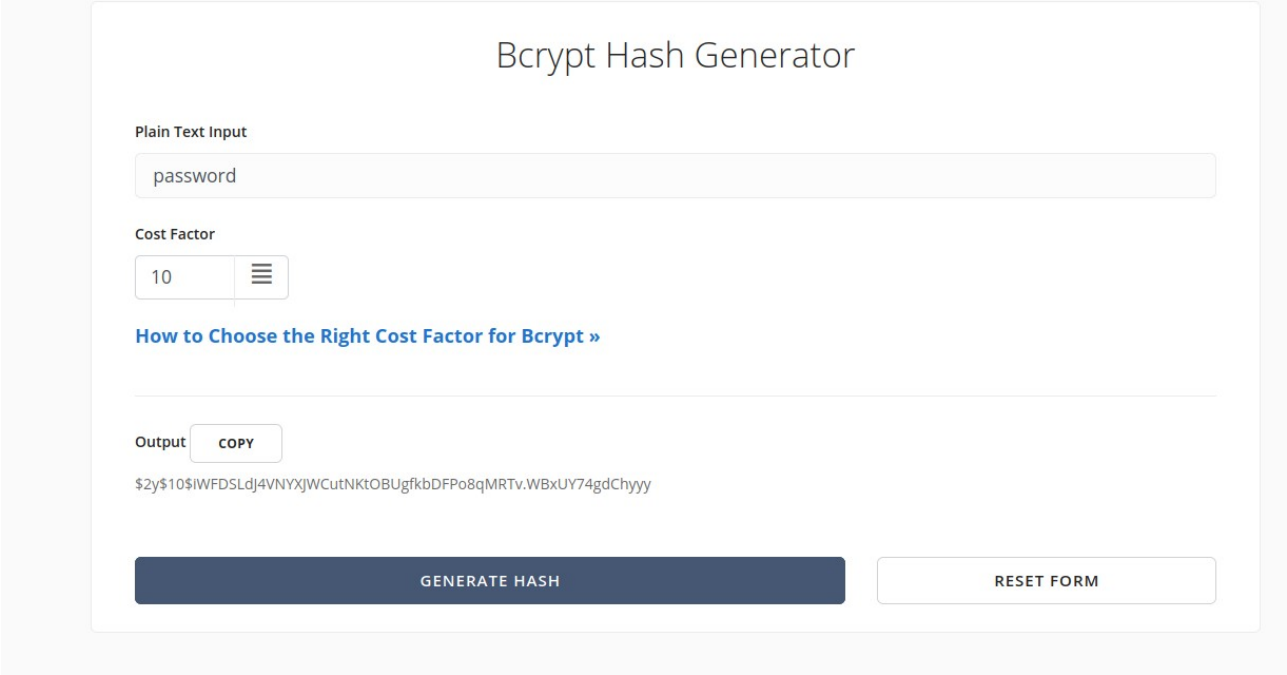
c. bcrypt

- Produces a long hash with special prefixes like `$2b$`

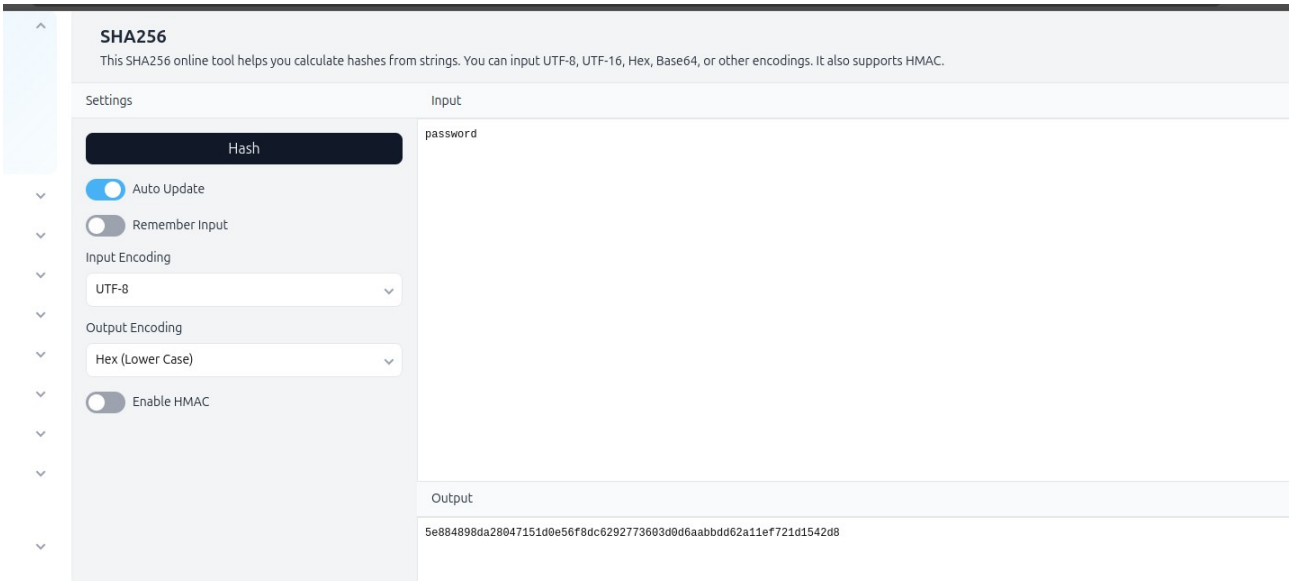
- Uses salt and multiple rounds
- Very secure

Example:

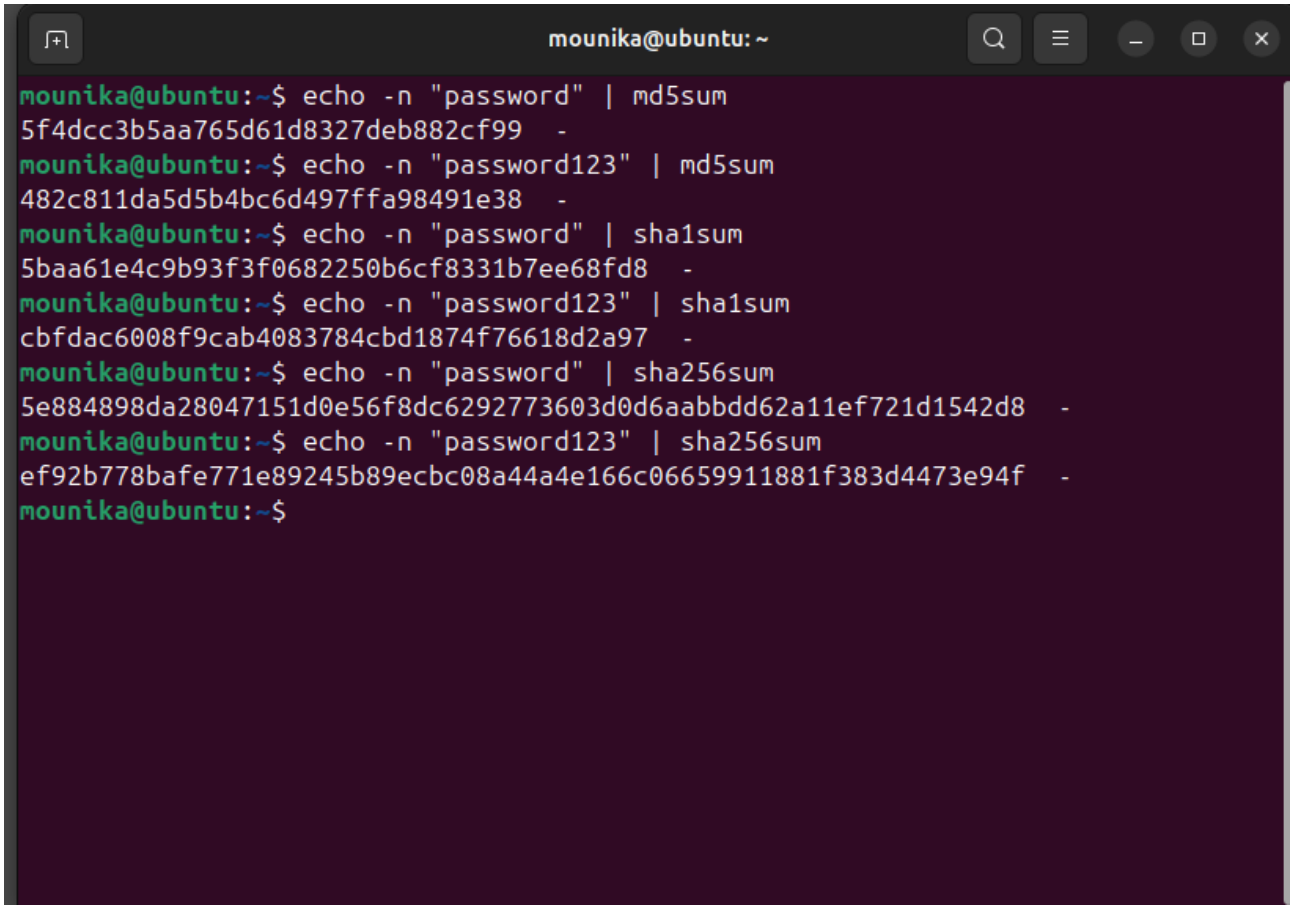
`$2b$12$eImiTXuWVxfM37uY4JANjQ`



SHA256



All in linux terminal:



From my understanding, **MD5 and SHA-1 are insecure**, while **bcrypt is recommended for password storage**.

3) Generating Password Hashes

Generating a password hash means applying a hashing algorithm to a password.

Example password:

`admin123`

Generated hashes:

- MD5: `0192023a7bbd73250516f069df18b500`
- SHA-1: `f865b53623b121fd34ee5426c792e5c33af8c227`
- bcrypt: `$2b$10$...`

This process ensures that the original password is never stored in the database.

4) Cracking Weak Password Hashes Using Wordlists

A **wordlist** is a file containing common and leaked passwords collected from previous data breaches.

Example wordlist entries:

```
123456
password
admin
qwerty
```

How wordlist cracking works

1. The attacker takes a word from the wordlist
2. Hashes it using the same algorithm
3. Compares it with the stored hash
4. If both hashes match, the password is cracked

Example:

```
Hash: 5f4dcc3b5aa765d61d8327deb882cf99
Word tried: password
Result: Match found
```

This shows why weak passwords are very dangerous.

5) Brute Force Attack vs Dictionary Attack

Dictionary Attack

- Uses a predefined list of common passwords
- Very fast for weak passwords

Example:

```
password
admin
welcome
```

Brute Force Attack

- Tries every possible combination of characters
- Slower but guaranteed to work eventually

Example:

```
a → aa → aaa → aaaa → ...
```

Comparison

Attack Type	Speed	Effectiveness
Dictionary	Fast	Works on weak passwords
Brute Force	Very slow	Works on any passw

6) Why Weak Passwords Fail

From my analysis, weak passwords fail because:

- They are short
- They use common words
- They follow predictable patterns
- They are reused across multiple websites

Examples of weak passwords:

```
password
admin123
123456
```

These passwords exist in wordlists and can be cracked within seconds.

7) Multi-Factor Authentication (MFA) and Its Importance

Multi-Factor Authentication (MFA) requires more than one form of verification to log in.

Example: Bank Login

1. Enter username and password
2. Receive OTP on mobile
3. Enter OTP to complete login

Even if the password is stolen, the attacker cannot log in without the second factor.

Types of Authentication Factors

- Something you know – Password
- Something you have – OTP, mobile phone
- Something you are – Fingerprint or face recognition

MFA greatly reduces the risk of unauthorized access.

8) Recommendations for Strong Authentication

Based on my learning, the following practices are recommended:

Strong Password Practices

- Use at least 12–16 characters
- Combine uppercase, lowercase, numbers, and symbols
- Avoid dictionary words

Example of strong password:

`T!9xQ@4LpZ#8`

Additional Security Measures

- Use bcrypt or Argon2 for hashing
- Enable MFA
- Limit login attempts
- Use password managers
- Avoid password reuse

Conclusion

Through this task, I learned how passwords are securely stored using hashing, how weak password hashes can be cracked using dictionary and brute force attacks, and why weak passwords are a major security risk. I also understood the importance of strong authentication mechanisms like MFA and secure hashing algorithms. Implementing strong password policies and multi-factor authentication is essential to protect modern systems from attacks.