Task 4: Password Security & Authentication Analysis

➢ How Password is stored ?

- Passwords are **not stored in plain text** in a database because it is unsafe. If hackers get access to the database, they can see all passwords.
- To avoid this, websites use a process called **hashing**.
- Hashing converts a password into a fixed-length coded value using a mathematical function. This hashed value **cannot be converted back** into the original password.
- Before hashing, a **salt** (random value) is added to the password. This makes the hash stronger and prevents common attacks like rainbow table attacks.
- Only the **hashed password and salt** are stored in the database, not the actual password.
- When a user logs in, the entered password is hashed again using the same salt. If the new hash matches the stored hash, the user is authenticated.
- Thus, even if the database is leaked, the original passwords remain safe.

➢ Different Hash Types:

Hashing algorithms are used to convert data like passwords or files into a fixed-length value called a hash. Different hash types are used based on security needs.

1. MD5 (Message Digest 5)

MD5 produces a 128-bit hash value.
It is **fast but insecure** and can be easily broken using collisions.
It is **not recommended** for password storage.

2. SHA-1 (Secure Hash Algorithm 1)

SHA-1 generates a 160-bit hash.
It is more secure than MD5 but now considered **weak** .
Modern systems avoid using SHA-1 for security purposes.

3. SHA-2 (SHA-256, SHA-512)

SHA-2 is a family of secure hashing algorithms.
SHA-256 and SHA-512 are commonly used.
They provide **strong security** and are widely used in applications and digital certificates.

➢ Bruteforce attack vs dictionary attack :

- A brute force attack is a method where an attacker tries **all possible combinations** of characters to guess a password. It does not use any logic or word list and keeps trying until the correct password is found, which makes it very slow but guaranteed to work if enough time is given. A dictionary attack, on the other hand, uses a **predefined list of common words and passwords** to guess the correct password. It is faster than brute force because it tries only likely passwords, but it fails if the password is complex and not present in the dictionary. Both attacks are used to crack passwords, but brute force relies on time and computing power, while dictionary attacks rely on common human password habits.

- Why Weak Passwords Fail?
  - Weak passwords fail because they are **easy to guess or crack** using common attack methods like brute force and dictionary attacks. Passwords that are short, use common words, names, numbers, or patterns can be broken very quickly by attackers. Hackers use automated tools that can try millions of passwords in seconds, so simple passwords do not provide enough security. Weak passwords also fail because many users reuse them on multiple websites, so if one site is hacked, other accounts can be compromised as well. Therefore, weak passwords do not provide proper protection against unauthorized access.
- What is MFA?
  - MFA stands for **Multi-Factor Authentication**. It is a security method where a user must provide **more than one type of verification** to log in to an account. Instead of using only a password, MFA requires an additional factor such as a one-time password (OTP), fingerprint, face recognition, or a mobile app approval. Even if an attacker steals the password, they cannot access the account without the second factor. MFA increases security by adding extra layers of protection and is commonly used in banking, email, and cloud services.
- Recommendations for Strong Authentication:
  - Strong authentication can be achieved by using **long and complex passwords** that include letters, numbers, and symbols, instead of common words or patterns. Users should enable **Multi-Factor Authentication (MFA)** so that even if a password is stolen, an extra verification step is required. Passwords should be **unique for every website** and should not be reused across multiple accounts. Systems should use **secure password hashing with salt** to store passwords safely. Regular password updates, account lockout after multiple failed attempts, and avoiding public or shared devices also help in improving authentication security.