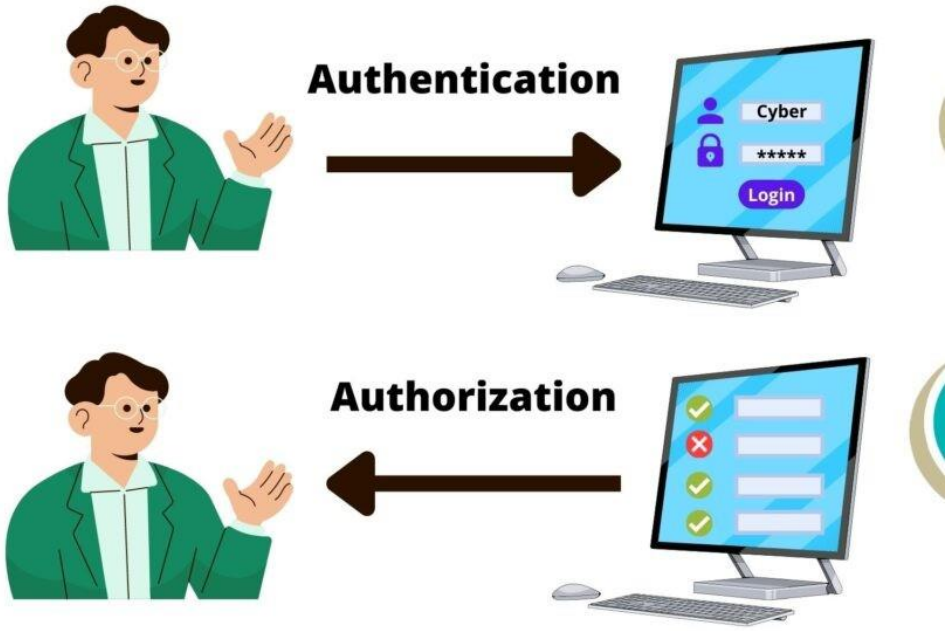
Securing Account

Authentication

- **Authentication** is the process of verifying a user's identity before granting access to a system or service.
- **Common Authentication Factors:**
  - **Something You Know** – Passwords, PINs
  - **Something You Have** – Smart card, OTP (one-time password)
  - **Something You Are** – Fingerprint, face recognition
- **Single vs. Multi-Factor Authentication (MFA):**
  - **Single-Factor:** Uses just one method (e.g., password).
  - **Multi-Factor:** Combines two or more methods for stronger security.
- **Why is Authentication Important?**
  It prevents unauthorized access, protects sensitive data, and enhances overall cybersecurity.

Authorization



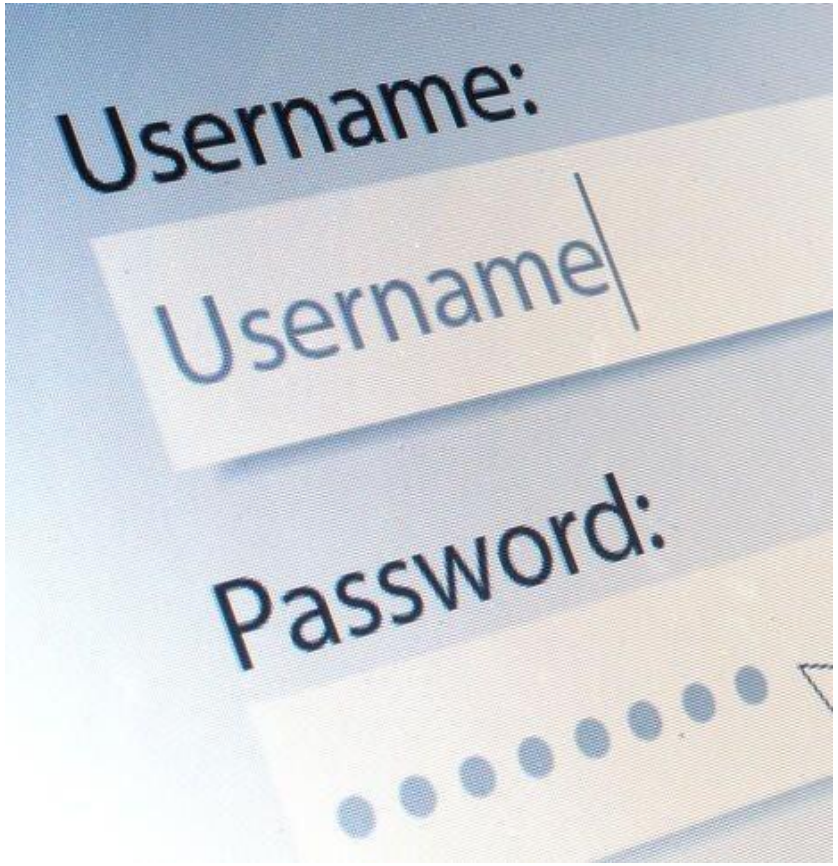
- **Authorization** is the process of determining what actions or resources a user is allowed to access **after authentication**.
- **Key Difference from Authentication:**
  - **Authentication** = "Who are you?" (Verifying identity)
  - **Authorization** = "What can you do?" (Granting permissions)
- **Examples of Authorization:**
  - A student can view their grades, but only a professor can edit them.
  - A bank customer can check their balance, but only an employee can approve loans.
- **Types of Authorization Controls:**
  - **Role-Based Access Control (RBAC):** Permissions based on roles (e.g., Admin, User).
  - **Access Control Lists (ACLs):** Specific rules for users or groups.

Username and Password

- A **username** identifies a user (mostly public), while a **password** verifies their identity (must be private).
- Think of a username as a **"login ID"** and a password as a **"secret key."**

Why is a **Strong Password** Important?

- Weak passwords make accounts vulnerable to hacking.
- A strong password protects personal and sensitive information.
- **Common Attacks on Passwords:**
  ⚠ **Brute Force Attack:** Hackers try many passwords until they get the right one.
  ⚠ **Phishing Attack:** Tricking users into revealing their passwords.



Strong Password

What Makes a Password Strong?

- **Length Matters!** Longer passwords are harder to crack using brute force attacks.
- **Mix of Characters Increases Security**
  - Use a combination of:
    Uppercase letters (A-Z)
    Lowercase letters (a-z)
    Numbers (0-9)
    Special characters (!, @, #, etc.)
- **Weak vs. Strong Passwords**
  - ❌ "123456" (Weak – easy to guess)
  - ❌ "password" (Weak – common word)
  - ✅ "G!v3M3@L0ng&SafeP@ss" (Strong – long & mixed)
- **How Long Does It Take to Crack a Password?**
  - 🔴 **6 characters (only letters)** → **Seconds** to crack
  - 🟡 **8 characters (with numbers)** → **Minutes to hours**
  - 🟢 **12+ characters (with mix)** → **Years to crack**

Dictionary Attack

- A **dictionary attack** is a method used by hackers where they try common words and phrases from a predefined list (dictionary) to guess passwords.
  - Attackers use common words, names, and even popular passwords to quickly break into accounts.
- **How Does a Dictionary Attack Work?**
  - The attacker starts with a list of commonly used passwords (e.g., "password", "123456", "let me in").
  - The attacker tries each word in the list until the correct password is found.
- **The more common the password, the easier it is to crack.**
- **Common Passwords to Avoid:**
  - **Simple Words:** "password", "qwerty", "abc123"
  - **Personal Information:** Names, birthdates, or favorite sports teams
  - **Repetitive Patterns:** "111111", "password1"
  - **Easy-to-Guess Sequences:** "1234", "password123", "letmein"

Brute Force Attack

Definition: A brute force attack is a trial-and-error method where attackers systematically try every possible password or encryption key until they find the correct one.

Types of Brute Force Attacks: These include simple brute force (trying all possible passwords), dictionary attacks (using common words and phrases), and credential stuffing (using leaked credentials from data breaches).

Impact and Risks: Brute force attacks can lead to unauthorized access, data breaches, identity theft, and financial losses for individuals and organizations.

Prevention Measures: Using strong, complex passwords, multi-factor authentication (MFA), account lockout policies, and CAPTCHA can help prevent brute force attacks.

Use of Automation: Attackers often use automated tools like Hydra, John the Ripper, and Hashcat to speed up brute force attacks, making them more effective against weak security measures.

Strong Password: Digits (4 char length)

```
#include <iostream>

int main() {
   for (char a = '0'; a <= '9'; ++a) {
      for (char b = '0'; b <= '9'; ++b) {
         for (char c = '0'; c <= '9'; ++c) {
            for (char d = '0'; d <= '9'; ++d) {
               std::cout << a << b << c << d << std::endl;
            }
         }
      }
   }
   return 0;
}
```

Possible passwords

10*10*10*10= **10000**

https://www.online-cpp.com/

Strong Password: Lowercase (4 char length)

```
#include <iostream>

int main() {
   for (char a = 'a'; a <= 'z'; ++a) {
      for (char b = 'a'; b <= 'z'; ++b) {
         for (char c = 'a'; c <= 'z'; ++c) {
            for (char d = 'a'; d <= 'z'; ++d) {
               std::cout << a << b << c << d << std::endl;
            }
         }
      }
   }
   return 0;
}
```

Possible passwords

26*26*26*26= **456,976**

Strong Password: Lowercase +uppercase (4 char length)

```
#include <iostream>

int main() {
   for (char a = 'A'; a <= 'z'; ++a) {
      if (a > 'Z' && a < 'a') continue; // Skip non-letter characters
      for (char b = 'A'; b <= 'z'; ++b) {
         if (b > 'Z' && b < 'a') continue;
         for (char c = 'A'; c <= 'z'; ++c) {
            if (c > 'Z' && c < 'a') continue;
            for (char d = 'A'; d <= 'z'; ++d) {
               if (d > 'Z' && d < 'a') continue;
               std::cout << a << b << c << d << std::endl;
            }
         }
      }
   }
   return 0;
}
```

Possible passwords

$52*52*52*52$**=7,311,616**

Strong Password: Lowercase +Uppercase+ punctuations and symbole (4 char length)

```
#include <iostream>
int main() {
   std::string chars =
      "abcdefghijklmnopqrstuvwxyz"
      "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
      "0123456789"
      "!\"#$%&'()*+,-./:;<=>?@[]^_`{|}~";  // All valid characters

   int n = chars.length(); // Total number of characters
   for (int i = 0; i < n; ++i) {
      for (int j = 0; j < n; ++j) {
         for (int k = 0; k < n; ++k) {
            for (int l = 0; l < n; ++l) {
               std::cout << chars[i] << chars[j] << chars[k] << chars[l] << std::endl;
            }
         }
      }
   }
   return 0;
}
```

Possible passwords

$94*94*94*94$**=78,074,896**

Strong Password: Lowercase +Uppercase+ punctuations and symbole (8 char length)

CODE?

Possible passwords

94*94*94*94*94*94*94*94=

6,095,689,385,410,816

> Six quadrillion

Usability vs Security

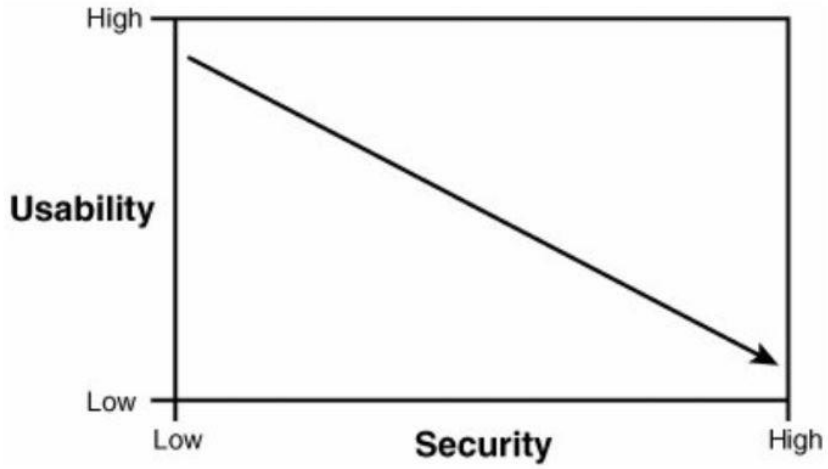
Security vs. Memorability: Highly complex passwords (e.g., long, random character strings) enhance security but reduce usability as they are harder to remember. Simpler passwords are easier to recall but more vulnerable to attacks.

User Behavior: Complex password requirements often lead users to write them down, reuse them across sites, or use predictable patterns, which can compromise security. Usable passwords balance security with ease of use.

Authentication Alternatives: To improve both usability and security, alternatives like password managers, multi-factor authentication (MFA), or biometric authentication (e.g., fingerprints, facial recognition) can be implemented.

Attack Resistance: More complex passwords are resistant to brute-force and dictionary attacks, while simpler passwords are more susceptible to hacking attempts.

System Design Considerations: Organizations must balance usability and security by implementing policies such as passphrases (long but easy-to-remember phrases) instead of overly complex passwords.

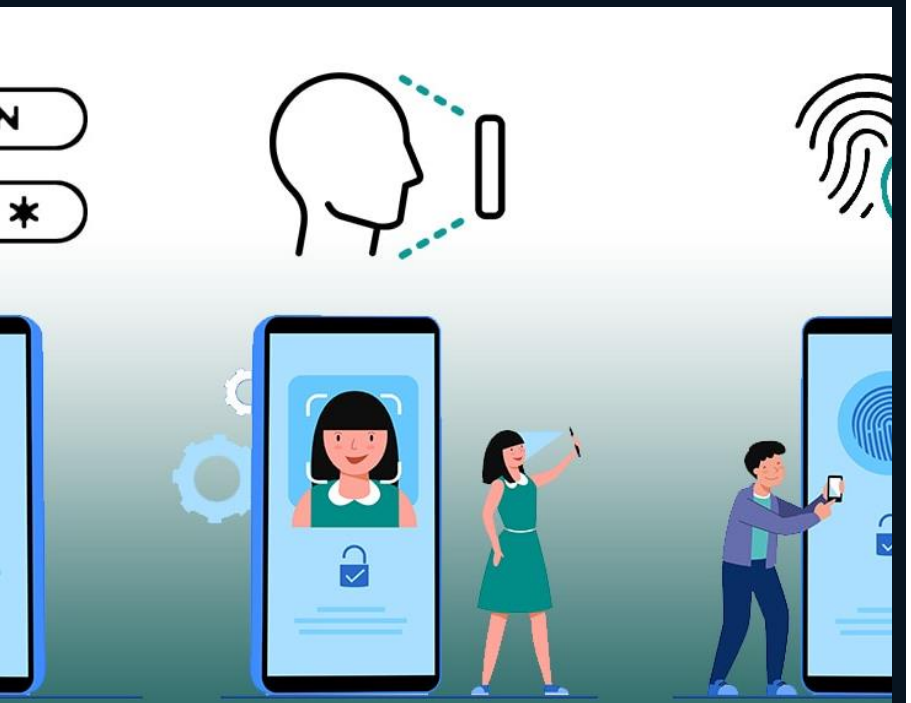


National Institute of Standard Technology (NIST)

1. Verifiers **SHALL** require passwords to be a minimum of **eight** characters in length.
2. Verifiers **SHOULD** permit a maximum password length of at least 64 characters.
3. Verifiers **SHOULD** accept all printing ASCII characters and the space character in passwords.
4. Verifiers **SHOULD** accept Unicode characters in passwords. Each Unicode code point **SHALL** be counted as a single character when evaluating password length.
5. Verifiers **SHALL NOT** impose other composition rules (e.g., requiring mixtures of different character types) for passwords.
6. Verifiers **SHALL NOT** require users to change passwords periodically. However, verifiers **SHALL** force a change if there is evidence of compromise of the authenticator.
7. Verifiers **SHALL NOT** permit the subscriber to store a hint that is accessible to an unauthenticated claimant.
8. Verifiers **SHALL NOT** prompt subscribers to use knowledge-based authentication (KBA) (e.g., “What was the name of your first pet?”) or security questions when choosing passwords.
9. Verifiers **SHALL** implement a rate-limiting mechanism that effectively limits the number of failed authentication attempts that can be made on the subscriber account

https://pages.nist.gov/800-63-4/sp800-63b.html

Multi-Factor Authentication



Definition:
MFA is a security system that requires multiple forms of verification to confirm a user's identity, making unauthorized access more difficult.

Why is MFA Important?

- Enhances security beyond just passwords
- Protects against phishing, brute force attacks, and credential theft
- Reduces the risk of unauthorized access

How Does MFA Work?
Users must provide two or more independent authentication factors from different categories (Knowledge, Possession, Inherence).

Main MFA Factor Types

Knowledge (Something You Know)

- Passwords, PINs, security questions
- Common but vulnerable to phishing and brute-force attacks

Possession (Something You Have)

- OTP (One-Time Password) via SMS, email, or authentication apps
- Hardware tokens, smart cards, or security keys

Inherence (Something You Are)

- Biometrics: fingerprint, facial recognition, retina scan
- Voice recognition or behavioural biometrics.

OTP (One-Time Password) advantages

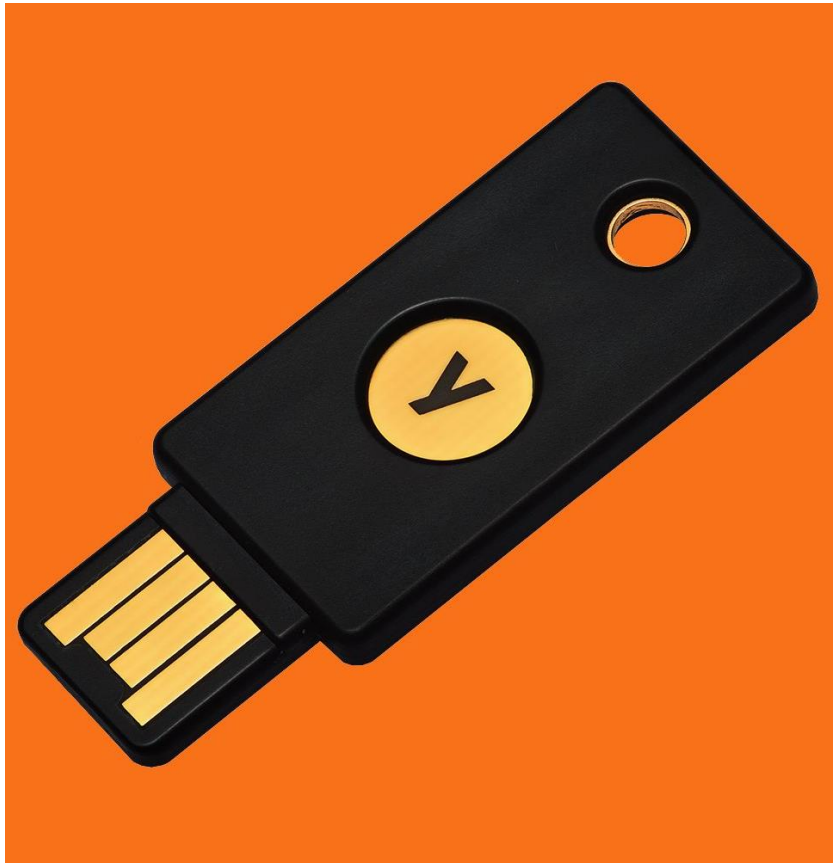
- **Enhanced Security:** OTPs provide an extra layer of protection beyond static passwords, reducing the risk of unauthorized access.
- **Limited Validity:** OTPs expire quickly, making them less useful to attackers even if intercepted.
- **User-Friendly:** Easy to use and implement without requiring additional hardware.
- **No Need to Remember:** Users don’t have to memorize OTPs, unlike passwords.
- **Wide Accessibility:** SMS and email OTPs work on most devices, and authentication apps provide offline access.

OTP (One-Time Password) disadvantages

- **Susceptible to Phishing & Social Engineering:** Attackers can trick users into revealing OTPs.
- **SMS-Based Vulnerabilities:** SMS OTPs can be intercepted via **SIM swapping**, malware, or network attacks.
- **Delivery Delays & Failures:** Network issues can cause OTPs sent via SMS or email to be delayed or undelivered.
- **Dependency on the Internet or Mobile Network:** Email OTPs require an Internet connection, and SMS OTPs need network coverage.
- **Authentication Apps Require Setup:** While more secure, authentication apps (e.g., Google Authenticator) require installation and initial setup, which may be inconvenient for some users.

Advantages of (Hardware tokens, smart cards, or security keys)

- **High Security:** Unlike OTPs sent via SMS or email, hardware-based authentication is resistant to phishing, SIM swapping, and network attacks.
- **No Internet or Mobile Network Required:** Security keys and smart cards work offline, reducing reliance on connectivity.
- **Difficult to Clone or Hack Remotely:** Physical possession is required, making remote attacks significantly harder.
- **Fast Authentication:** Security keys (e.g., YubiKey) provide quick and seamless authentication with just a tap.
- **Enterprise-Grade Protection:** Widely used in organizations to secure access to sensitive systems.



Disadvantages of (Hardware tokens, smart cards, or security keys)

- **Risk of Loss or Theft:** If a user loses the hardware token or security key, access recovery can be challenging.
- **Higher Cost:** Purchasing hardware tokens or security keys is more expensive than using free authentication apps or SMS-based OTPs.
- **Device Compatibility Issues:** Some older systems or devices may not support modern security keys (e.g., USB-C or NFC-based keys).
- **Setup Complexity:** Initial setup and integration into authentication systems require time and IT support.
- **Limited Backup Options:** If lost, users may need backup methods like recovery codes, adding complexity to account recovery.

Advantages of (Biometrics as a MFA)

- **High Security & Uniqueness:** Biometric traits (fingerprints, facial recognition, retina scan) are unique to each individual, making them difficult to forge or steal.
- **Convenience:** No need to remember passwords or carry physical devices—authentication is quick and seamless.
- **Fast Authentication:** Scanning a fingerprint or face is much faster than entering a password or OTP.
- **Resistance to Phishing & Credential Theft:** Biometrics cannot be easily shared or intercepted like passwords or OTPs.

Disadvantages of (Biometrics as a MFA)

- **Privacy Concerns:** Biometric data is sensitive, and if compromised, it cannot be changed like a password.
- **Hardware Dependency:** Requires compatible devices (fingerprint scanners, facial recognition cameras, etc.), which may not be universally available.
- **False Positives & Negatives:** Environmental factors (e.g., poor lighting for facial recognition, and injuries affecting fingerprints) can lead to authentication failures.
- **Potential for Spoofing:** Advanced spoofing techniques (e.g., deepfake attacks on facial recognition) can sometimes bypass biometric security.
- **Data Storage Risks:** If biometric data is stored improperly or hacked, it poses a significant security risk.

Credential Stuffing

Definition: A credential stuffing attack occurs when attackers use stolen username-password pairs from data breaches to gain unauthorized access to multiple accounts, exploiting users who reuse credentials across different sites.

Automated Execution: Attackers use **bots** and scripts to **test large volumes of credentials** on **multiple websites quickly**, increasing the success rate.

Common Targets: Online banking, email services, e-commerce platforms, and social media accounts are frequent targets since users often reuse passwords for convenience.

Impact & Risks: Successful attacks can lead to data breaches, financial fraud, identity theft, and account takeovers, affecting both individuals and organizations.

Prevention Measures: Using unique passwords for each account, enabling multi-factor authentication (MFA), monitoring login attempts, and implementing CAPTCHA or rate limiting can help prevent credential stuffing attacks.

Social Engineering

Definition: Social engineering is a psychological manipulation technique used by attackers to trick individuals into revealing confidential information, such as passwords, financial details, or security codes.

Common Techniques: Includes phishing (fraudulent emails or messages), pretexting (impersonating someone with a fabricated scenario), baiting (offering something tempting, like free software), and tailgating (physically following someone into a restricted area).

Exploits Human Trust: Unlike traditional cyberattacks that exploit software vulnerabilities, social engineering targets human emotions, such as fear, urgency, curiosity, or authority.

Impact & Risks: Can lead to identity theft, financial loss, unauthorized system access, and data breaches in both personal and corporate environments.

- **Prevention Measures**: Security awareness training, verifying requests for sensitive information, using MFA, being cautious of unsolicited communications, and implementing email filters can help mitigate social engineering attacks.

Phishing

Definition: Phishing is a cyberattack where attackers impersonate legitimate entities (e.g., banks, and social media platforms) to trick users into revealing sensitive information, such as login credentials, credit card details, or personal data.

Common Types:

- **Email Phishing** – Fraudulent emails with malicious links or attachments.
- **Spear Phishing** – Targeted phishing attacks aimed at specific individuals or organizations.
- **Smishing (SMS Phishing)** – Fake text messages tricking users into clicking malicious links.
- **Vishing (Voice Phishing)** – Phone-based scams impersonating trusted institutions.

Exploits Human Psychology: Phishing attacks create a sense of urgency, fear, or curiosity, tricking victims into acting quickly without verifying the legitimacy of the message.

Impact & Risks: Successful phishing attacks can lead to financial fraud, identity theft, malware infections, and large-scale data breaches.

- **Prevention Measures**: Verify sender emails, avoid clicking suspicious links, enable multi-factor authentication (MFA), use email filters, and educate users on recognizing phishing attempts.

Man in the Middle

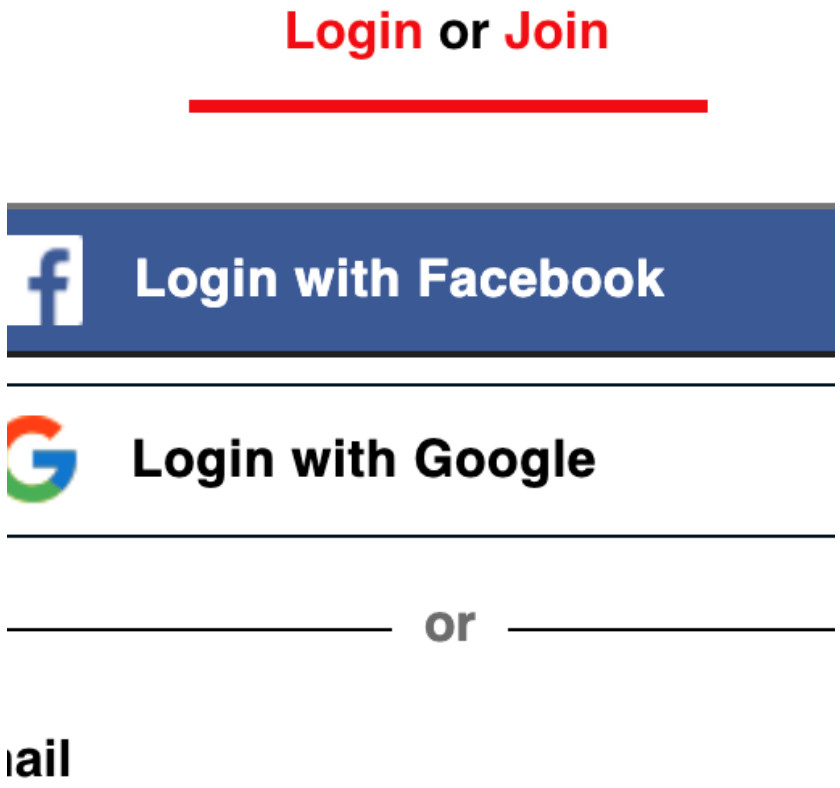
- **Definition**: the attack occurs when an attacker intercepts and manipulates communication between two parties (e.g., a user and a website) to steal sensitive information like passwords.
- **How It Works**: The attacker places themselves between the victim and the intended recipient, capturing login credentials, banking details, or personal data.
- **Common Techniques**:
  - **Wi-Fi Eavesdropping** – Intercepting unencrypted data over public or unsecured Wi-Fi networks.
  - **Session Hijacking** – Stealing session cookies to gain unauthorized access to accounts.
  - **DNS Spoofing** – Redirecting victims to fraudulent websites that mimic legitimate ones.
  - **HTTPS Stripping** – Downgrading a secure HTTPS connection to an insecure HTTP one to capture credentials.

Single Sign On

Single Sign-On (SSO) is an authentication mechanism that allows users to log in once and gain access to multiple applications or systems without needing to re-enter credentials. It streamlines access management across different platforms.

🔷 **How SSO Works**

1. A user logs in once to an **identity provider (IdP)**.
2. The IdP authenticates the user and issues a **session/token**.
3. The token is shared with authorized applications, granting seamless access.
4. The user can now access multiple services without additional logins.



Single Sign On

Benefits of SSO

- **Convenience** – Users only need to remember one set of credentials.
- **Improved Security** – Reduces password fatigue and reuse across multiple accounts.
- **Enhanced Productivity** – Faster access to multiple applications without repeated logins.
- **Simplified User Management** – IT admins can manage access centrally.

Risks & Challenges of SSO

- **Single Point of Failure** – If the SSO system is compromised, attackers can access all linked accounts.
- **Dependency on the IdP** – If the identity provider goes down, users may lose access to all services.
- **Implementation Complexity** – Requires integration with multiple systems, which can be challenging for enterprises.

Password Manager

What is a Password Manager?

A **password manager** is a software tool that securely stores, generates, and auto-fills passwords for users. It helps manage complex passwords across multiple accounts without the need to remember them.

Benefits of Using a Password Manager

- **Strong & Unique Passwords** – Generates complex passwords for each account, reducing the risk of credential stuffing attacks.
- **Convenience** – Autofills login credentials, saving time and effort.
- **Encrypted Storage** – Stores passwords securely using strong encryption (e.g., AES-256).
- **Cross-Device Synchronization** – Allows access to passwords across multiple devices (desktop, mobile, browser extensions).
- **Secure Sharing** – Some password managers allow the safe sharing of credentials with trusted individuals without revealing the actual password.
- It can prevent phishing links.

Password Manager

Risks & Challenges

- **Single Point of Failure –** If the master password is compromised, all stored passwords could be at risk.
- **Data Breach Risks –** If the password manager service is hacked, user credentials may be exposed (though encryption minimizes risk).
- **Device & Browser Compatibility –** Some password managers may not work on all devices or browsers.

Best Practices for Using a Password Manager

- Use a **strong master password** and enable **multi-factor authentication (MFA)**.
- Keep the software **updated** to patch security vulnerabilities.
- Enable **backup and recovery options** in case of account lockout.
- Use **a reputable password manager** with strong encryption and zero-knowledge architecture.

Questions