Week 5 revision Notes

IFB240 Week 5 - User Authentication

Overview

This week's focus is on **identity management** and **user authentication** mechanisms. We explore why user authentication is essential for securing systems, different authentication methods (knowledge-based, object-based, biometric, and location-based), and the strengths and weaknesses of each type.

1. Why is Identity Management Important?

Key Concepts

- **Identity Management** is crucial because it ensures that entities (users or systems) interacting with sensitive resources are legitimate.
- **Optus Data Breach (Sept 2022)**: Highlighted the importance of identity management. Approximately **10 million customers' data** was compromised, underlining the risks when systems fail to verify identity properly.

Reasons for Identity Management

1. **Access Control**: Identity management is used to decide whether a user can access a specific resource.
2. **Accountability**: To hold users accountable for actions, systems must ensure that only authorized users perform certain actions. For instance, logging into an account to perform critical updates must be traceable to the right individual.

Scenarios

- **Logical Scenarios**: Systems verify identity electronically, such as logging into your email.
- **Physical Scenarios**: Access control to a restricted building, using an ID card or biometric scanner.

2. What is User Authentication?

Definition

- **User Authentication** is the process of verifying whether the identity a user claims is authentic. It typically requires a **unique identifier** (username) and an **authenticator** (password, token, etc.).

Steps in User Authentication

1. **Identification**: User claims an identity by providing a **User ID**.
2. **Authentication**: User provides a credential to validate the claim (e.g., a password or a token).
3. **Authorization**: System checks if the user is allowed to perform the requested action.

- **Example**: Logging into a QUT system involves entering a username and password, and the system then verifies if the credentials are valid.

3. Types of User Authentication Mechanisms

Knowledge-Based Authenticators

- **What is it?**: Authentication based on something the user knows, such as a **password** or **PIN**.
- **Common Examples**:
  - **Passwords**: Most common form of authentication, but can be weak if not managed properly.
  - **Security Questions**: e.g., "What is your mother's maiden name?"

Security Issues with Passwords

1. **Weak Passwords**: Common passwords like **123456** or **password** are easy to guess.
2. **Reusability**: Users often reuse passwords across multiple accounts, making it easier for attackers to compromise multiple systems.
3. **Storage Requirements**: Passwords must be stored securely to prevent unauthorized access—ideally, **hashed and salted** to protect them even if the password database is compromised.

Password Management Recommendations

- Use a **password manager** to generate and store strong, unique passwords for each service.
- Avoid storing passwords in plaintext—always hash and add salt to prevent easy reverse-engineering of passwords.

Object-Based Authenticators (Something You Have)

- **Examples**:
  - **Tokens**: Physical devices, such as **swipe cards** or **RSA SecurID tokens**.
  - **One-Time Password Generators (OTP)**: Devices or mobile apps generating dynamic codes.

Advantages and Disadvantages

- **Advantages**: Difficult to share and provides an additional layer of security if combined with a password (multi-factor authentication).

- **Disadvantages**: Can be lost, stolen, or damaged, which might lead to access issues or require replacement.

Real-World Example

- **RSA SecurID Tokens**: Generate a unique code every 30-60 seconds. Tokens must be synchronized with the system to ensure that the code provided matches.

ID-Based Authenticators (Something You Are)

- **Biometrics**: Characteristics unique to the user, such as **fingerprints**, **retina scans**, or **facial recognition**.

Key Advantages

- **Uniqueness**: Biometrics cannot be forgotten or easily stolen.
- **User Presence**: Ensures the user is physically present, making remote attacks more challenging.

Key Challenges

- **Difficulty in Replacement**: Unlike passwords, biometric data cannot be easily changed if compromised.
- **Error Rates**: Systems can make errors, such as **False Match Rate (FMR)** (granting access to an unauthorized user) or **False Non-Match Rate (FNMR)** (denying access to an authorized user).

Location-Based Authenticators (Somewhere You Are)

- **Definition**: Authenticates based on the user's location, using GPS or IP address data.
- **Examples**:
  - **Geo-Fencing**: Restricting system access to certain geographical areas.

Privacy Considerations

- **Tracking**: Location-based authentication can raise privacy issues, as it involves tracking the user's movements.

Multi-Factor Authentication (MFA)

- **Definition**: Combines two or more authentication methods from different categories, such as **password + token** or **password + fingerprint**.
- **Why Use MFA?**
  - **Increased Security**: Even if one factor is compromised, the attacker still needs access to the second factor.
  - **Common Usage**: Used by many online services such as **Google** or **Facebook**.

4. Hash Functions for Secure Password Storage

What is a Hash Function?

- **Hash Function**: A cryptographic function that takes an input (e.g., a password) and produces a fixed-length, unique output known as a **hash value**.
- **Properties**:
  - **One-Way**: It should be easy to generate a hash from input data but computationally infeasible to retrieve the original input from the hash.
  - **Collision-Resistance**: It should be hard to find two different inputs that produce the same hash value.

Using Hashes for Password Storage

- Passwords should never be stored in plaintext. Instead, they should be **hashed** and **salted** to ensure security.
- **Salting**: Adds random data to passwords before hashing them to ensure even identical passwords have different hash values.

Example of Issues with Poor Hash Practices

- In **2011**, the **RSA SecureID tokens** suffered a security breach, where attackers allegedly gained access to the seed values, compromising token generation. This breach illustrated the need for robust encryption and hash functions to secure sensitive data.

5. Summary of Authentication Methods

Comparison of Authentication Types

Type	Description	Advantages	Disadvantages
Knowledge-Based	Passwords, PINs	Low-cost, easy to implement	Easy to forget, susceptible to guessing and breaches
Object-Based	Tokens, swipe cards	Hard to share, easy to detect loss	Can be lost, replacement costs
ID-Based (Biometrics)	Fingerprints, retina scans	High security, non-replicable	Expensive, cannot easily be changed
Location-Based	GPS, IP address	Useful for restricting access	Privacy concerns, can be spoofed
Multi-Factor (MFA)	Combines multiple methods	Very secure	More complex for users

Key Takeaways

- **Identity Management** is the cornerstone of security, ensuring that resources are accessed only by those who are supposed to.
- **Authentication Mechanisms** vary in strength and applicability. For maximum security, **multi-factor authentication** is recommended.
- **Secure Password Practices** include using strong, unique passwords for each account, stored as salted hashes.

Real-World Lessons

- **Optus Breach** and **RSA Token Breach** demonstrate the consequences of inadequate identity management and weak authentication security.
- Ensuring **secure password storage** and adopting MFA are crucial steps to mitigating security risks.

These notes should help solidify your understanding of user authentication mechanisms, the principles of secure identity management, and practical applications of each concept in cybersecurity.