OWASP Top 10:

Broken Authentication

01.12.2024

—

Nikhil Kumar

Introduction

Broken Authentication is a critical security risk listed in the OWASP Top 10. It occurs when an application's authentication mechanisms are improperly implemented, allowing attackers to compromise passwords, session tokens, or exploit other flaws to assume the identities of legitimate users.

This document provides a detailed yet simple explanation of the concept, how it can be exploited, and best practices to mitigate the risk.

What is Authentication?

Authentication is the process of verifying the identity of a user, application, or system. It typically involves:

1. **Something the user knows**: Passwords or PINs.
2. **Something the user has**: OTPs, tokens, or devices.
3. **Something the user is**: Biometrics like fingerprints or facial recognition.

When this process is poorly designed or implemented, it can lead to **broken authentication vulnerabilities.**

How Does Broken Authentication Occur?

Broken authentication often arises due to:

1. Weak Password Policies

- Allowing users to set simple passwords (e.g., "123456" or "password").
- Lack of password complexity requirements.
- Absence of account lockout mechanisms after multiple failed login attempts (brute-force attacks).

2. Credential Stuffing

- Reusing usernames and passwords across multiple sites.
- Attackers use stolen credentials from data breaches to log in to other systems.

3. Session Management Issues

- Session IDs are not rotated after login, making it easier for attackers to hijack active sessions.
- Session IDs are exposed in URLs or not securely stored.
- Failure to implement session timeout or logout mechanisms.

4. Insecure Implementation of Multi-Factor Authentication (MFA)

- Poorly configured MFA systems, such as using predictable or easily bypassed tokens.

5. Exposed Endpoints

- APIs or endpoints without proper authentication checks, allowing unauthorized access.

Examples of Broken Authentication Exploits

1. Credential Stuffing Attack

- **Scenario**: An attacker uses a list of stolen credentials from a breached site and tests them on another website.
- **Impact**: Unauthorized access to user accounts.

2. Session Hijacking

- **Scenario**: A session ID is exposed in the URL or intercepted over an insecure channel (e.g., HTTP instead of HTTPS).
- **Impact**: The attacker uses the session ID to impersonate the user.

3. Brute-Force Attacks

- **Scenario**: An application allows unlimited login attempts without account lockout or CAPTCHA.
- **Impact**: The attacker systematically guesses passwords until they gain access.

4. Insecure Password Recovery

- **Scenario**: Password recovery flows that rely on weak security questions or unverified email links.
- **Impact**: The attacker exploits this to reset passwords and gain access.

Impacts of Broken Authentication

1. **Unauthorized Access**: Attackers can access sensitive user data or resources.
2. **Account Takeover**: Compromised accounts may lead to further exploitation.
3. **Reputational Damage**: Organizations lose user trust after such incidents.
4. **Financial Loss**: Stolen credentials can lead to fraudulent transactions or theft.

How to Prevent Broken Authentication

1. Enforce Strong Password Policies

- Require passwords to be long and complex (e.g., at least 12 characters with a mix of letters, numbers, and symbols).
- Implement password expiration and rotation policies.
- Prevent the use of commonly used passwords through a blacklist.

2. Implement Multi-Factor Authentication (MFA)

- Require users to verify their identity through additional factors like OTPs, biometric authentication, or hardware tokens.

3. Secure Session Management

- Rotate session IDs after login.
- Store session IDs securely (e.g., in cookies with the `HttpOnly` and `Secure` flags).
- Enforce session timeouts and automatic logout after inactivity.
- Use strong encryption for session tokens.

4. Limit Login Attempts

- Implement account lockout mechanisms after a set number of failed login attempts.
- Use CAPTCHAs to prevent automated brute-force attacks.

5. Protect Password Recovery Processes

- Avoid relying on security questions; instead, use verified email or phone-based recovery.
- Ensure password reset tokens are time-limited and single-use.

6. Monitor and Log Authentication Events

- Track failed login attempts, unusual login patterns, and access from suspicious IP addresses.
- Implement real-time alerts for account compromise attempts.

7. Use Secure Communication Channels

- Enforce HTTPS for all communications.
- Avoid exposing sensitive information in URLs (e.g., session tokens).

OWASP Tools to Address Broken Authentication

Broken Authentication vulnerabilities can be detected using tools like:

- **Burp Suite**: For testing authentication flows, session management, and identifying vulnerabilities in web applications.
- **Nessus**: For scanning systems and applications to detect misconfigurations or weak authentication practices.
- **Nmap**: For identifying open ports, services, and potential authentication flaws in networked systems.

- **ZAP (Zed Attack Proxy)**: For simulating attacks and testing authentication mechanisms in web applications.
- **OWASP Dependency Check**: For identifying vulnerabilities in third-party libraries that might impact authentication.
- **Custom Scripts**: Designed to test specific authentication mechanisms, such as brute-force attempts or session hijacking simulations.

Conclusion

Broken Authentication is a critical risk that can lead to devastating consequences for organizations and users. By understanding its causes and implementing robust authentication mechanisms, you can significantly reduce the attack surface and enhance the overall security posture of your application.

Always prioritize secure authentication practices, monitor for suspicious activity, and educate users about maintaining strong passwords and good security hygiene.