UTT

UNIVERSIDAD TECNOLÓGICA DE TIJUANA

GOBIERNO DE BAJA CALIFORNIA

TOPIC:

Identify points of vulnerability in mobile applications

STUDENT:

Maciel Leyva Ariana Lizeth

GROUP:

10A

SUBJECT:

Comprehensive Mobile Development

TEACHER:

Ray Brunett Parra Galaviz

Tijuana, Baja California, January 23rd of 2025

“Identify points of vulnerability in mobile applications”

What are Mobile Application Vulnerabilities?

Mobile application vulnerabilities refer to weaknesses or flaws in the design, implementation, or configuration of mobile applications that attackers could exploit to compromise the application's security, the device it runs on, or the data it processes. These vulnerabilities can arise due to various factors, including:

- Coding errors
- Insecure data storage
- Insufficient input validation
- Inadequate authentication mechanisms
- Lack of encryption
- Insecure communication channels
- Reliance on outdated or vulnerable third-party libraries

Why is Mobile Application Security Important?

According to Bankmycell statistics, more than 6.64 billion smartphone owners worldwide. Approximately 83% of the global population is connected to the internet and likely uses two or more mobile applications in daily routines. These significant figures highlight the importance and potential of global implementation of strong application security on active mobile apps. The absence of such security measures can jeopardize your company's sensitive data and the valuable digital assets owned by your consumers.

The primary significance of application security lies in safeguarding digital properties such as identities, finances, and sensitive data. Ensuring that your business's mobile application is well-equipped with the necessary security protocols can help prevent security breaches that may endanger you and your consumers.

Common mobile application vulnerabilities

1. Binary Protection

Inadequate Root Detection / Jailbreak Data security and encryption mechanisms on the OS are undone when a device is rooted or jailbroken. When a device is hacked, it may run any malicious code. It can also dramatically modify the application logic's intended behavior. Often, recovery and data forensic tools work on rooted devices adequately. Proper root/jailbreak detection is necessary; this can layer up the data from being exposed.

2. Insufficient Authorization/Authentication

When an application fails to execute adequate authorization checks to verify that the user is executing a function or accessing data in accordance with the security policy, this is known as insufficient authorization. What a user, service, or application is allowed to perform should be monitored by authorization processes. When a user logs in to a website, this does not always imply that the user has complete access to all information and capabilities. A solution can calm your efforts, that is, implementing a tried-and-true authorization system that values policy-based configuration files over the robust authentication/authorization analyses wherever possible.

3. Insecure storage of information

The vulnerability may also occur when the sensitive data isn't securely kept in the device. People must constantly bear in mind that data stored on devices isn't safe since it may be stolen, and sensitive data stored on the device can also be stolen. Apps should save sensitive data in keychain pairs to avoid this issue. If the app saves information in the form of data, then the data needs to be encrypted.

4. Server-Side Vulnerability

Unauthenticated access can be preventable on the server-side, but app design needs to integrate input validation checks and restrictions to decrease the server's workload. When the app processes the server, it is important to verify input data and halt any unusual behavior. You know one can whitelist the necessary forms of data, and the rest can be denied from the app side. Both the app and the server should use encryption while receiving and transmitting data.

5. Secure App Source Code

We all know well that bugs and vulnerabilities in the application code are the initial points of breaking into the application. But attackers are ever ready to reverse your code and don't leave even a single point to come over your logic. They would just a public copy of your app to manipulate things the way it was. In this case, you can create a copy of your original source code and keep it for maintenance purposes.

To secure code, a developer can think of a Code Signing certificate that ensures code integrity and strong security. The certificate assures that the code has not been modified since it is signed. Moreover, it also verifies the publisher's identity. You can find many low-cost SSL providers in the SSL industry that can give surety about application code integrity.

6. Cryptography- Improper Certificate Validation

This app can either validate the SSL/TLS certificates or won't do it; it may not correctly verify the state. What a client can do is drop the connection if the certificate can't be verified. The data can be used for unauthorized access if it is not properly been validated.

You need to ensure that the certificate validation in your app is done properly to cross-check whether a certificate is from a trusted source and it should be from a reputable certificate authority. You should be implementing some recent standard forums for the best validation.

7. Lack of an Expiration Process

A session is the period of time that a user is active on your site or application. If an application's session expiration process is insufficient (or even non-existent), the user's account is exposed to attacks and the violation of their personal data.

It is therefore necessary to find a balance between UX (not asking the user to reconnect to make his experience less pleasant) and the security of the application.

8. Inadequate Cyber Security

When a cybersecurity strategy exists in the organization, the release of an application must, at a minimum, comply with the defined rules. However, it may happen that the strategy in place does not include rules specific to applications, or that it is imprecise or incomplete. In this case, it is necessary to define the security standard.

For example, a mobile backend can be set up. This involves setting up an intermediary space between data exchanges concerning the company and client-side security. The mobile backend will be the only access point for applications to the information system. It will generate alerts to IT teams in case of inappropriate use and will connect in a really secure way, for example via VPN, to the company's information system. The mobile backend will be the only access point for applications to the information system. It will generate alerts to IT teams in case of inappropriate use and will connect in a truly secure way, for example via VPN, to the company's information system.

Other vulnerabilities in mobile applications

- **Bugs in the Operating Systems (OS):** Vulnerabilities in platforms like Android and iOS can enable attackers to exploit flaws in the media playback engine (e.g., Stagefright) or inject malicious code into iOS apps (e.g., XcodeGhost).
- **Weak Authentication:** Poorly implemented authentication mechanisms, such as weak passwords or lack of multi-factor authentication, can lead to unauthorized access to user accounts.

- **Insufficient Data Encryption & Insecure Data Storage:** Weak encryption or insecure storage of sensitive data can result in data breaches, allowing attackers to access and manipulate confidential user information.
- **Inadequate Input Validation:** Failure to properly validate user inputs can lead to security vulnerabilities, such as SQL injection or cross-site scripting (XSS) attacks, which allow attackers to manipulate app behavior.
- **Lack of Secure Session Management:** Weak session handling can leave apps vulnerable to session hijacking or fixation attacks, allowing attackers to gain unauthorized access.
- **Code Obfuscation and Reverse Engineering:** Apps lacking code obfuscation techniques are susceptible to reverse engineering, which enables attackers to analyze the source code and exploit vulnerabilities.
- **Flaws in Hardware or Processors:** Vulnerabilities in hardware components or processors, such as Qualcomm Snapdragon or Samsung Exynos chipsets, can pose security risks to mobile devices and apps.

Types of mobile app security tests

To ensure comprehensive protection of your applications, a variety of security tests should be employed. Each type of test serves specific purposes and together, they provide a robust defense against potential security threats. Essential categories of security tests include:

- Vulnerability scanning
- Penetration testing
- Risk assessment
- Security posture assessment
- Vulnerability scanning

Vulnerability Scanning

Vulnerability scans use automated tools to check an app’s ecosystem for areas that can be compromised during an attack. Vulnerability scanners look for known vulnerabilities, particularly in software dependencies.

Vulnerability scanning also detects easily missed loopholes application code, checking against a record of common vulnerabilities and their characteristics. The matches are then reported to the developers or the quality assurance (QA) team.

Penetration testing

Penetration testing simulates attacks to test an app’s security and identify its weaknesses. This differs from vulnerability scanning in that it involves human input (in this case, an ethical hacker). They use several techniques to break into an app and check where attackers may take advantage.

Unlike vulnerability scanning, which can sometimes raise false positives, the threats identified by penetration testing are are typically actionable and realistic. These tests can usually provide more detail on a loophole’s precise location and how it could be exploited in real-world scenarios.

Risk assessment

Risk assessment involves identifying and evaluating all people, processes, and tools in an app’s ecosystem to identify their individual and collective risks in case of a cyber-attack. This involves cataloging assets, recognizing potential threats, and analyzing how vulnerabilities could be exploited.

The goal is to understand the severity of each risk in terms of its potential impact on operations, reputation, and finances, as well as its likelihood of occurrence. This information helps teams gain a holistic view of the threat landscape and make informed decisions to improve their security posture.

Posture assessment

Based on findings from a risk assessment, organizations prioritize risks and develop targeted mitigation strategies to enhance their security posture. Specific recommendations might include strengthening authentication procedures, updating and patching software, developing incident response plans, or implementing continuous monitoring tools for improved visibility.

Posture assessments may also include forms of compliance auditing, which ensure that all security practices align with relevant regulatory and industry standards. This can help safeguard against legal and financial penalties by ensuring that the organization meets required security obligations. Posture and risk assessments work hand in hand, and they may also incorporate other types of security testing. All these have a common goal, to help you identify security loopholes, prevent an attack, and mitigate risk.

Sources

Mariano, M. (2024, 26 agosto). *10 most common mobile application vulnerabilities*. I.S. Partners. Retrieved January 23rd, 2025, of https://www.ispartnersllc.com/blog/mobile-application-vulnerabilities/

Schmitt, J. (2022, 9 septiembre). *Mobile app security testing: Tools and best practices*. CircleCI. Retrieved January 23rd, 2025, of https://circleci.com/blog/mobile-app-security-testing/

Bawa, J. (s. f.). *Top 7 Mobile Application Vulnerabilities You Must Know*. Seasia Infotech Blog. Retrieved January 23rd, 2025, of https://newtiertech.com/blog/mobile-application-vulnerabilities/index.html

Support. (2025, 15 enero). Mobile application security: vulnerabilities to watch out for. *Blue Soft*. Retrieved January 23rd, 2025, of https://www.bluesoft-group.com/en/mobile-application-security-vulnerabilities-to-watch-out-for/