Mobile application testing
Mobile app security testing

With source code access

• Checking if best practices are followed e.g.
  • How is data stored?
  • How are web requests made?
  • Does the server trust data from app?

Without source code access

• Rooting
  • Rooting an Android phone or jailbreaking an iPhone
  • Access to all files, can install useful tools

• Monitoring and intercepting web requests

• Reverse engineering and code tampering (method hooking and swizzling)
Open Web Application Security Project

not-for-profit group that helps organizations develop, purchase, and maintain software applications that can be trusted

OWASP mobile Top 10: regularly-updated report outlining security concerns for mobile application security, focusing on the 10 most critical risks

Link: https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10
1. Improper platform usage  
   (misuse of a platform feature or failure to use platform security controls)

2. Insecure data storage  
   (covers insecure data storage and unintended data leakage)

3. Insecure communication

4. Insecure authentication  
   (problems authenticating the end user or bad session management)

5. Insufficient cryptography

6. Insecure authorization

7. Client code quality  
   (catch-all for code-level implementation problems in the mobile client)

8. Code tampering

9. Reverse engineering

10. Extraneous functionality  
    (hidden backdoor functionality or other internal development security controls not intended for production environment)
Statistics from NowSecure

- % mobile apps violate at least one top 10 risk
Statistics from NowSecure

• 85% mobile apps violate at least one top 10 risk

• First half of 2018 disclosed vulnerabilities:
  • 312 on Android
  • 87 on iOS

[Diagram showing OWASP Mobile Top 10 Violation Rates]
Improper platform usage

- misuse of a platform feature or failure to use platform security controls

- Might include:
  - Android intents
  - Platform permissions
  - Misuse of TouchID
  - Misuse the Keychain
  - Misuse of other security controls
Example: Citrix Worx apps

- Article: https://medium.com/@c0nk3r/bypassing-apples-touch-id-but-from-certain-apps-e7c6fbe3d848 (2016)
- Possible to bypass TouchId
- TouchId was not asked when app was closed and opened again after a reboot
- Problem: secret was saved and used again
Insecure data storage

• covers insecure data storage and unintended data leakage

• Might include:

  • Wrong keychain accessibility option
    (f. ex. kSecAttrAccessibleWhenUnlocked vs. kSecAttrAccessibleAlways)

  • Insufficient file data protection
    (f. ex. NSFileProtectionNone vs NSFileProtectionComplete)

  • Access to privacy resources when using this data incorrectly
Example: Tinder

- Feature: showing people logged on near you
- Problem: the exact location of every person near you was sent to the device
- Fix: provide distance only —> spoof location and triangulate
- Fix: data without precision
- An app was created by users to show tinder users with exact location
Insecure communication

- Might include:
  - poor handshaking/weak negotiation (f. ex. lack of certificate pinning)
  - incorrect SSL versions
  - cleartext communication of sensitive assets
  - HTTP
Example: Misafe smart watches


- Communication was not encrypted

- Not correctly authenticated

- Attackers could:
  - Retrieve real-time GPS coordinates of the kids’ watches.
  - Call the child on their watch.
  - Create a covert one-way audio call, spying on the child.
  - Send audio messages to the child on the watch, bypassing the approved caller list.
  - Retrieve a photo of the child, plus their name, date of birth, gender, weight and height.
Insecure authentication

• Problems authenticating the end user or bad session management

• Might include:
  
  • Failing to identify the user at all when that should be required

  • Failure to maintain the user's identity when it is required

  • Weaknesses in session management
Example: Grab Android app

- Able to bypass 2FA by brute forcing 4 digit code
- Problem: gain access to account
- Account has info on rides, payment methods, orders
Insufficient cryptography

- Cryptography applied but insufficient in some way
- Use standard libraries or write your own cryptographic algorithms?
Example: Ola app


- weak cryptographic key, did discover the key as PRODKEYPRODKEY12

- passwords also encrypted with this key
  -> all other accounts vulnerable if passwords reused

- could intercept request between app and server
  -> fake a request for money
  -> get money
Insecure authorization

- Failures in authorization (e.g., authorization decisions in the client side, forced browsing, etc.)

- Able to execute over-privileged functionality

- Distinct from authentication issues (e.g., device enrolment, user identification, etc.)
Example: Viper smart start


• Problem:
  • App makes request to server
  • After login: no real authentication, can change id number
  • Access to all information about car, location
  • Can change data, open car etc.
Client Code Quality

- catch-all for code-level implementation problems in the mobile client

- Might include:
  
  - buffer overflows

  - format string vulnerabilities

  - various other code-level mistakes where the solution is to rewrite some code that's running on the mobile device.
Example: WhatsApp

• Article: https://thehackernews.com/2019/05/hack-whatsapp-vulnerability.html (2019)

• Making a call and sending a specially crafted series of packets —> Buffer overflow

• Call does not need to be answered

• Adversary can run arbitrary code

• Used to install spyware that spies on users
Code tampering

- Might include:
  - binary patching
  - local resource modification
  - method hooking and swizzling
  - dynamic memory modification
Example: Pokemon GO


- Fans reverse engineered the application

- Fed wrong geolocation data and time to find rare pokemon and make eggs hatch faster
Reverse engineering

- Might include analysis of the final core binary to determine its
  - source code
  - libraries
  - algorithms and other assets
- Exploit other vulnerabilities in the application
- Reveal information about backend servers, cryptographic constants and ciphers, and intellectual property.
- Used for most examples above
Extraneous Functionality

• Might include:
  • Hidden backdoor functionality
  • Other internal development security controls not intended for production environment
Example: Wifi File Transfer


• App opens port on Android device to allow connection from computer

• Intended use: transfer files, photos, anything stored on SD card

• Problem: no authentication like password, anyone can connect to device and have full access

• In total in Google Play store:
  • 1,632 apps open ports
  • 410 have potentially no or weak protection
  • 57 apps confirmed as exploitable
How to develop a secure app?

- Follow best practices. Some examples:
  - do not write your own cryptographic algorithms
  - do not use hard coded secret keys
  - do not trust the client
  - hash passwords
- Familiarise yourself with OWASP mobile top 10, look at
  - "Am I Vulnerable To <risk>"
  - "How Do I Prevent <risk>?"
- Example Attack Scenarios
For more detailed information:

- OWASP Mobile Security Testing Guide
  https://github.com/OWASP/owasp-mstg

- Apple’s Secure Coding Guide

- Android application security
  https://source.android.com/security/overview/app-security
Found a security vulnerability?

- In your app:
  - Fix vulnerability, study the extent of vulnerability/breach
  - Notify users of a possible breach of data

- In a 3rd party app:
  - Responsible disclosure
  - Inform the company or coordinating entity

- Risks?
- Deadline?
- Company has time to fix vulnerability
- Disclosure to public
Found a security vulnerability?

• In your app:
  • Fix vulnerability, study the extent of vulnerability/breach
  • Notify users of a possible breach of data

• In a 3rd party app:
Found a security vulnerability?

• **In your app:**
  - Fix vulnerability, study the extent of vulnerability/breach
  - Notify users of a possible breach of data

• **In a 3rd party app:**
  - Responsible disclosure
  - Inform the company or coordinating entity
    - Risks?
    - Deadline?
  - Company has time to fix vulnerability
  - Disclosure to public

https://www.div0.sg/single-post/2017/06/03/Responsible-Disclosure