proofpoint

Best Practices for Writing Secure iOS and Android Apps

Mobile Defense

The following guidelines should be used when developing apps for iOS and Android. These guidelines address security, and should be followed in addition to standard coding best practices.

1. Use native SSL libraries on the OS. Some third party libraries have vulnerabilities such as HeartBleed, which can be exploited for man-in-the-middle attacks.

2. Use mutual SSL authentication in your apps to validate server connections to ensure that your app is communicating to the server you expect and not a man-in-the-middle.

3. All app communication should be encrypted. Do not disable this in iOS 9.

4. Pin certificates use for encrypted communications and mutual authentication. Do not rely on root certificates stored in the OS, as new roots can be added, which can lead to man-in-the-middle attacks.

5. Avoid using precompiled third party libraries, since you do not know what they do. These could be ad libraries, encryption libraries and graphics libraries. Do not download libraries from non-trusted parties.

6. Only enable inter-app communication for apps you trust and that your app must communicate with.

7 Ensure inter-app communication is encrypted.

Do not store files unencrypted.
Ensure that encryption libraries are fully utilized. 8

9 Do not store non-essential personally-identifiable information inside your app.
It is better to download this from servers as needed.

Do not store passwords on the device.
If you must, store only a hash of the password. 10

11 iOS apps should store secrets and credentials in the KeyChain.
This leverages the security that is built into the KeyChain.

Do not send confidential information via SMS or APNS messages.
They can be read by anyone who has access to the phone, even if the phone is locked. 12

13 Be wary of any plug-ins that your app uses. These are often vectors for introducing security vulnerabilities into apps.

Only use code and development tools from trusted vendors, and only download core development libraries and tools from Apple or Google's actual download sites. Do not download development tools from unofficial sites, or your app could be infected with malware such as XcodeGhost. 14

15 Enable Position Independent Execution when compiling your app. PIE is important to reduce the chance that malicious apps and tools can access known memory locations in your app.

16 Only declare permissions that you actually need and use in the app. Do not just copy a permissions list from a generic app.

17 Be very wary of embedding API keys into your app where the API can be used to access sensitive data or accounts on cloud services. Assume API keys have been copied (because they can be copied from any jailbroken or rooted phone). Ensure that access controls are enforced by the addition of a password or other user credential that is entered by the user and not common to all apps.

18 Financial services apps should consider adding code to check for jailbroken and rooted phones, and not allow transactions from compromised phones.

19 Have code reviews by team members or external parties.

20 Have a privacy policy that accurately describes what your app and your servers do with data. Have your privacy policy reviewed by internal legal counsel. Ensure that your privacy policy is published with your app and linked to your app store entry.

21 Analyze your app with Proofpoint Mobile Defense tools to validate its behavior prior to release.

about proofpoint

Proofpoint Inc. (NASDAQ:PFPT) is a leading security-as-a-service provider that focuses on cloud-based solutions for threat protection, compliance, archiving & governance, and secure communications. Organizations around the world depend on Proofpoint's expertise, patented technologies and on-demand delivery system to protect against phishing, malware and spam, safeguard privacy, encrypt sensitive information, and archive and govern messages and critical enterprise information.

proofpoint™

892 Ross Drive
Sunnyvale, CA 94089

1.408.517.4710
www.proofpoint.com