



KEEPING THE INTERNET
OPEN • INNOVATIVE • FREE

www.cdt.org

1634 I Street, NW
Suite 1100
Washington, DC 20006

P +1-202-637-9800
F +1-202-637-0968
E info@cdt.org

May 30, 2014

Federal Trade Commission
600 Pennsylvania Avenue N.W.
Room H -113 (Annex B)
Washington, DC 20580

Re: Mobile Security Project, Project No. P145408

The Center for Democracy & Technology¹ (CDT) is pleased to submit comments in response to the Federal Trade Commission's (FTC) call for submissions on mobile security to further explore issues raised by last year's FTC forum examining the state of mobile security. In our comments we focus on a few discrete questions posed by the RFC, highlighting areas for improvement in the creation and promulgation of mobile security standards.

I. Secure Distribution Channels

• Is application review and testing scalable given the explosive growth of mobile applications? What techniques have proven effective in detecting malicious or privacy-infringing applications?

Apple and Google are the two major mobile platform providers in the market today, and the associated operating systems – iOS and Android – come with their own advantages and disadvantages. While both platforms have app stores where users can search for and download new apps and updates, the providers have chosen different approaches to app store requirements, guidelines, and best practices available to developers. Both platform providers have guidelines in place that tell developers what kind of apps can and cannot be uploaded to the app store. Google publishes its guidelines online while Apple keeps its guidelines confidential; they are made available to registered Apple developers. Becoming an Apple developer is a non-zero yearly cost and registered developers are bound by a non-disclosure agreement not to disclose app store requirements.

¹ CDT is a non-profit Internet and technology advocacy organization working to keep the Internet and digital life open, free, and innovative. CDT promotes public policies that preserve privacy.

This lack of transparency makes it challenging to compare requirements and guidelines for each of the app stores as well as compare them to other sources of best practices in privacy and security. The two providers' platforms also differ when it comes to reviewing apps that as submitted to the app stores by developers. While Apple reviews and tests² every mobile app submitted to the App Store before it is made available to end users, Google scans³ the Android Market for potentially malicious software but it does not require developers to go through an application review and approval process. In February 2012, Google announced a service, codenamed Bouncer, which provides automated scanning of the Android Market for potentially malicious software. The service had been running for some time at the time of the announcement and Google reported a 40% decrease in the number of potentially malicious downloads from the Android Market in the first and second halves of 2011. Ongoing, dynamic programs such as these may be beneficial in detecting and reducing the number of malicious apps.

II. Secure Development Practices

• *What resources (e.g., application programming interfaces, development guides, testing tools, etc.) are available for third-party developers interested in secure application development?*

There are a number of resources available to app developers on the topic of secure application development, but developers must seek out that information. In addition to the secure coding guidelines and best practices published by both Apple⁴ and Google⁵, the Mobile Security Project by the Open Web Application Security Project (OWASP)⁶ is another strong resource covering tips and tricks for secure mobile application development. OWASP also maintains a list of the top 10 mobile security risks⁷ present today, as well as a list of both free and commercial source code analysis tools⁸.

While both Apple and Google review applications for malicious software, it is unclear whether applications are also reviewed to assess how well they follow secure coding guidelines and best practices, or if they behave insecurely. An application that is ignorant of best practices designed by the developer to send sensitive data over an unencrypted connection may not be considered to behave maliciously, but tools that alert developers to problems like these should be in place.

² *App Review*, Apple Developer Website (May 16, 2014, 11:00 AM), <https://developer.apple.com/app-store/review/>.

³ Hiroshi Lockheimer, *Android and Security*, Google Mobile Blog (May 16, 2014, 9:57 AM), <http://googlemobile.blogspot.com/2012/02/android-and-security.html>.

⁴ *Introduction to Secure Coding Guide*, Apple Developer Website (May 18, 2014, 10:28 AM), <https://developer.apple.com/library/mac/documentation/security/conceptual/SecureCodingGuide/Introduction.html>.

⁵ *Best Practices for Security & Privacy*, Android Developer Website (May 18, 2014, 10:29 AM), <http://developer.android.com/training/best-security.html>.

⁶ *OWASP Mobile Security Project*, Open Web Application Security Project (May 18, 2014, 10:30 AM), https://www.owasp.org/index.php/OWASP_Mobile_Security_Project.

⁷ *Top 10 Mobile Risks*, Open Web Application Security Project (May 18, 2014, 10:32 AM), https://www.owasp.org/index.php/OWASP_Mobile_Security_Project#tab=Top_10_Mobile_Risks.

⁸ *Source Code Analysis Tools*, Open Web Application Security Project (May 18, 2014, 10:47 AM), https://www.owasp.org/index.php/Source_Code_Analysis_Tools.

• *Is the developer community taking advantage of these resources? Are they making common security mistakes?*

As previously discussed, there are freely available resources online to help developers write trustworthy code. However, it is unclear to what extent these guidelines serve simply as informational guidance or if they serve as requirements that can be enforced during the application review process. That said, it may be difficult to turn secure coding guidelines into strict evaluation requirements, as different applications will have different needs. An application that stores information unencrypted on a smartphone may not necessarily be malicious – intentionally behaving against the user’s interests – and the application may not be insecure. Reasonable and robust security standards depend on the type of application and what kind of functions it performs to what kind of data it stores and transfers.

One major challenge for mobile applications developers is securely using third-party ad libraries for the purpose of generating added revenue. Studies⁹ have shown how these ad libraries – regardless of the platform – are often insecure, out of date, or request far more information from an end user’s device than what is strictly necessary.

• *Do consumers have the information they need to evaluate the security of an application? Are they aware of potential security risks (e.g., the insecure transmission of data)? Are there ways to make the security of applications more transparent to the end-user?*

There is unfortunately no easy way for consumers to evaluate the security of an application. Doing so today involves a combination of trusting the application and/or the app store, reading reviews online, and asking colleagues and friends for input. Most consumers will rarely question the security of an application and instead assume that data is stored and transferred securely.

While both Apple and Google have removed¹⁰ malicious apps from their respective app stores in the past, it is rare to see poorly insecure or outdated apps being removed. Last year, Google reportedly¹¹ removed a large number of applications that were using an outdated and vulnerable version of the third-party ad library AppLovin. However, having an application removed from an app store does not mean that it is removed from each user device upon which it has been installed. It is still possible to use an application that no longer exists in the app store, but updates will no longer be available. Without any kind of indication that an app is no longer being actively developed or that it may be insecure, app users have no basis for questioning their ongoing use of these apps.

Platform providers should also be encouraged to inform the public about the decision making process behind pulling applications from the app stores; e.g. applications that are clearly malicious, versus applications that are using outdated and vulnerable libraries. Transparency

⁹ Yuvraj Agarwal & Malcom Hall, *ProtectMyPrivacy: Detecting and Mitigating Privacy Leaks on iOS Devices using Crowdsourcing* (May 18, 2014, 11:21 AM), http://www.synergylabs.org/files/Agarwal_MobiSys2013_ProtectMyPrivacy.pdf; Theodore Book, Adam Pridgen, and Dan S. Wallach, *Longitudinal Analysis of Android Ad Library Permissions* (May 18, 2014, 11:23 AM), <http://pubs.cs.rice.edu/sites/pubs.cs.rice.edu/files/book-most2013.pdf>.

¹⁰ Lucian Constantin, *Google, Apple Remove Malware Application From Official App Stores*, PC World (May 18, 2014, 12:05 PM), http://www.pcworld.com/article/258870/google_apple_remove_malware_application_from_official_app_stores.html.

¹¹ Liam Tung, *Over 200m Android devices exposed to buggy AppLovin ad library*, CSO (May 18, 2014, 12:09 PM), http://www.cso.com.au/article/532086/over_200m_android_devices_exposed_buggy_applovin_ad_library/.

enables more accountability from platform providers, and allows for consumers to more accurately and effectively exercise their consent to particular services.

• *What more can platforms and other industry players do to ensure that third-party developers have the resources and incentives necessary to implement secure development practices?*

The platform providers should do more to highlight the secure application development guidelines that are available, attempt to enforce best practices where feasible through algorithmic inspection and, at the same time, work to make the application lifecycle for a given app store more transparent – from development to review and inclusion in the store to after-market monitoring.

Last year, UK-based computer security firm MWR Labs documented¹² critical security issues in a number of ad libraries affecting all Android platforms and apps using the vulnerable libraries. In this case, the developers using these libraries were not informed about these critical security issues or new releases of the ad libraries that fixed these issues. Developers often have little incentive to invest in updating their applications with what may seem to be a minor update to a third-party library. Platform providers and major industry players could do more to ensure that app developers know about new updates and security fixes. This can be accomplished through periodic disclosures via email, push notifications, or other means.

III. Security Lifecycle and Updates

• *What are consumer expectations with respect to the security lifecycle of their mobile devices? Do consumers have the appropriate information (e.g., at the time of purchase) to factor security into their device purchasing decision? Do consumers receive notice when a device has reached “end-of-life” with respect to security support?*

The majority of consumers likely expect that their devices are secure as long as they are operational and the main functions still work as expected. Security updates and product “end-of-life” dates are not critical selling points at the time of purchase. Additionally, there is no mechanism for manufacturers to alert users when a device reaches “end-of-life.”¹³ The lack of an effective security update mechanism for some platforms has become a serious problem. Last year, the American Civil Liberties Union (ACLU) filed a complaint¹⁴ with the FTC asking the agency to investigate the major wireless carriers for failing to warn their customers about unpatched security flaws in the software running on their phones.

¹² *AppLovin Ad Library SDK: Remote Command Execution via Update Mechanism*, MWR Labs (May 18, 2014, 12:36 PM), <https://labs.mwrinfosecurity.com/blog/2013/11/20/applovin-ad-library-sdk-remote-command-execution-via-update-mechanism/>; *Millennial Media Ad Library*, MWR Labs (May 18, 2014, 12:37 PM), <https://labs.mwrinfosecurity.com/blog/2013/11/27/millennial-media-ad-library/>; *PontiFlex Ad Library – Remote JavaScript Command Execution*, MWR Labs (May 18, 2014, 12:37 PM), <https://labs.mwrinfosecurity.com/blog/2013/12/20/pontiflex-ad-library---remote-javascript-command-execution/>; *Google AdMob Ad Library – Arbitrary Intent Activity Invocation*, MWR Labs (May 18, 2014, 12:38 PM), <https://labs.mwrinfosecurity.com/blog/2013/12/20/pontiflex-ad-library---remote-javascript-command-execution/>.

¹³ See e.g., Gordon Kelly, *Microsoft Abandons Windows 8.1: Take Immediate Action or Be Cut Off Like Windows XP*, *Forbes* (April 15, 2014, 7:24AM), <http://www.forbes.com/sites/gordonkelly/2014/04/15/microsoft-abandons-windows-8-1-take-immediate-action-or-be-cut-off-like-windows-xp/>.

¹⁴ Chris Soghoian, *ACLU Files FTC Complaint Over Android Smartphone Security*, ACLU (May 18, 2014, 12:58 PM), <https://www.aclu.org/blog/technology-and-liberty/aclu-files-ftc-complaint-over-android-smartphone-security>.

In some sense, users with smartphones running older OS versions are lucky if critical security updates are applied to their older OS versions. For example, it was a relief to many iOS 6 users running on older iOS devices – some iPhone models will not run the newer iOS 7 – when the security update for the GotoFail bug was simultaneously applied to both iOS 6 and 7. Encouraging mobile OS providers to create protections and adequate notice for older versions of their operating systems will promote stronger security by encouraging a broader range of users to upgrade to more secure releases.

• What are the challenges in creating, testing, and distributing security updates to end-user devices? What, if any, are the implications of slow update cycles? Are there steps that platforms, manufacturers, telecommunications carriers, and other players can take to streamline this process?

The implications of slow security update cycles can be serious. For Android, fixes for the core OS aren't packaged and distributed to consumers by the OS manufacturer, but must be coordinated by both the wireless carrier and the handset manufacturer, a process that can be very slow if it occurs at all. Given the variety of Android handsets, potentially both the wireless carrier and handset manufacturer have to agree that there exists a business case to offset the costs of pushing an Android OS update. For iOS, the issue is that updated apps are released to the App Store with a delay, due to the App Store's review and testing process. Slow update cycles mean that consumers are left vulnerable while the existence of security holes are widely known for attackers to take advantage of.

An acute example of this was Apple's recent GotoFail bug. On February 21, 2014, Apple pushed a security update for iOS to patch a bug in its implementation of SSL/TLS. Without this security update, an attacker could easily listen in on what normally would have been encrypted traffic while a user sent emails, updated her calendar, tweeted, used Facebook, and so on. Independent researchers quickly determined that the same bug also affected Apple's laptop and desktop operating system, OS X, including applications such as FaceTime, Mail, and Software Update. It took Apple almost four days to distribute a fix for OS X. While immediate responses are of course not always feasible, working to create and release updates that may depend on one another – and providing a more specific timeline for those updates – would provide more certainty and reassurance to consumers.

IV. Conclusion

We thank the Commission for soliciting additional comments following the successful workshop last year on mobile security.

Sincerely,

/s/

Joseph Lorenzo Hall
Chief Technologist; CDT

/s/

G.S. Hans
Ron Plessner Fellow; CDT