Department of Homeland Security
Science and Technology Directorate

Securing Mobile Applications for First Responders

December **6**, 2017

Version 1.0

Table of Contents

Table of Figures

1 Introduction

This report describes a mobile application (app) pilot testing program designed to serve a public safety purpose. The Department of Homeland Security Science and Technology Directorate (DHS S&T), the Association of Public-Safety Communications Officials (APCO) International, and Kryptowire LLC collaborated to identify security vulnerabilities and privacy issues important for public safety users and to recruit app developers to participate in testing and evaluation. This report describes findings from the testing, feedback from the developers who participated in the pilot, technical and program-level lessons learned, and recommended next steps.

1.1 Background

Consumers and businesses rely on mobile devices and mobile apps for daily communications, consumption of news and information and delivery of services. In emergency and disaster situations, mobile devices and mobile apps enable first responders and public safety professionals to receive and share critical information in real-time, enabling delivery of life-saving services. However, as our reliance on mobile technology continues to increase, mobile apps have become the new target for cyberattacks using malware, ransomware, spyware and app-coding vulnerabilities that may expose personal data, drain the device's battery, compromise the security of the device altogether or provide fraudulent information resulting in the disruption of time-critical services. The pace of changing technology—new apps, app updates, device operating system updates, service provider updates—presents a broad and varied attack surface subject to new threats, vulnerabilities and exploits. Unfortunately, users have few options to assess the security state of apps; apps may be benign, malicious or contain errors that pose risk to the user. Even the official Android and iOS mobile app stores are not immune to malware and apps that contain bugs and vulnerabilities.[1,2] The consequences of app vulnerabilities are especially critical when the apps are intended for public safety or emergency response.

APCO created the Application Community (www.AppComm.org) to serve as the single trusted site for public safety apps. APCO has engaged in several efforts to ensure that public safety apps are safe and effective. APCO previously worked with DHS S&T's First Responders Group (FRG) and the Public Safety Communications Research (PSCR) program of the National Institute of Standards and Technology (NIST) to identify security requirements for public safety mobile apps. The initiative sought to identify and categorize the different types of public safety data needed by first responders that could be used by the apps and examine how those data types influence cybersecurity requirements for mobile apps. The resulting evaluation pilot project focus was to improve mobile app security for the first responder and broader public safety community.

The consequences of app vulnerabilities are especially critical when the apps are intended for public safety or emergency response.

[1] iOS XcodeGhost Malware, https://researchcenter.paloaltonetworks.com/2015/09/novel-malware-xcodeghost-modifies-xcode-infects-apple-ios-apps-and-hits-app-store/

[2] Android Dresscode Malware, https://blog.checkpoint.com/2016/08/31/dresscode-android-malware-discovered-on-google-play/

Prior DHS S&T work in this area involved funding Kryptowire LLC, a mobile app vetting solution provider, to automate mobile app vetting based on government standards (i.e., National Information Assurance Partnership [NIAP] Requirements for Vetting Mobile Apps from the Protection Profile for Application Software[3]). S&T invested in this research to address the federal government's need for standardized, cost-effective, automated methods and tools to develop, vet, deploy and manage mobile apps—a key enabler to the federal government's adoption of mobile technologies. Today, the S&T-funded testing platform is used by several federal agencies to test mobile apps used by the federal government.

For this pilot, S&T provided the funding and technical support through its funded research with Kryptowire. APCO selected a set of apps for testing, Kryptowire provided access to its testing platform, which was integrated with AppComm to streamline the testing process, and Kryptowire tested the apps based on security criteria identified by the pilot partners.

1.2 Pilot Partners

DHS S&T Cyber Security Division (CSD): The Mobile Security Program aims to accelerate the safe and secure adoption of mobile technologies by government and industry to enable the homeland security mission. The program has three research and development (R&D) efforts: mobile device security, mobile app security, and security and resilience of mobile network infrastructure. CSD's mobile app security R&D project with Kryptowire aims to establish continuous automated assurance of mobile apps for the federal government. By combining mobile app archiving and app vetting technologies as well as incorporating government and industry security standards, the project captures app changes made over the app's lifecycle and tests against known vulnerabilities and emerging threats.

DHS S&T FRG: The First Responder Group focuses specifically on providing the nation's 3.3 million first responders the tools they need to increase safety and effectiveness as well as positioning American companies to be highly competitive in the public safety industry sector. FRG identifies, validates and facilitates the fulfillment of first responder capability gaps through the use of existing and emerging technologies, knowledge products and the acceleration of standards. FRG manages working groups, teams and other stakeholder outreach efforts to better understand the needs and requirements of local, tribal, state and federal first responders, including those on the front line of border protection and transportation security.

APCO International: APCO is the nation's oldest and largest organization of public safety communications professionals with more than 30,000 members, primarily consisting of state and local government employees who manage and operate public safety communications systems—including 911 Public Safety Answering Points (PSAPs), emergency operations centers, radio networks and information technology—for law enforcement, fire, emergency medical and other public safety agencies. APCO serves the needs of public safety communications practitioners worldwide—and the welfare of the public as a whole—by providing expertise, professional development, technical assistance, advocacy and outreach. APCO's AppComm maintains a listing of more than 180 public safety and emergency response apps.

[3] Requirements for Vetting Mobile Apps from the Protection Profile for Application Software

Kryptowire LLC: Kryptowire's mobile app vetting platform automatically tests and validates the security of mobile and IoT firmware and apps to the highest government and industry software assurance standards. Kryptowire, which was jumpstarted by the Defense Advanced Research Projects Agency (DARPA) and DHS in 2011, is based in Fairfax, Virginia, USA and has a customer base ranging from government agencies to national cable TV companies.

2 Pilot Overview

The app testing pilot sought to determine the degree to which public safety apps are vulnerable and to lay the foundation for a sustainable model for testing the security and privacy of public safety mobile apps. Using APCO's AppComm to identify popular public safety apps, the pilot consisted of 33 apps (counting iOS and Android versions separately) created by 20 developers that were tested over a three-month period. Kryptowire provided access to its mobile app software testing platform and mobile app security experts at DHS and NIST identified a subset of the government's NIAP criteria as testable app characteristics most relevant to public safety users.

2.1 Methodology

Figure 1 illustrates the pilot workflow. The pilot involved:

- Identifying a subset of the most frequently used public safety apps from AppComm
- Recruiting mobile app developers to participate in app testing
- Establishing a registration process and submission portal for app developers
- Submitting the apps to Kryptowire directly from AppComm (no source code needed)
- Using a combination of dynamic testing and code analysis to evaluate the mobile apps
- Generating confidential app testing reports for app developers/developer organizations
- Engaging in a remediation dialog with app developers
- Resubmitting the app for evaluation after remediating or providing rationale for the identified app issues
- Reviewing test results and developer responses and making an app suitability determination

Note: Future steps may include posting of the mobile app in the directory if all checks are passed and a positive determination is given.

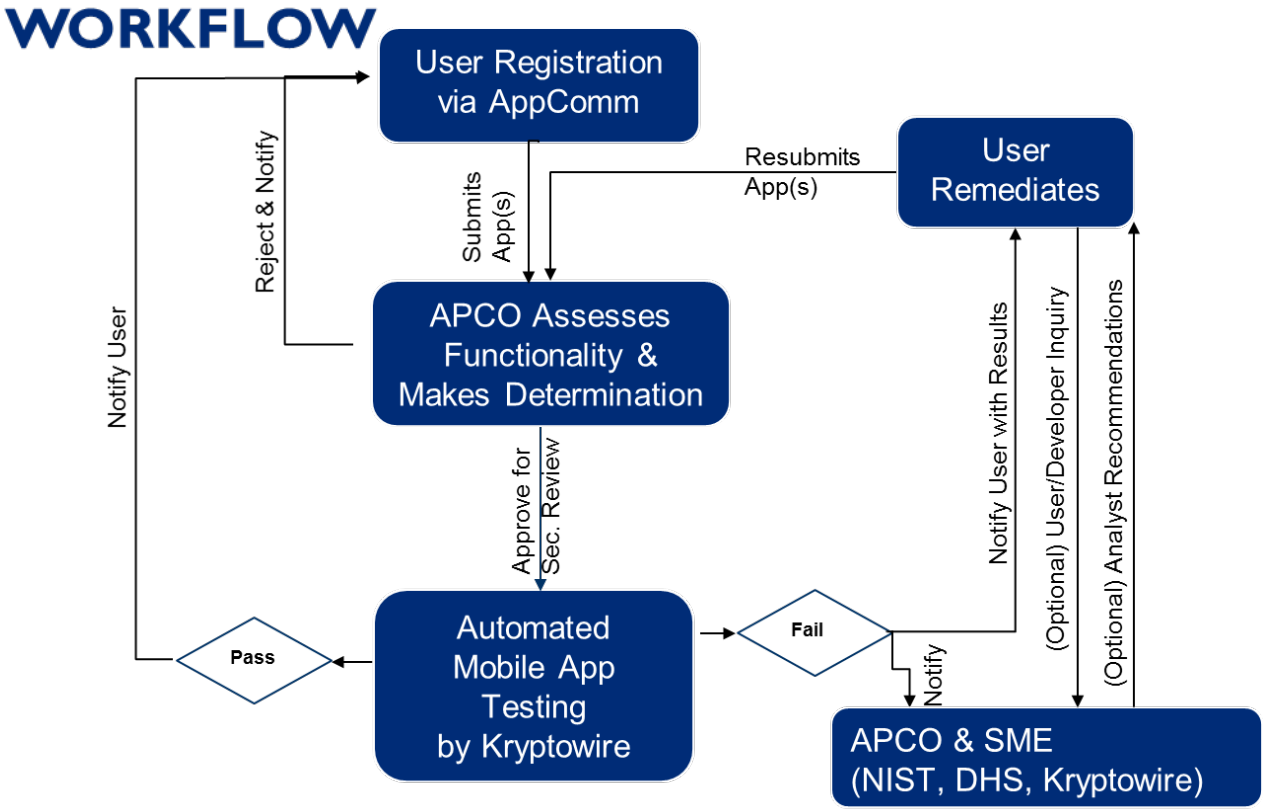

Figure 1. Mobile App Vetting Pilot Workflow

2.2 Timeline

During a kick-off meeting on April 20, 2017, the goals of the pilot and testing process were explained to app developers. Kryptowire then conducted testing and produced reports for developers that spelled out items requiring remediation. After the testing and remediation efforts were completed, the partners solicited feedback from developers on the process and the criteria and presented the findings and results at APCO's Annual Conference.

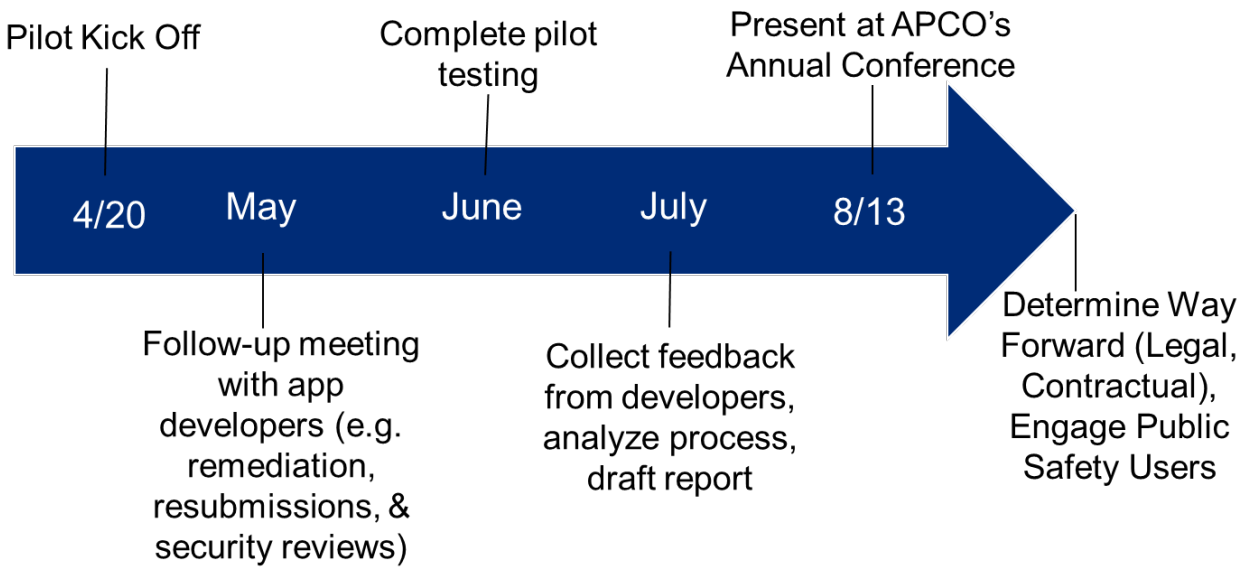

Figure 2. Mobile App Vetting Pilot Timeline

2.3 Testing Criteria

The mobile app analysis tools produce concrete evidence about potential security or privacy issues in an app's code and the conditions under which the issue is triggered. This evidence includes any potential code vulnerabilities, bad coding practices or weaknesses an app might exhibit. The test report forms the basis to inform APCO and the developer about potential vulnerabilities in the app. The report identifies the specific code that contains the offending capabilities, enabling developers to use the report as a guide to fix the identified issues. The analysis approach leveraged static and dynamic[4] analysis methods to explore the code and behaviors of the target apps, including any third-party code and libraries used by the app, and report all of the app's performed activities, network communications and program functionality. Armed with this information, security and privacy concerns can be identified as items for the developer to either fix or justify as necessary risks for the app's proper functionality. The security evaluation categories are explained in the following sections. The principal areas evaluated for each evaluation category are listed in Appendix A and described in detail in the sample Kryptowire test report.[5, 6]

2.3.1 Security

Kryptowire's analysis scans for various security issues that can be present in apps and make them vulnerable to exploitation. These issues can lead to sensitive data handled by the app to be compromised by malicious parties. Each issue should be considered carefully by the reviewing

[4] Static analysis techniques examine the code without running the app, these techniques provide insights into the properties of the app and can detect many vulnerabilities, while dynamic analysis techniques reveal app behaviors that only occur at runtime.

[5] Sample report for Android: www.kryptowire.com/apco-pilot/android-report.pdf

[6] Sample report for iOS: www.kryptowire.com/apco-pilot/ios-report.pdf

party and developers to ensure apps are as secure as possible before they are distributed to users. Examples of security items analyzed are:

- Use of proper encryption practices
- Malware scans
- Verification of encrypted network communication

2.3.2 Privacy and Information Access

Mobile apps potentially can access a wealth of sensitive information about the user and/or device. Findings in this evaluation category range from integration of the app with ad networks to improper handling of a user's password. Developers should justify that each piece of information accessed by the app is necessary to the core functionality needed for its operation. Examples of privacy and information access items are:

- Proper usage of a user's credentials
- Accessing the user's calendar and contacts
- Obtaining unique device information such as device ID or Subscriber Identity Module (SIM) serial number

2.3.3 Device Access

Most developers who completed the pilot reported spending approximately one hour to remediate identified concerns.

This category seeks to identify which sensitive device functionality the app might access. The analysis identifies both the use of the functionality as well as the context of use to better understand the nature of the access. As with privacy and information access, the evaluation should include determining whether the app requires the functionality for its intended purpose. Examples of device access items include:

- Access to wireless communications (Wi-Fi, Near-Field Communication [NFC], Bluetooth)
- Use of recording functionality such as camera and microphone
- Communication with outside parties through phone calls, Short Message Service (SMS) or Multimedia Messaging Service (MMS) messages

3 Pilot Results and Findings

The pilot test evaluated 33 mobile apps (18 iOS and 15 Android) from 20 developers. Summary pilot results include:

- Of the 33 mobile apps evaluated, 32 had security or privacy concerns (e.g., access to camera, contacts, or SMS messages);18 of the apps contained critical flaws[7](e.g., hard-coded credentials stored in binary, app accepts all Secure Sockets Layer (SSL) certificates and is susceptible to man-in-the-middle attacks).

[7] Critical flaws are defined as apps with Red Flags (see Appendix A)

- Of the 20 app developers that began the pilot, 10 completed it. Some companies and developers dropped out due to lack of time, perceived level of difficulty to fix identified concerns/issues or did not respond after the pilot's kick-off.
- Because 10 developers dropped out of the pilot, the security and privacy concerns identified for the 14 mobile apps were addressed.
- Most participating developers considered the app testing results and remediation process helpful and indicated a willingness to pay for security evaluation if approval can be provided by an authoritative body or organization.
- Developers who completed the pilot reported that the pilot test results were easy to interpret. Most reported spending approximately one hour on remediation of identified concerns/issues.

3.1 Providing Context for Identified Concerns and Issues

To better assist developers in tracking down and solving the identified issues—where possible—each finding included context to support the adverse result. This contextual information may include specific network traffic or line numbers in decompiled code that point to the issue. Kryptowire identifies the specific location in the app that does not meet the security requirements, allowing developers to automatically identify and remediate the security concerns/issue.

It also is important to help developers understand the specific nature of the security threats and why it should be of concern to developers and, ultimately, to the app's users. An example of one item detected in the app evaluation process is improper usage of SSL certificates in Android apps. Any time the mobile app communicates with a third-party via the web it can do so in an unencrypted (standard Hyper Text Transfer Protocol [HTTP]) or encrypted (HTTPS using SSL certificates) fashion. Using HTTPS is preferred as use of unencrypted HTTP allows anyone on the same network to intercept the data being sent or received by the app.

However, using HTTPS alone does not provide security assurance if the SSL certificates used for encryption are not handled properly. Each certificate is signed by a trusted third-party to verify the identity of the host that was issued the certificate. This preferred approach gives the mobile app assurance that the party it is communicating with is who they say they are, but only if the app properly verifies the certificate. If not, a malicious actor positioned between the mobile app and the target of the communication session can intercept and even modify the traffic.

Another example of an issue identified during analysis is the presence of hard-coded credentials within the app code. Typically, when any sensitive operation takes place in an app such as encrypting data or communicating with a secured back-end server credentials must be provided to complete the function. The proper way to handle these credentials is to derive them from a user's password or personal identification number or to retrieve them once from a server and store them in a secure, platform-provided location on the device. Occasionally, app developers leave the credentials hard-coded inside an app's code. This approach provides an attacker the ability to extract the credentials from the app. Equipped with these credentials, the attacker can access any information or service that requires authentication such as encrypted data or sensitive back-end services.

3.2 Example Findings

As shown in Figure 3, findings from analysis of the 33 apps fell into two categories:

1. Red flag items that require developer action
2. Orange flag items that require developer explanation

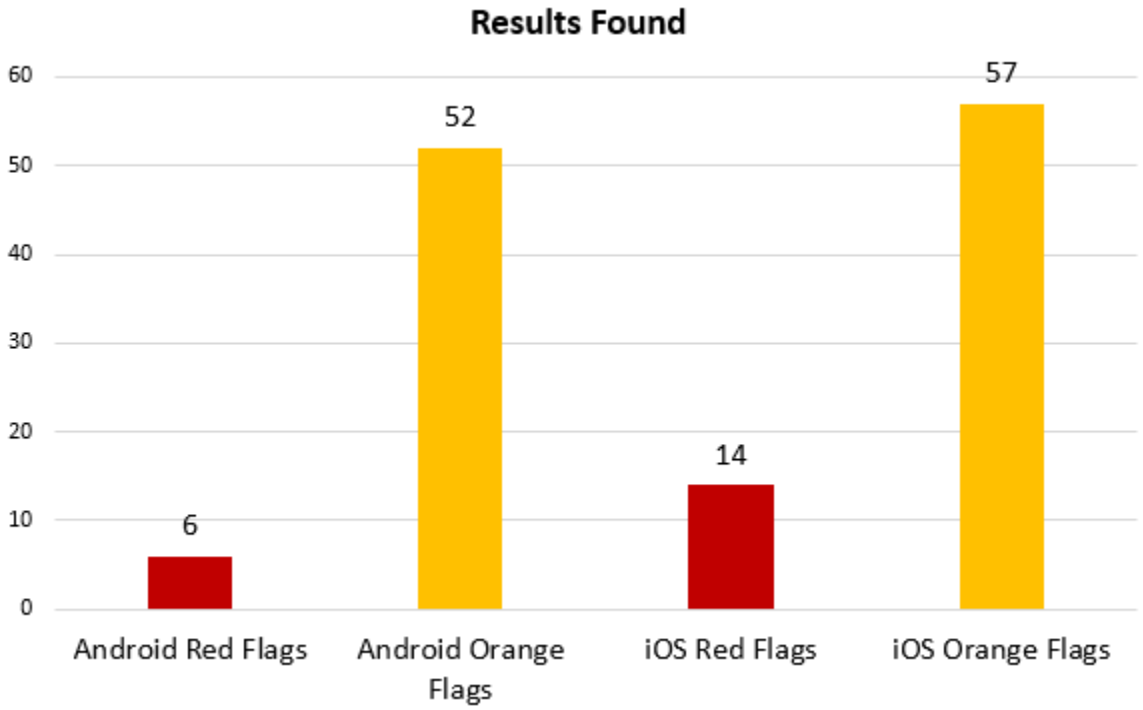

Figure 3. Summary Findings

Figure 4 shows the frequency of vulnerabilities found across all iOS and Android apps evaluated against the items listed in Appendix A. Note that some vulnerabilities are unique to the particular operating system. Only one app of the 33 tested did not have an orange or red flag. Of the 15 Android apps tested, five had red flags and 15 had orange flags. Of the 18 iOS apps analyzed, 13 had red flags, and 17 had orange flags.

App Security Concerns Identified	iOS (18 Apps Tested)	Android (15 Apps Tested)
Accesses Camera	5	0
Accesses Contacts	2	1
Disables Apple's TLS Enforcement	10	N/A
Personally Identifiable Information (PII) Exposure	4	2
Records Audio	7	7
Sends SMS	3	1

Uses Hard Coded Credentials	0	2
Vulnerable SSL	N/A	2

Figure 4. Frequency of Security and Privacy Concerns Discovered in Apps

Figure 5 and Figure 6 replicate example findings provided to developers, requiring developer action or explanation. Each finding included information indicating why the item was important as well as any context for the item to assist the developer in resolving or tracking down the issue. Orange flag items that required developer explanation contained fillable form fields the developer could use to provide a response, then be returned for validation by Kryptowire and APCO.

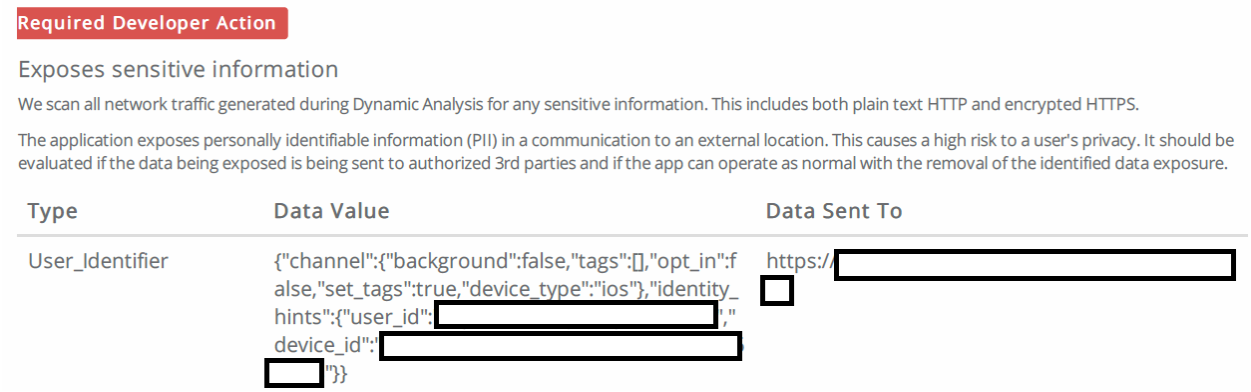

Figure 5. Example Finding: Exposes Sensitive Information

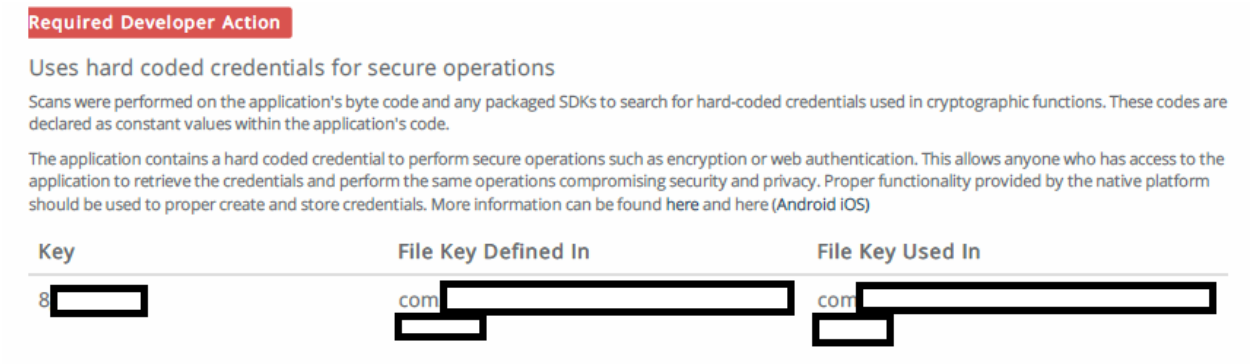

Figure 6. Example Finding: Uses Hard-Coded Credentials

3.3 Remediation Process

After each app developer provided responses to the identified concerns, a feedback loop was created by which APCO and Kryptowire verified the developer's responses or replied to them with additional comments or questions. Developers also were able to request clarification or provide feedback regarding testing or any of the provided results. Providing developers access to Kryptowire to ask questions proved to be very efficient. For two developers, separate calls were

set up with Kryptowire to discuss each developer's questions about the analysis results. In both cases, direct communication between the two parties led to an expeditious resolution of the identified security issue and enabled the pilot to proceed.

Many developers were able to use the results provided in Kryptowire's reports to make changes to their app enhancing security. For example, Kryptowire's analysis showed that certain apps had disabled Apple's App Transport Security (which enforces encrypted communications). The app developers recognized the issue and remediated it in their next release. Kryptowire's analysis also identified certain apps that requested permission to write to the external storage of the device. The external storage on an Android device is insecure; any app on the device has read access to that data. Developers for these apps confirmed that the permission was no longer needed and removed it from updated app versions.

The information that developers provided on items that required their explanation also proved useful to APCO and Kryptowire evaluators. In many cases, it provided insight into their development process and methodology, enabling the team to better understand the security precautions being taken when accessing critical device functionality. For instance, one Android app was flagged for using the device's external storage. However, in their response, the developer noted that all information written to external storage was encrypted using a secure algorithm with a key derived from the user's password. This provided assurance that while the app performed what could potentially be an insecure operation, proper precautions had been taken by the developer to ensure the data remains secure.

4 Developer Feedback

After completing the remediation process, APCO sent a brief questionnaire (Appendix B) to the app developers who participated throughout the pilot. The goal of the questionnaire was to learn about developer backgrounds and obtain their perspective on the testing process to inform future app testing efforts. The questionnaire consisted of 19 multiple choice and short answer questions that covered the developer team's background, app testing experience and overall pilot process experience. The number and format of questions were limited to reduce the burden on developers. Nine of 10[8, 9] questionnaires were completed and returned.

4.1 Developer Backgrounds

The initial questions asked about each developer's organization and the resources the developer devoted to the pilot testing process. In terms of developer skillsets, all developers reported they build apps for the iOS platform; 89 percent build Android apps; and 33 percent build apps for Windows PC. The number of apps supported ranged from one to eight. The number of developers each organization committed to working on the app testing process ranged from one to five.

[8] Developers for the following apps completed the questionnaire: Active911, DForce, GeoSafe, Hiplink, STING, NowForce, PulsePoint, Rave Mobile Safety, and WatchTower.

[9] The outstanding questionnaire (1 of 10) was sent to the only developer whose app testing did not reveal any critical issues or issues requiring explanation. This developer did not have feedback that was informed by participation in the remediation process.

4.2 Developer Feedback

Developers were generally positive about the test reports and most said no tests or feedback should be removed. Eighty-nine percent of respondents reported the pilot test results were easy to interpret. Regarding the "red flag/remediation required" designations, developers highlighted a few instances of false-positives in the reports and disagreement with whether the detected app behavior constituted a critical security issue. No developers reported disagreement with the "orange flag/explanation required" criteria.

Most respondents reported that their organizations considered the app testing results and remediation process to be helpful and/or valuable, with some asking for even more rigorous testing. Most respondents (90 percent) reported that the app testing process placed a low burden on their workforce. The range of time spent on remediation was from zero to eight labor hours, with most developers spending approximately one hour on remediation. Following are examples of the changes developers made to apps because of the remediation process:

- Removed old and unused code
- Enabled application transport security
- Solidified code permissions in iOS
- Changed some permanent memory to volatile memory used in the Android app
- Removed unused contact framework
- Removed unnecessary permissions to allow app to write to external storage

Most of the developer respondents reported that their organizations considered the app testing results and remediation process valuable, with some asking for even more rigorous testing.

The testing platform's methodology was an important feature. Several developers reported that protecting intellectual property is a critical issue for their participation in app testing. Because the Kryptowire platform only requires a link to download an app such as the app's listing in a public app store participation in the testing process did not significantly increase the risk of intellectual property exposure.[10]

Most pilot testing participants reported app security testing is very important to their organizations. Most also reported they would consider paying for a security testing service of this nature in the future, particularly if it would lead to a certification that provided access to new customers or a marketing opportunity.

4.3 Developer Suggestions

The overall remediation process was deemed "satisfactory" by 56 percent of respondents and an equal amount (22 percent each) said it was "excellent" or "needs improvement" (see Figure 7).

[10] Kryptowire only requires binary code for mobile app testing and does not need access to source code. This alleviates concerns about protecting the intellectual property of mobile app developers.

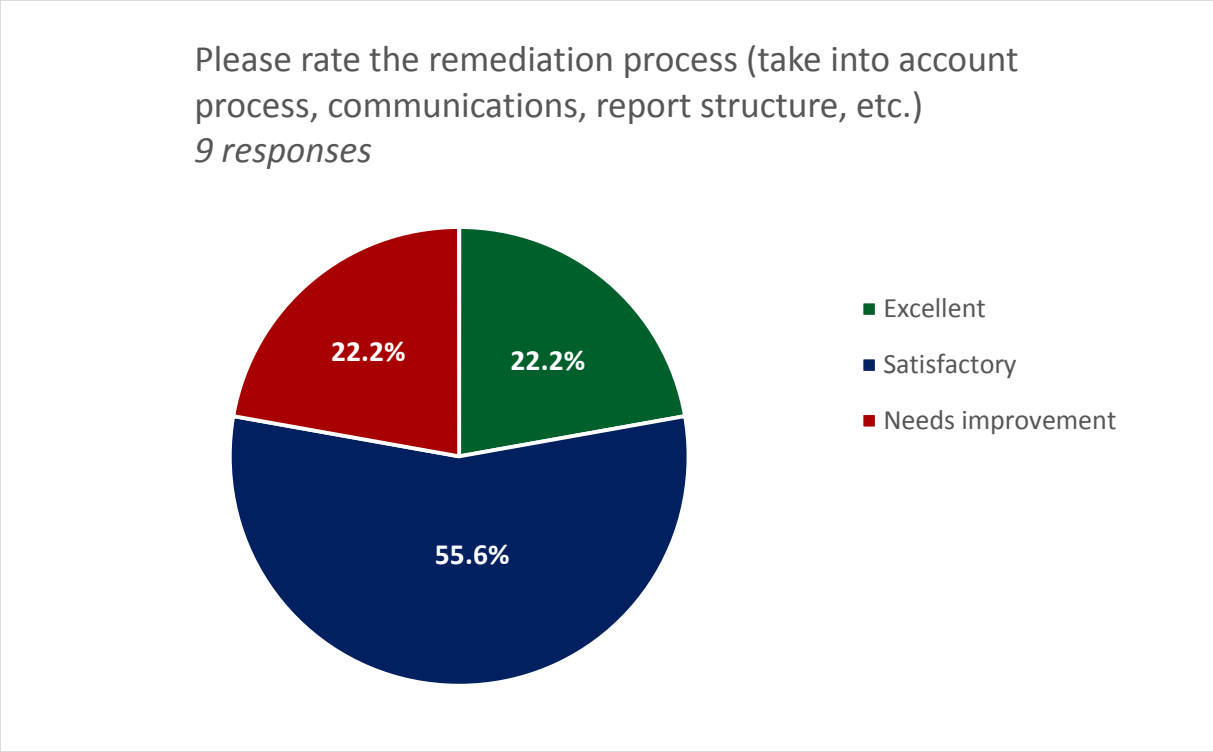

Figure 7. Developer Feedback on the Evaluation Process

While most developers reported the testing process made sense and integrated well into their organization's workflow, they did suggest several areas for improvement, including quicker turnaround times, more detailed information about how testing works and methods to improve communications throughout testing. For example, a testing portal could serve as a centralized resource for information about a testing program to ensure that consistent information is available to all developer personnel as well as provide real-time updates on testing or remediation progress. Some developers suggested including additional test components such as penetration testing and review of the server code.

5 Lessons Learned

This section explores lessons learned regarding the app testing model used in the pilot and its perceived value.

5.1 Pilot Model Effectiveness

The pilot demonstrated that an automated mobile app vetting capability could be integrated into a resource like AppComm to provide public safety app users assurance that apps have met certain security criteria. The interaction between app developers and app security evaluators should:

- Result in remediation of vulnerabilities identified by the security evaluator
- Illustrate software quality concerns/issues in app source code that—once addressed—improved the overall quality of the app
- Demonstrate how the analysis process provides a mechanism for developers to justify app behavior that may be deemed risky

- Improve state-of-the-art in app vetting by providing valuable feedback to the security experts concerning false-positives

5.2 The Value of Application Vetting from the Developer's Perspective

A key takeaway was developers must be sufficiently incentivized to participate in a model such as the one used for this pilot. While the model's cost burden is relatively low, developers must perceive a benefit to justify the time and potential fees of app vetting. Because the pilot did not result in a formal badge of approval, a lack of perceived value may have been the cause for the high developer dropout rate observed during the pilot.

5.3 The State of Public Safety Mobile Applications

Based on the number of security concerns and vulnerabilities identified in the limited test set considered by the pilot, mobile app security should be of concern to the public safety community. Further, the high attrition rate of developers after the testing phase was completed left unaddressed nearly half of the red and orange flag concerns discovered in the test set, and the developers who completed remediation could at any time introduce an updated version that has not been evaluated. This finding demonstrates the need for a formal, ongoing app evaluation process with appropriate incentives for developer participation.

5.4 Technical Lessons Learned

5.4.1 Additional Context for Findings Would Expedite the Remediation Process

For developers to quickly and accurately address a finding in their code, it was imperative that the test report provide as much context as possible. For a select few of the analysis findings, Kryptowire could have provided additional context that would have made it easier for the developers to track down the issue. This context could have included items such as lines in code where the finding was discovered, the literal constant that was declared and triggered the finding, or the specific web server that was contacted and raised a finding. Providing more information in the test report would make the remediation process more efficient.

5.4.2 Static Findings Are Not Reliable

In typical security analysis scenarios where a user is vetting a third-party app using Kryptowire's analysis suite, Kryptowire reports all static findings because they can pose potential security risks down the line for the app, even if they are not invoked in the app's current iteration. It is important to ensure that the process includes vetting an app after each new version release which would greatly minimize the risks posed by these static findings. By placing less emphasis on or removing these findings, Kryptowire also can eliminate some confusion that came from findings based on a third-party library's code.

5.4.3 Establishing Direct Communication with Developers Was Beneficial

In two cases, Kryptowire set up separate calls with the app developer to provide more information about a finding, enabling the developers to track it down. In both cases, the issue was resolved expeditiously and the pilot process moved forward. These communications proved to be extremely valuable to both parties. It also gave Kryptowire better insight into how

developers used its findings and how it could better tailor its reported findings to improve the developer's workflow.

5.4.4 A Dedicated Web Portal Could Improve the Process

All communication for the pilot, aside from the two direct calls with developers, was performed via email between the involved parties. While this approach worked for this small-scale pilot, it would be beneficial for a larger scale implementation of this effort to incorporate a central web portal to manage the entire process of app vetting, a step that was suggested by some pilot participants. This portal would allow all parties involved in the process to have a central location from which to retrieve and share information. It also would provide developers a dashboard to easily view the status of their app in the vetting process and inform them of needed information/response to facilitate completing the process.

6 Conclusion and Next Steps

Apps that are used for public safety services must be reliable and secure. Many app users assume the developer has taken the necessary steps to secure the app and the user's data and/or the app store tested the app before public release. From the pilot, the partners concluded that:

- Mobile apps used by first responders and members of the public for emergency response or other public safety purposes are vulnerable
- App security evaluations can be accomplished using semi-automated testing based on established criteria, combined with human analysis to make a risk-based assessment
- Continuous app security evaluations are necessary any time a mobile app is updated or a new version is submitted
- Developers are willing to pay for app evaluations if the right incentives are in place
- Education for the first responder community is needed to raise awareness of the state of mobile app security and increase demand for app security evaluation

A foundational question of the pilot was whether there is a financial model to support public safety app evaluations. The pilot's findings provide preliminary evidence for this financial model to be true. Developers recognize sufficient value in an app evaluation process to support a model in which developers pay for a subscription to a public safety app certification program. Not only did this finding validate the importance of the pilot, it suggests that expanding and refining a testing program for the broader public safety app ecosystem is feasible and desirable. Engagement with the app developer community for public safety apps is necessary to encourage and raise awareness of the need to build security in during the development process and to test the security of the apps prior to releasing them to an app store and the public. DHS S&T is investing in mobile app security R&D to integrate security into mobile app development platforms.[11] The result of this research will enable developers using the platform to improve the security of their mobile apps.

[11] DHS S&T Awards $8.6 Million for Five Mobile Application Security R&D Projects, https://www.dhs.gov/science-and-technology/news/2017/09/06/news-release-dhs-st-awards-86-million-5-rd-projects

The pilot also generated several lessons learned about the security criteria, testing platform and workflow. Notably, a knowledgeable mobile app security evaluator is essential to effective remediation and confidence in the evaluations. A mobile app analysis tool, although automated, should still require a human in the loop to make a risk-based assessment and decision. It is expected that introducing additional variables such as the preferences, policies and laws for individuals or sponsoring public safety agencies will increase the nuances of app evaluations and the need for human judgment as part of the process.

The next step is to carry the lessons learned from the app testing pilot into consultation with public safety stakeholders, including state and local public safety agencies and entities such as the First Responder Network Authority (FirstNet) and SAFECOM. The preferences, policies and laws affecting public safety organization use of mobile apps will dictate the ultimate form of a sustainable app evaluation process. Ongoing consultation and an awareness program for public safety stakeholders at the local, tribal, state and federal levels is essential for the establishment and continued success of an app evaluation process supporting the public safety mission.

Acknowledgements

The APCO app testing pilot would not have been successful without the dedicated efforts of organizations and individuals committed to ensuring that public safety apps perform as they should, are reliable and include appropriate protections to secure app data and more importantly the vital services provided by the public safety community.

- APCO
  - Mark Reddish, Senior Counsel & Manager of Government Relations
  - Megan Bixler, Technical Program Manager
  - Lauren Corcoran, former Government Relations Associate
- DHS S&T CSD
  - Vincent Sritapan, Mobile Security R&D Program Manager
  - John Drake, Mobile Security R&D Science Engineering Technical Assistance (SETA) Support
- DHS S&T FRG
  - John Merrill, Director, FRG Office for Interoperability and Compatibility
- Kryptowire
  - Tom Karygiannis, Vice President of Product
  - Brian Schulte, Senior Mobile Software Engineer
- NIST
  - Michael Ogata, Public Safety Communications Research Cybersecurity Project Co-Lead

Appendix A Analysis Items Tested

Item	Category	Platform(s)	Level
Uses external storage	Device Access	Android	Orange
Obtains unique ID of device	Device Access	iOS & Android	Orange
Records audio	Device Access	iOS & Android	Orange
Accesses device camera/photos	Device Access	iOS & Android	Orange
Accesses device location	Device Access	iOS & Android	Orange
Accesses SIM serial number	Device Access	Android	Orange
Sensitive information exposure[12]	Privacy	iOS & Android	Red
SMS/MMS interaction	Privacy	iOS & Android	Orange
Native email client interaction	Privacy	iOS	Orange
Accesses device's calendar	Privacy	iOS & Android	Orange
Accesses device's contacts	Privacy	iOS & Android	Orange
Integration with ad network	Privacy	iOS & Android	Orange
Hard-coded credentials	Security	iOS & Android	Red
Accepts all SSL certificates	Security	Android	Red
Malware	Security	iOS & Android	Red
Privilege escalation	Security	Android	Red
Admin privileges requested	Security	Android	Red
Disabled iOS app TLS	Security	iOS	Red

[12] Sensitive information exposure includes sharing of user or device information over the network.

Appendix B APCO Mobile App Vetting Pilot Questionnaire

The following questions were part of a questionnaire that was provided to the app developers who participated in the APCO Mobile App Vetting Pilot:

Your team's honesty in answering the following questions is greatly appreciated. All answers will be kept confidential and are intended solely for improving the processes of future app vetting requirements for AppComm.

DEVELOPER TEAM BACKGROUND

1. What platforms does your team currently target for mobile app development (iOS, Android, etc.)?
2. How many applications does your organization currently support, either in public app stores or private enterprise app stores?
3. How many individuals in your organization were required to take part in the app vetting process?
4. What other, if any, third-party app testing or software assurance processes does your organization use?
5. How many times do you update your organization's applications a year?
6. How much money would your organization expect to pay for a service like the one provided in the pilot?

OPERATIONAL APP VETTING EXPERIENCE

1. How many times was remediation required of your application?
2. What was the longest period your organization waited for an action associated the app vetting process? For what action was your organization waiting (app vetting results, clarification to an inquiry, etc.)?
3. If remediation was required of your application, how many hours were spent applying the required changes?
4. If remediation was required to your application, did your organization agree with or disagree with the validity of the requested changes?
5. Was there a remediation that was not implementable by your team? If so, what was it? For what reason did your team determine the remediation was untenable?

OVERALL PILOT PROCESS FEEDBACK

1. Did the general flow of the app vetting process make sense and integrate into your organization's workflow?
2. Did the reporting provided by the app vetting process meet the needs of your organization's developers?
3. Does your organization perceive processes such as those found in this pilot to be the deterrent for future AppComm participation?
4. Where there any reservations your organization had with respect to the app vetting process, with special regard to issues concerning the exposure of intellectual property?
5. How much burden did the app vetting process place on your organization?

   Low Moderate Heavy
6. What importance does your organization place on app vetting evaluations like those present in the pilot?

   Low Moderate Heavy

Appendix C Text Equivalent for Figures 5 & 6

FIGURE 5. EXAMPLE FINDINGS: EXPOSES SENSATIVE INFORMATION

The graphic is an example of an “Exposes sensitive information” finding that requires developer remediation. Under the “Exposes sensitive information” title, the explanation states: “We scan all network traffic generated during Dynamic Analysis for any sensitive information. This includes both plain text HTTP and encrypted HTTPS.”

The issue is then described as: “The application exposes personally identifiable information (PII) in a communication to an external location. This causes a high risk to a user's privacy. It should be evaluated if the data being exposed is being sent to authorized third parties and if the app can operate as normal with the removal of the identified data exposure.”

Under this description, the example shows fields titled, “Type”, “Data Value”, and “Data Sent To” indicating where in the code the issue was identified. Under the “Type” field the User_Identifier has been noted. Under the “Data Value” field the following code is provided {“channel”:background”:false,”tags”:,”opt_in”:false,”set_tags”:true,”device_type”:”ios”},”identify_hints”:{“user_id”:(redacted sensitive information)”.”device_id”: (redacted sensitive information)”}}. Under the “Data Sent To” field notes https:// (redacted sensitive information).

FIGURE 6. EXAMPLE FINDINGS: USING HARD-CODED CREDENTIALS

The graphic is an example of a “Uses Hard-Coded Credentials” finding that requires developer action. Under the “Uses hard-coded credentials for secure operations” title, the explanation states: “Scans were performed on the application's byte code and any packaged SDKs to search for hard-coded credentials used in cryptographic functions. These codes are declared as constant values within the application's code.”

The issue is then described as: “The application contains a hard-coded credential to perform secure operations such as encryption or web authentication. This allows anyone who has access to the application to retrieve the credentials and perform the same operations compromising security and privacy. Proper functionality provided by the native platform should be used to properly create and store credentials. More information can be found at the CWE-321: Use of Hard-coded Cryptographic Key website as well as the Android Keystore System website and iOS Cryptographic Services Guide.”

Under this description, the example shows fields titled, “Key”, “File Key Defined In”, and “File Key Used In” indicating where in the code the issue was identified. Under the “Key” field states 8 (redacted sensitive information). Under the “File Key Defined In” field states com (redacted sensitive information) and under field titled “File Key Used In” states com (redacted sensitive information).