Mitigating Bring Your Own Device Risks by Static Analysis Empowered by Knowledge Graphs from Open Web Application Security Project

by
Suzanna Schmeelk

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

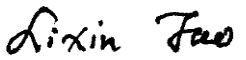
at

Seidenberg School of Computer Science and Information Systems

Pace University

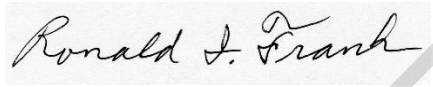
May 2020

We hereby certify that this dissertation, submitted by Suzanna Schmeelk, satisfies the dissertation requirements for the degree of *Doctor of Professional Studies in Computing* and has been approved.



_____-__4/27/2020_____
Dr. Lixin Tao Date
Chairperson of Dissertation Committee

_____-__4/27/2020_____
Dr. Charles Tappert Date
Dissertation Committee Member



_____-__4/27/2020_____
Dr. Ron Frank Date
Dissertation Committee Member

School of Computer Science and Information Systems
Pace University 2020

Abstract

Mitigating Bring Your Own Device Risks by Static Analysis Empowered by Knowledge Graphs from Open Web Application Security Project

by
Suzanna Schmeelk

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

May 2020

Many organizations, to save costs, are moving to Bring Your Own Mobile Device (BYOD). In these scenarios, organizations have a Mobile Device Management (MDM) system in place. MDM solutions lower cyber security risks by providing remote wipe procedures, geo-location fencing, among others. However, MDM systems are not yet focused on application-level security with a fine-level of granularity. MDM systems currently may not monitor for data loss prevention (DLP), or even standard web-application vulnerabilities that a penetration tester would examine. In addition, organizations around the world are adopting applications built by third-parties at an unprecedented rate.

This research contributes an examination of mobile application security through the construction of a knowledge graph for the OWASP Top 10 2014 and 2016 threats. The knowledge graph contribution links threats from the different years to show changes in time and to help determine security changes over time. Currently, only the National Institute of Standards and Technology (NIST) Bug Framework has built any such graph representation to inform analysis. This high-level graphic shows potential vulnerabilities such as the insecure storage of sensitive data and insufficient cryptography, which depending on how the code is utilized can occur heavy fines for the mismanagement of sensitive information.

This research then contributes how specific mobile device source code, specifically Android in this research, can useful to inform static analysis. In this research we focus on source code analysis; however, the knowledge-graph can be connected to byte code or entirely other mobile device application languages such as Swift, JavaScript, and C/C++.

We then make a contribution to analyze over 200 healthcare Android applications source code from GitHub to learn what, if any, security concerns are being deployed to improve

secure source code development for sensitive data. Some of the applications analyzed collect highly sensitive information such a body weight, body signals (e.g. blood pressure, temperature), obstetrics/gynecology measurements, mental health measurements, among others. Specifically, in this research, we analyzed applications for components of the constructed-knowledge graphs, specifically, components of the confidentiality and integrity of their sensitive information.

As our world moves more-and-more to the edges with the Internet of Things and mobile application development, security concerns and the storage and transmission of sensitive data is becoming a serious concern. In fact, recent regulatory changes are occurring at unprecedented rates with adoptions of new laws at local, national and international levels. Having a clearer picture of security on our mobile devices is now an industry necessity.

Acknowledgements

This dissertation has been a crescendo on my work for cybersecurity software assurance of mobile applications. This work began with deep mobile application software assurance research during my fulltime career work with a former Bell Laboratories subsidiary, later involved mobile application cybersecurity analysis for top New York City hospital(s) and most recently with my industry (pro bono) work as a fulltime Assistant Professor of Cybersecurity. Throughout my fulltime work with these organizations, I observed a lack of cybersecurity completeness in research and practice, which was deeply disturbing. The faculty at Pace University have helped me address many of these observed cybersecurity gaps for the benefit of the research, industry, and general public communities-at-large.

I would like to first thank Dr. Lixin Tao, my adviser, for all his time, support and advice during the years working on this dissertation thesis. Dr. Tao, Pace University Seidenberg School of CSIS Professor and IEEE Senior Member, introduced me to the domain of the Semantic Web Knowledge Graphs, which was a perfect cybersecurity fit for adapting and building more complete representations and adaptable knowledge sets for cybersecurity software assurance. Dr. Tao's contributions to my own personal growth have been consistently domain-enhancing, consistently encouraging, and will never be forgotten. Thank you so much, Dr. Tao!

Thank you to my dissertation committee, especially Dr. Charles Tappert and Dr. Ron Frank, who have been so generous with their technical experiences and teachings. Dr. Tappert, Pace University Seidenberg School of CSIS Clinical Professor, has contributed to every class we took in Pace University. He regularly facilitated discussions based on his outstanding and long career at IBM and earlier professional experience at West Point Military Academy. His contributions to his students' learning are endless and will never be forgotten. In fact, I recall an episode at an International Machine Learning conference, where Dr. Tappert was generously helping all students from Pace University prepare and present their research. He came to all their sessions and made sure session track chairs were on schedule. Thank you, Dr. Tappert!

Dr. Ron Frank, Pace University Seidenberg School of CSIS Associate Professor, has been so generous with his time and sharing his industry expertise. Dr. Frank spent his first career working in industry at IBM before coming to Pace University. He has phenomenal experience and an outstanding repertoire. His stories are extremely authentic and geared to enlightening all his students to prepare them for the next steps in their careers. I recall an episode where Dr. Frank taught all his students about maintaining research memoirs in notebooks which could endure in intellectual property court disputes. This is one example of many of his extremely forward-thinking teaching moments to prepare students to think in advance about preserving their own intellectual property. Thank you, Dr. Frank!

In addition, all the faculty at Pace University have been extremely supportive--especially of their advice, time and industry expertise. I would like to thank our DPS classroom faculty: Dr. Tilak Agerwala, Paul Dantzig, Dr. Sung-Hyuk Cha, Dr. Li-Chiou Chen, Dr. Yegin Genc, Rinaldo DiGiorgio, and Istvan Barabasi. In addition, the Pace faculty and staff, especially the Dean Dr. Jonathan Hill, Dr. Susan Feather-Gannon, Dr. Christelle Scharff, Dr. Miguel Mosteiro, Dr. Thomas Schmidt, Michelle Lang, Jose Cueto, and Jill Olimpieri have also been generous of their time, advice, and expertise. Finally, a big *thank you* to Fred Grossman, Charles Tappert, Joe Bergin, Susan M. Merritt, Allen Stix, Judith E. Sullivan, and David A. Sachs for working so hard to imagine and establish a research doctoral program for fulltime working professionals in the first place.

Thank you to the other DPS Students, who are all fulltime working professionals and/or industry veterans. First, I would like to thank my NYPD Cyber Detective industry veteran colleague and peer, Denise Dragos, for our many research undertakings together. Second, I would like to thank my amazing DPS Team 2 Cohort: Lisa R. Ellrodt, Tonya L. Fields, Ion C. Freeman, and Ashley J. Haigler (including the founding group members Lynne Larkin and Ronald Williams.) Thank you also to our entire DPS 2020 Cohort for sharing their wealth of industry knowledge on our long Friday nights together and all day Saturdays together. Our DPS 2020 Cohort consists of: Miguel Zhindon, Philip Ricciardi, Joseph Porter, Edmund Miller, Paxton Louis, David Lasecki, Rajesh Khemraj, and Binu Jacob. Lastly, many thanks to Dr. Ning Jiang for his work with me on Pace Protegee.

Finally, thank you to my family who have been extremely supportive especially when I have missed family events due to my studies.

As the proverb says, "It takes a village to raise a child." We have learned so much about the computing industry, collaborated with our (invited) industry leading peers, and expanded our own computing industry experience. We have spent many days together both through in-person and virtual communications and conferences. Overall, the learning experience has been extremely rewarding, enlightening, positive, and entirely unforgettable.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Many organizations, to save costs, are moving to Bring Your Own Mobile Device (BYOD). In these scenarios, many organizations have a Mobile Device Management (MDM) system in place. MDM solutions lower cyber security risks by providing remote wipe procedures, geo-location fencing, among others. However, MDM systems are not yet focused on application-level security with a fine-level of granularity. MDM systems currently may not monitor for data loss prevention (DLP), or typically web-application vulnerabilities that a penetration tester would examine.

In addition, the BYOD paradigm supports users installing personal, and professional applications on the same work device. In such scenarios, penetration tests typically perform tests on applications specific to the organization; however, they have no way of knowing in advance which applications will be run on the device in addition to the work-related application. Penetration tests of mobile devices traditionally may include both static and dynamic application tests. Since mobile computing is relatively new, many penetration testers and application developers have no way of knowing how best to analyze the applications. Currently, penetration tests are ad hoc and can differ between organizations. This dissertation research is significant as it introduces knowledge graphs

as a methodology to examines how mobile security is covers by static analysis and identifies issues which have not been examined in academics. The OWASP knowledge graphs can be used standalone, by penetration testers, by developers, by quality assurance testing teams, by security community researchers (e.g. in industry and by government standardizing organizations such as the National Institute of Standards and Technology (NIST)), and by static analysis tool producers to further enhance their tools and coverage representations. In this dissertation, we show how the knowledge graphs can be used during static analysis to improve the overall mobile security code during development. Over 200 Android Healthcare applications were reviewed from GitHub to improve their mobile cybersecurity risks in accordance with OWASP.

1.1 Opportunities and Challenges in Mobile Security Analyses

Analysis mobile applications for security issues is extremely relevant to many organizations; however, it comes with challenges. Specifically, typically tools developed to test for security concerns have four types of findings: true positive, false positive, true negative and false negative. A true positive security issue detection means that a tool properly identified a security concern. A false positive security issue identified by the tool has been improperly characterized by the tool as a security issue when in fact it is not a security issue. The lower the false positive rate the more sound a tool is considered. A true negative is when a tool properly labels an issue as a non-issue. A false negative, one of the most serious cases, is when a tool improperly identifies a security issue as a non-issue. The lower the false negatives the more complete a tool is considered to be.

Currently very little, if any research is properly working to identify false negative statistics for security analysis tools. Generally standardizing bodies such as NIST work to help develop benchmarks to help properly classify and analyze security analysis tools.

Two popular security analysis tools are built around static analysis or dynamic analysis or the hybrid of the two. Static analysis examines source, intermediate, or machine code to identify issues. Traditional static analysis applications are within compiler tools. Dynamic analysis tools examine code during execution. There are many methodologies of dynamic analysis such as code coverage, code behavior analysis and sandboxes. Finally, research tools can combine methodologies such as in malware analysis to statically remove timers or other obfuscating code before actually executing the code to examine the behavior.

As discussed in Chapter 2 Review of Literature the research community, especially in mobile static analysis, is not systematically examining software issues. As explained in the chapter literature, systematic analysis is transpiring over certain well-known security issues leaving others entirely unresearched increasing security risks.

1.2 Problem Statement

Organizations around the world are adopting mobile technology and applications built by third parties at an unprecedented rate. This research examines software assurance methodologies specifically through source code analysis to improve mobile application cybersecurity risks in accordance with OWASP best practices. Security analysis coverage of static analysis in Android can be further implanted for malware prevention, mitigation,

and detection. This research examines current static analysis research of Android mobile application to discover trends in analysis. Recent academic publications present the coverage and distribution of current application of static analysis on Android detection, mitigation, and prevention challenges. This research then develops a new knowledge-graph for Android security based on OWASP best practices as contribution. This research links the knowledge graphs with Android source code as empirical evidence that static analysis techniques can be a useful methodology for threat detection, mitigation, and prevention. The dissertation then presents compiles new vulnerabilities useful for detection by static analysis methodologies and derives conclusions about the research trend, which subareas are suitable for static analysis and needs more research. Finally, we then make a contribution to analyze over 200 healthcare Android applications source code from GitHub to learn what, if any, security concerns are being deployed to improve secure source code development for sensitive data. Some of the applications analyzed collect highly sensitive information such a body weight, body signals (e.g. blood pressure, temperature), obstetrics/gynecology measurements, mental health measurements, among others. Specifically, in this research, we analyzed applications for components of the constructed-knowledge graphs, specifically, components of the confidentiality and integrity of their sensitive information.

Methodology Solution Statement

Neither a mobile cybersecurity knowledge graph nor its respective cybersecurity static analysis knowledge graph currently exists. As such, there is considerable effort not only

in building a knowledge graph but also in graph justification. As a result, we have started with the industry standard OWASP Top 10 Mobile Threats to construct initial components of a security knowledge graph for one particular mobile platform, Android. The two most current OWASP Top 10 Mobile Threats published on their main page are from 2014 and 2016. The OWASP frameworks are highly utilized by industry professionals to analyze their code for vulnerabilities and to learn how to solve penetration test remediation problems. We explore both years of OWASP Tope 10 Mobile Threats in more detail to see the differences and similarities between the years to work towards building a theoretically complete knowledge-graph to capture all known threats at a point in time.

This research examines mobile application security from the perspective of static analysis to develop new security-focused software analysis methodologies by making three contributions. First, this research contributes knowledge graphs for security threats in mobile applications. These knowledge graphs can be employed by industry to detect software weaknesses during software analysis (e.g. static, dynamic, penetration testing, legal, etc.). Second, the knowledge graphs are connected directly with mobile Android software code to provide real-world empirical evidence to the context of the analysis. Third, the research contributes what static analysis techniques could be employed to capture different categories of mobile application security threats that have previously not been researched or identified in research. Specifically, the source code for 200+ Android healthcare mobile applications are statically examined to improve their mobile cybersecurity risks with OWASP. The field at large can apply and extend any of the contributions to further guide their mobile application cybersecurity analysis.

Specifically, the solution methodology are as follows:

- Review the literature on current cybersecurity static analysis research trends
- Develop knowledge-graphs for Android mobile security based on OWASP
- Evaluate the knowledge-graph with Android source code to improve static analysis techniques capable of detecting cybersecurity risks
- Analyze the mobile source code for 200+ open source Android healthcare applications hosted on GitHub to improve their mobile cybersecurity risks with OWASP

1.3 Expected Contributions

Little systematic research exists for mobile cybersecurity software assurance. As such, industry and academic experts find it difficult to compare and contrast assurance analyses. In fact, penetration tests at one organization may not be equivalent to a penetration test at another organization. Our expected contribution is to improve systematic mobile software security research and to improve their mobile cybersecurity risks. Specifically, the contributions are as follows:

- Develop knowledge-graphs for Android mobile application security based on both of the current OWASP's top ten mobile threats to identify similarities, identify differences and discover longitudinal changes to the OWASP threats for more complete analyses.
- Evaluate knowledge-graph with Android source code connections to elicit types of static analysis techniques capable of detecting

- Identify un-researched mobile software vulnerabilities detectable via static analysis.
- Analyze the software assurances in 200+ open source Android healthcare applications hosted on GitHub to improve their cybersecurity risks with respect to OWASP.

1.4 Approach Validation

As empirical evidence we will examine both the developed OWASP 2014 and 2016 knowledge-graphs through the lens of Android mobile source code by examining 200+ open source healthcare applications which store inherently sensitive information for two of the top ten threats. Future research can extend the developed OWASP knowledge-graphs as known security issues develop.

1.5 Research Outline

This dissertation research examines through literature, building knowledge graphs, and analyzing Android source code from GitHub, new classes of issues to be considered with static analysis.

Chapter 2 reviews the related literature on mobile cybersecurity software assurance with respect to static analysis. The chapter examines research use cases for cybersecurity ontologies. It then examines methodologies to systematically identify and communicate vulnerabilities. The next section explores current literature on mobile application security analyses. Finally, the chapter explores current knowledge graph techniques and literature.

Chapter 3 introduces the Android mobile application structure focusing on the client-side applications structure—the scope of this dissertation research. This section includes the Android Linux sandbox structure, accessing mobile resources from within an application and discusses additional client-side application features such as including external packages and native code. In addition, this section includes an overview discussion on mobile client-server architecture will be discussed to inform on client-side research scope.

Chapter 4 identifies current applications of static analysis to various mobile threats, and argues/describes, based on the knowledge graph dependencies among the threats and among the mobile application infrastructure components, what additional application of static analysis could be used to other related threats.

Chapter 5 examines detectable security management coding patterns to improve their mobile cybersecurity risks with the constructed OWASP knowledge-graph. The Android source code for 200+ Android healthcare mobile applications is statically analyzed in accordance with multiple OWASP Top 10 Threats to improve their risks. The applications store and handle extremely sensitive healthcare information which have higher impact if exposed through loss or theft.

The last chapter, Chapter 6, concludes the research with a summary of findings, limitations and future work directions.

Chapter 2

Review of Literature

The mobile security field contains disjoint security concerns and best practices, which originate from many geographic sources, many perspectives, and do not conform to a given standardized language. One way to unify disjoint topics is through building ontologies and knowledge graphs. Ontologies, or semantic models, have been successful in helping people to assemble their knowledge to understand "their world by forming an abstract description that hides certain details while illuminating others [1, p. 15]" Liyang Yu [2] defines an ontology as, "An ontology formally defines a common set of terms that are used to describe and represent a domain … An ontology defines the terms used to describe and represent an area of knowledge. [3, p. 151]" Three traditional ways, according to Allemang and Hendler [1, p. 15] that models assist with "understanding is through: (1) communication, (2) explain and make predictions, and (3) mediate multiple viewpoints. Ontologies traditionally are based on hierarchical relationships (e.g. "is a", taxonomy). Knowledge Graphs, however, support other relationships beyond simply hierarchical (e.g., *partOf*, *comprisedOf*, etc.). Figure 1 below shows an ontology reported by Kafali et al. [4,