Perspectives on Static Analysis of Mobile Apps (Invited Talk) *

Marco Autili[1], Ivano Malavolta[2], Alexander Perucci[1], Gian Luca Scoccia[2]
[1]University of L'Aquila, Department of Information Engineering, Computer Science and Mathematics (Italy)
[2]Gran Sasso Science Institute, L'Aquila (Italy)
marco.autili@univaq.it, alexander.perucci@graduate.univaq.it,
{ivano.malavolta | gianluca.scoccia}@gssi.infn.it

ABSTRACT

The use and development of mobile apps is growing at a tremendous rate in the last years. Even if this growth is making the mobile apps market very attractive for software developers, it is also continuously presenting new challenges. Indeed, mobile platforms are rapidly and continuously changing, with the addition of diverse capabilities like the support for new sensors, APIs, programming abstractions, etc. In this respect, a number of static analysis methods and techniques have been proposed in research as a powerful instrument for developing more qualitative mobile apps.In this invited talk we report on the results of a preliminary survey we conducted on *static analysis methods and techniques of mobile apps*.

Categories and Subject Descriptors

D.2 [**Software**]: Software Engineering

Keywords

Mobile applications, Static analysis, Systematic study

1. INTRODUCTION

Motivation – Programming languages and tools for developing mobile apps are platform-specific (e.g., Java code for Android apps, and Objective-C code for Apple iOS apps), and present many challenges that may hamper the success of a mobile app as a whole. Indeed, recently, an empirical study indicates a strong need of mobile app developers for better analysis and testing support, with a focus on important features like mobility, location services, sensors, as well as different gestures and inputs [3]. Under this perspective, *static program analysis* [4] can be an effective and viable instrument to predict and evaluate (precise or approximated) quantitative and qualitative properties related to the run-time behaviour of a mobile app without actually executing it. Static analysis of mobile apps can be a valuable instrument for both app developers an app store moderators (e.g., Google, Apple) because it helps in creating products with better quality in a world where a low-quality release can have devastating consequences [3]. In this respect, a number of static analysis approaches have been proposed in the literature, each of them with specific peculiarities, features, and capabilities.

Contribution – In this invited talk we report on the results of a preliminary survey we conducted on methods and techniques for static analysis which estimate specific properties and features of mobile apps. The *main outcomes* of this study are: (i) a reusable comparison framework for understanding, classifying, and comparing techniques for static analysis of mobile apps, and (ii) an initial overview of the current state of the art about existing methods and techniques for static analysis of mobile apps.

*This work is partially supported by the MIUR, prot. 2012E47TM2, Prin project IDEAS.

DeMobile'15, August 31, 2015, Bergamo, Italy
ACM. 978-1-4503-3815-8/15/08
http://dx.doi.org/10.1145/2804345.2804352

2. STUDY DESIGN

We designed this study by borrowing some principles from the well-known guidelines for performing systematic mapping studies in software engineering [5]. This kind of empirical strategy is extremely powerful in structuring a given research area (static analysis for mobile apps in our case) by collecting and analysing existing work in it, while minimizing as much as possible the chances of bias [5]. To the best of our knowledge, this study presents the first systematic investigation into static analysis for mobile apps.

Research questions. Goal of our study is to *identify and classify the characteristics and evaluation quality of methods and techniques for static analysis of mobile apps*. This goal can be refined into the following research questions:

RQ1. What are the existing methods and techniques for static analysis of mobile apps?

RQ2. What are the characteristics of existing methods and techniques for static analysis of mobile apps?

RQ3. What is the quality of the evaluation performed on existing methods and techniques for static analysis of mobile apps?

Studies search and selection. As the research area of static analysis for mobile apps is very recent (the concept of mobile app exists only since 2007) and given the preliminary nature of this study, we decided to focus exclusively on high-quality publications in top-level scientific venues in the software engineering area. Based on this, we performed a manual search on the top-level software engineering conferencesand international journals. The time span of our search is from January 2007 to March 2015, summing up to 6541 potentially relevant studies. Once identified the data sources, we considered all the selected studies and filtered them according to a set of well-defined inclusion and exclusion criteria. As recommended in the guidelines for performing systematic literature reviews [5], the selection criteria of this study have been decided upfront, so to reduce the likelihood of bias. In order to reduce bias, two researchers performed the studies selection independently. At the end of this stage we obtained the 9 primary studies.

Data extraction. The first goal of this stage is to create a comparison framework that fits well with the studies under analysis. We followed a systematic process called *keywording* [5] for defining the various parameters of our comparison framework. Once the comparison framework has been setup, we considered all primary studies and we populated the comparison framework with the extracted data. In order to mitigate the presence of biases, (i) two researchers extracted the data from all the primary studies independently, and (ii) we performed a sensitivity analysis to analyse whether the results are consistent independently from the researcher performing the analysis.

Data synthesis. We performed content analysis, mainly for categorizing and coding approaches under broad thematic categories. Then, we performed narrative synthesis for explaining in details and interpreting the findings coming from the content analysis.

3. RESULTS

In the following we report a selection of the main results of our survey and briefly discuss them. To allow easy replication and verification of our study, we make publicly available the replication package containing all the extracted data of the study[1].

Analysis Goal. Most of the primary studies focus on energy consumption. The second most common goal is to improve the security of mobile apps with techniques aimed at detecting malware and leaks of sensitive information. Other goals are (i) the detection of bugs that undermine performances, (ii) the reduction of memory consumption, and (iii) the detection of bugs.

Hybrid. In two cases the proposed static analyses are complemented with some kind of dynamic analysis. More specifically, in the first case the authors "*execute the instrumented code on the emulator and record the system calls invoked for each event trace* ", whereas in the other case the approach is complemented with a "*runtime measurement*".

Analysis Presteps. Three are the primary studies requiring preliminary steps before the static analysis. In the first one "*the Workload Generator is responsible for converting the user-level actions, for which the developer wants an estimation, to the path information used by the Analyzer and Source Code Annotator*"; in the second one "*preprocessing of application can be divided into three steps: (i) EFG [Event Flow Graph] extraction (ii) Event trace generation (iii) Extraction of system calls sequence for each event trace*"; finally, in the third one a preliminary run-time measurement is performed before the static analysis.

Abstraction Level. Static analyses may operate at different levels of abstraction: bytecode, source code, and intermediate model. Most of proposed approaches work on the bytecode or source code level of the app. Only one approach runs on an intermediate model of the app, which specifies the signature of malware families via a Datalog-based language. Interestingly, in two other case the bytecode of the app is used in conjunction with its source code in order to produce source code annotations for the developer.

Platform. The majority of the approaches are specific to the Android platform. A possible interpretation of this trend may be due to the open-source nature of the Android platform. Also, Android app binaries can be straightforwardly disassembled with off-the-shelf software libraries and their internal structure and contained static resources are easily analyzable in an automatic manner. Interestingly, the approaches presented in three cases are generic and applicable to a variety of platforms (e.g., Android, iOS, Windows, Blackberry), however their tool is implemented for the Android platform only.

[1] `http://cs.gssi.infn.it/demobile_2015`

4. RELATED WORK

In [2] a survey about static analysis and model checking approaches for searching patterns and vulnerabilities within a software system is proposed. Peculiarity of this research is the comparison of static analysis algorithms against mathematical logic languages for model checking. The authors of [6] conducted a survey about static analysis for identifying security issues and vulnerabilities in software systems in general (not specific to mobile apps). For each type of security vulnerability the authors present both relevant studies and the implementation details of the used static analysis algorithms. A systematic mapping study has been conducted in[1] for classifying and analysing approaches that combine different static and dynamic quality assurance technique. The study included a discussion about reported effects, characteristics, and constraints of the various existing techniques. In conclusion, even if there are studies about static analysis and some aspects of mobile apps, none of them is actually focussing on the analysis of mobile apps. Since the need of analysis of apps has been raised in some recent works (e.g., in [3]), our study falls exactly in this area of research by providing an initial overview of existing approaches for static analysis of mobile apps.

5. FUTURE WORK

As future work, we are working on the design and conduction of a full-fledged systematic mapping study about static analysis of mobile apps, with the chief aim, among the others, of performing a more systematic search and selection activity and a more thorough analysis of the obtained data. Also, according to the research gaps we will likely identify in our systematic map, we will focus on a specific research challenge in the area and we will propose methods and techniques for addressing it.

6. REFERENCES

[1] F. Elberzhager, J. Münch, and V. T. N. Nha. A systematic mapping study on the combination of static and dynamic quality assurance techniques. *Information and Software Technology*, 54(1):1–15, 2012.

[2] I. Garcıa-Ferreira, C. Laorden, I. Santos, and P. G. Bringas. A survey on static analysis and model checking. In *International Joint Conference SOCO'14-CISIS'14-ICEUTE'14: Bilbao, Spain, June 25th-27th, 2014, Proceedings*, volume 299, page 443. Springer, 2014.

[3] M. E. Joorabchi, A. Mesbah, and P. Kruchten. Real challenges in mobile app development. In *Empirical Software Engineering and Measurement, 2013*, pages 15–24, 2013.

[4] F. Nielson, H. R. Nielson, and C. Hankin. *Principles of program analysis*. Springer, 2015.

[5] K. Petersen, S. Vakkalanka, and L. Kuzniarz. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18, 2015.

[6] M. Pistoia, S. Chandra, S. J. Fink, and E. Yahav. A survey of static analysis methods for identifying security vulnerabilities in software systems. *IBM Systems Journal*, 46(2):265–288, 2007.