6-2016

Poster: Android whole-system control flow analysis for accurate application behavior modeling

Huu Hoang NGUYEN
Singapore Management University, hhnguyen@smu.edu.sg

DOI: https://doi.org/10.1145/2938559.2948874

Poster: Android Whole-System Control Flow Analysis for Accurate Application Behavior Modeling

NGUYEN Huu Hoang
School of Information Systems, Singapore Management University
hhnguyen@smu.edu.sg

ABSTRACT
Android, the modern operating system for smartphones, together with its millions of apps, has become an important part of human life. There are many challenges to analyzing them. It is important to model the mobile systems in order to analyze the behaviors of apps accurately. These apps are built on top of interactions with Android systems. We aim to automatically build abstract models of the mobile systems and thus automate the analysis of mobile applications and detect potential issues (e.g., leaking private data, causing unexpected crashes, etc.). The expected results will be the accuracy models of actual various versions of Android system and apps for top apps selected from Google Play Store.

CCS Concepts
• Software and its engineering~Automated static analysis

Keywords
Android system; Android application; whole-system analysis; control flow; data flow; static analysis; modeling

1. INTRODUCTION
Android users were able to choose millions of apps. The analysis of these application becomes necessary. There are some approaches were released such as Flow Droid [1], GATOR [4], IccTA [2]. However, these approaches just analyze the control/data flow inside Android apps. Different from many studies on mobile application analysis built on top of manually constructed and assumed to be correct behavior models of the system (e.g., lifecycle APIs for Android apps, asynchronous task APIs, etc.).

We aim to automatically build the models for the Android systems and identify its interaction points with any mobile application. Then, we carry out mobile application analysis based on combined models of the system and the app.

With automatically built models, we can deal with various versions of the systems. We don't have to assume the availability and the correctness of the system models. In addition, we aim to enable a grey-box directed testing of apps that can reveal application behaviors more comprehensively to detect app crashes and private data leaks more accurately.

2. APPROACH
Our approach focuses on whole-system modeling of Android system and application code. We follow the steps as

(1) We model Android system behaviors involving system bootstrapping, application start-up, event handling, to automatically construct whole-system control-flow and call graphs;

MobiSys'16 Companion, June 25-30, 2016, Singapore, Singapore
ACM 978-1-4503-4416-6/16/06.
http://dx.doi.org/10.1145/2938559.2948874

(2) We model interactions between system and apps through GUI elements, callbacks, and system calls, from both system and application bytecode;
(3) We design algorithms for analyzing the whole-system graphical representations of the system and app code to identify points of interest more accurately, such as private data leaks.

Our approach is based-on Soot framework with call graph construction systems such as Spark and the incremental BDD propagation algorithm [3].

3. EVALUATION
We evaluate our approach on recent major different versions of Android systems and top apps from Google Play Store. First, we focus on the comprehensiveness and accuracy of the abstract models for Android systems, such as various event handlers and thread handlers. Second, we measure the coverage and precision of the models for apps when analyzed together with system models. Third, we measure test coverages and issue exposition rates for crashes and private data leaks in apps, and compare analysis results against other tools, such as Flow Droid [1].

4. RELATED WORK
Static analysis the accurate behavior is essential for modeling the control flow of Android applications. There are some approaches such as GATOR [4], a control-flow analysis of user-event-driven callbacks, or IccTA [2], detecting inter-component privacy leaks in Android apps.

5. CONCLUSIONS
We build a prototype system that can construct and visualize the interaction model between the system and apps. Our system models whole-system control flow for accurate app behaviors.

6. ACKNOWLEDGMENTS
This research is supported by the NRF, Prime Minister's Office, Singapore under its IDM Futures Funding Initiative.

7. REFERENCES
[1] Arzt, S., Rasthofer, S., Fritz, C., Bodden, E., Bartel, A., Klein, J., ... & McDaniel, P. (2014, June). Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps. In *ACM SIGPLAN Notices* (Vol. 49, No. 6, pp. 259-269).

[2] Li, L., Bartel, A., Bissyandé, T. F., Klein, J., Le Traon, Y., Arzt, S., ... & McDaniel, P. (2015, May). IccTA: Detecting inter-component privacy leaks in Android apps. In *Proceedings of the 37th International Conference on Software Engineering-Volume 1* (pp. 280-291). IEEE Press.

[3] Vallée-Rai, R., Co, P., Gagnon, E., Hendren, L., Lam, P., & Sundaresan, V. (1999, November). Soot-a Java bytecode optimization framework. In *Proceedings of the 1999 conference of the Centre for Advanced Studies on Collaborative research* (p. 13). IBM Press.

[4] Yang, S., Yan, D., Wu, H., Wang, Y., & Rountev, A. (2015, May). Static control-flow analysis of user-driven callbacks in Android applications. In*Proceedings of the 37th International Conference on Software Engineering-Volume 1* (pp. 89-99). IEEE Press.