Program Analysis for Mobile: How and Why to Run WALA on Your Phone

Julian Dolby

IBM Thomas J. Watson Research Center
United States
dolby@us.ibm.com

Abstract

As mobile devices become ubiquitous, security of such devices has become a serious concern. Attacks on the devices themselves are a danger, as is theft of data they contain. Static analysis of the devices' software is one approach to verifying the absence of security, and several tools have been created to analyze apps for potential attacks and vulnerabilities. Many tools focus on single apps, but there are starting to be tools that look for possible vulnerabilities or attacks due to multiple apps on a single device that can communicate. Such analysis depends on having access to the relevant apps, and hence has been proposed to be performed on app stores. One challenge in the Android environment is that apps are often installed from multiple sources, such as development builds of apps installed from developer sites, e.g. Mozilla Aurora pre-released of Firefox. Ultimately, sometimes the device itself is the only place with the full set of apps used on that device.

This suggests that running analysis on the device itself is attractive, at least in terms of having all the relevant code. Furthermore, app communication can be configured on the device itself, raising the possibility of analyzing communication risk when it is configured. However, this approach has a variety of challenges: 1) analysis tools are not typically mobile apps themselves, yet they somehow need to be built for and deployed on mobile devices. 2) Analysis tools are often resources intensive, and mobile devices need the resources to perform analysis. 3) Analysis can also be a major drain on battery life, so care must be taken not to heedlessly drain power. We describe our preliminary work toward running program analysis on mobile devices, focusing on running the WALA framework on Android devices. We describe how WALA can be built and deployed for Android; since WALA is Java code, it is actually straightforward to do this, both using Eclipse and Maven-based command-line tools. We also provide some evidence that performance is reasonable.

Keywords WALA, analysis, mobile

General Terms Program Analysis

Categories and Subject Descriptors D.3.4 [*Programming Languages*]

1. Introduction

Mobile devices have become the primary means for interaction with online information and services; being able to find information anywhere and any time has revolutionized many aspects of life. During casual conversation, we no longer wonder idly how tea came to India but can find out in real time.[1] Beyond general knowledge, though, we also expect our mobile devices to know more and more about us: to inform us of flight delays, public transit service changes, arrival of packages, and much more.

This requires our devices to have a lot of personal information and potentially a variety of specialized apps to deliver services. And once these apps have our information, we need to be aware of how these apps use that information, and make sure we are comfortable with that use. These has been much work devoted to analyzing how apps use private information; one recent example is DroidInfer [5] that combines program dataflow analysis and a type system to detect misuse of private information.

And these apps communicate: if you have ever been given, without asking, driving directions to the airport shortly before an upcoming flight, you have experienced multiple apps working together to get travel plans, likely from your email, and combine that with location-based services. These apps need to communicate with each other, but this opens further avenues for attacks, in which multiple apps together have permission to release personal information for purposes unintended by the user. There have started

MobileDeLi'15, October 26, 2015, Pittsburgh, PA, USA
© 2015 ACM. 978-1-4503-3906-3/15/10...$15.00
http://dx.doi.org/10.1145/2846661.2846673

[1] Large-scale tea production in India started under the British East India Company[4]

to be tools designed to detect situations in which multiple apps can communicate private data; one example is ComDroid [1].

In the Android environment, apps are often installed from multiple sources, such as development builds of apps installed from developer sites, e.g. Mozilla Aurora prereleased of Firefox. Thus the device itself may be the only place with the full set of apps used on that device. Hence, we have been investigating running program analysis on Android devices. In particular, we have been running WALA [7]; WALA is an open-source program analysis project that is written in Java and can be built using with Eclipse and Maven-based command-line tools. Both of these build mechanisms are very compatible with Android.

- The Android Development Tools within Eclipse provide a convenient interface that takes ordinary Eclipse Java projects and deploy them to Android platforms.
- The Maven build tools supports apk packaging, and hence it is possible to compile Java code at the command line and deploy it to Android.

WALA has been extensively used for security analysis both in products [6] and in research work [2, 3].

One concern once the code can be built for Android is whether analysis can be run in practice. We have carried out experiments using a recent top-of-the-line tablet, an Asus ZenPad S 8 (Z580CA), which has 4 64-bit Intel Atom cores running at up to 2.33GHz, and it has 4GB of RAM. It runs Android 5.0 in our experiments, which allows the system to make use of the full RAM with a 64 bit address space. This device also has a large enough battery that we can experiment with program analysis for at least a day on battery power; however, we anticipate that analysis will mostly be done while installing apps, and hence will not be a major drain in normal usage.

2. Approach

We have made our analysis infrastructure for mobile devices available on GitHub [8]. The version there makes use of the Android Development Tools for Eclipse. Another approach we have successfully used for development is to install LinuxDeploy on the device, and then use the standard Maven-based command line tools to build and run WALA on the tablet. The only restriction is that the device must be rooted for this approach to work.

Overall, we have shown that WALA can run on Android devices, as the first step toward a mobile analysis system for mobile apps.

References

[1] E. Chin, A. P. Felt, K. Greenwood, and D. Wagner. Analyzing inter-application communication in android. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, pages 239–252, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0643-0. doi: 10.1145/1999995.2000018. URL `http://doi.acm.org/10.1145/1999995.2000018`.

[2] A. Feldthaus, M. Schäfer, M. Sridharan, J. Dolby, and F. Tip. Efficient construction of approximate call graphs for javascript IDE services. In D. Notkin, B. H. C. Cheng, and K. Pohl, editors, *35th International Conference on Software Engineering, ICSE '13, San Francisco, CA, USA, May 18-26, 2013*, pages 752–761. IEEE / ACM, 2013. ISBN 978-1-4673-3076-3. URL `http://dl.acm.org/citation.cfm?id=2486887`.

[3] S. Guarnieri, M. Pistoia, O. Tripp, J. Dolby, S. Teilhet, and R. Berg. Saving the world wide web from vulnerable javascript. In M. B. Dwyer and F. Tip, editors, *Proceedings of the 20th International Symposium on Software Testing and Analysis, ISSTA 2011, Toronto, ON, Canada, July 17-21, 2011*, pages 177–187. ACM, 2011. ISBN 978-1-4503-0562-4. doi: 10.1145/2001420.2001442. URL `http://doi.acm.org/10.1145/2001420.2001442`.

[4] history of tea in India. history of tea in india. https://en.wikipedia.org/wiki/History_of_tea_in_India.

[5] W. Huang, Y. Dong, A. Milanova, and J. Dolby. Scalable and precise taint analysis for android. In *Proceedings of the 2015 International Symposium on Software Testing and Analysis*, ISSTA 2015, pages 106–117, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3620-8. doi: 10.1145/2771783.2771803. URL `http://doi.acm.org/10.1145/2771783.2771803`.

[6] IBM Security AppScan Source. Ibm security appscan source. http://www-03.ibm.com/software/products/en/appscan-source.

[7] wala. T.J. Watson Libraries for Analysis (WALA). http://wala.sf.net.

[8] wala mobile. T.J. Watson Libraries for Analysis (WALA) Mobile. https://github.com/wala/WALA-Mobile.