FUSE: Inter-Application Security for Android

Joe Hurd*
joe@galois.com
Galois, Inc.

The Android platform currently provides only limited support for preventing collusion between multiple apps installed on the same device: the main security mechanism is that at installation time each individual app is required to declare a manifest of the Android permissions it needs to operate. However, the Android platform is also a rich source of security threats: there are several sensitive information sources (Wi-Fi state, contacts, microphone, etc.), several potentially risky sinks (internet, SMS, phone, etc.), and a potentially large number of inter-app information flow paths from the sources to the sinks. The Android permission mechanism is widely considered as lacking the expressive power to discriminate exploitable security vulnerabilities from benign behaviors.

Current program analysis tools that might be employed to compensate for this lack typically focus on finding defects (or security risks) in apps, but do not seek to assure their absence, so their applicability to security analysis is limited. Typically, users employ them by marking specific variables in the app under test, and then analyzing tool-discovered information flows between those variables. This analysis approach requires significant user effort and expertise, and is error-prone because evaluators often fail to identify information flow paths representing exploitable security vulnerabilities. In Android marketplaces, these shortcomings are accentuated by the possibility of multiple apps colluding to support an exploitable security vulnerability.

At Galois, we are addressing these challenges in app evaluation by developing single- and multi-app security analysis tools for deployment on Android marketplaces. We have demonstrated a research prototype tool, called FUSE, that computes sound approximations of app intent calls. By sound we mean that our tools produce no false negatives, but may produce false positives. FUSE computes the possible intent calls for an app by analyzing the bytecode of all methods in all app components. FUSE then populates an SQL database of all the components in a marketplace of apps and the intent calls and intent filters they support. Queries over this database are run to find all intra- and inter-app control flow paths based on intent calls and matching intent filters. The resulting control flow graph can be traversed to ensure the absence of control flow paths that violate a marketplace security policy. We have run this inter-app control flow analysis on an artificial marketplace consisting of 104 real apps, with analysis run-times of ten seconds or so, discovering over 3,000 possible intent calls between components, which represent 1,100 intent calls between apps. Our current work is focused on extending the scope of the security analysis tool to track information flow through and between apps.

*The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. Approved for Public Release, Distribution Unlimited.