Software Security for Mobile Devices

Eric Bodden

contributors:
Prof. Dr. Heiko Mantel, Prof. Dr. Markus Müller-Olm, Prof. Dr. Alexander Pretschner, Prof. Dr. Wolfgang Reif, Prof. Dr. Gregor Snelting, Dr. Alexandre Bartel, Kuzman Katkalov, Dr. Enrico Lovat, Martin Mohr, Benedikt Nordhoff, Matthias Perner, David Schneider, Dr. Artem Starostin, Dr. Kurt Stenzel, …
Android App Security

Android is the most widely used mobile operating system

- 85% market share at the end of 2015
  http://www.gartner.com/newsroom/id/3169417

Security problems of Android apps are frequent

- “Another Popular Android Application, Another Leak”
  https://www.fireeye.com/blog/threat-research/2015/08/another_popular_andr.html

- “Mobile Threat Monday: Evil Android App Steals Text Messages”
  http://www.pcmag.com/article2/0,2817,2478552,00.asp

- “Mobile Threat Monday: Android Malware Looks Safe, Steals Your Photos and Messages”

- “Top Mobile Apps Overwhelmingly Leak Private Data: Study”
Android Malware

BadAccent malware, 2014/2015
Android App Security

Must aid developers in protecting their applications
Ways to enhance security

Avoid & Detect Vulnerabilities

Notify

Developers

Notify & Inform

Available on the App Store
Get it on Google play

End User

Enforce
Our Goals and Approach

**Goal:** enable users to specify their personal security requirements
- confidentiality of sensitive data (i.e., information-flow control)
- limited uses of resources (i.e., usage control)
- protection shall complement Android’s built-in protection mechanisms

**Approach:**
- develop interfaces for users to specify their security requirements
- develop solutions for establishing these security requirements
- display the security guarantees or potential problems to users

Our Prototype: The RS³ Certifying App Store
The RS³ Certifying App Store

Provides typical functionality of an app store
- browsing apps and displaying details about them, e.g., a description
- intention: familiar to users, hence easy to use

Important: many RS³ technologies integrated
- two static information-flow analyses
- run-time monitoring and enforcement
- security certificates from security-by-design
Support of User-Defined Policies

Users may have different security concerns
- enable users to specify their individual security requirements

Accessible to non-expert users
- user interface abstracts from technical details
- users select categories of data they want to keep secret
- a short description is given for each category to help the user decide
- if users are unsure, they can just check-mark category

Further exploration of alternative interfaces is planned.
Integration of Multiple Analyses

Static security analysis
- ... is performed at design- or at compile-time
- ... provides guarantees about all possible executions
- if an app passes the analysis for some property, then the property is satisfied by every execution of the app
- static analysis can be used for information-flow control

Dynamic security enforcement
- ... is performed on-the-fly during execution of an app
- ... provides guarantees for an actual execution of an app
- if an execution of an app causes an action that would violate the property of interest then this action is prevented at run-time
- dynamic enforcement can be used for usage control

We use two approaches for two orthogonal security concerns.
The Type-based Security Analysis
(from Cassandra)

Analysis specified by a security type system
- security types are used to specify whether data is confidential or not
- security types are used to specify whether an information sink is trusted
- security type systems are sets of rules

Type system proven to be sound
- each application the analysis validates has secure information flow

Formal foundation of the analysis
- formal model of Dalvik bytecode operational semantics
- formal definition of security based on the semantics

First such analysis for Android with soundness proof.
Architecture of the App Store

Client-server architecture

- server stores and distributes apps
- client is an Android app that runs on off-the-shelf devices

Security technology is integrated in both client and server

- resource-intensive operations are performed on the server
- the user can specify his security requirements in the client app
Integration of Analysis Tools

JoDroid

app store server

app database

certifier using static analysis

 instrumentation for dynamic enforcement

DroidForce

Cassandra

also provides the infrastructure

mobile device

client app

app browser

verifier of static analysis results

dynamic enforcement

policy editors
Proof-Carrying Code
(from Cassandra)

General idea (Necula, 1997):
- provide program code together with proof of a property
- essential property: proof can easily be verified

In our case:
- server receives user policy and analyzes app (type inference)
- client obtains app and analysis result and verifies result (type check)
  - type check is fast
Presentation of Results (from Cassandra)

Results of static analyses are displayed to the user visually

- shows all flows of potentially sensitive information to an output channel

Users can quickly make informed decisions whether to install an app.
Overview of Integrated Tools

**Cassandra**: static information-flow analysis
- based on a security type system
- policy editor for selecting categories of data to be kept secret

**JoDroid**: static information-flow analysis
- based on program-dependence graphs
- uses the same policy editor as Cassandra

**DroidForce**: dynamic usage control
- prohibits unwanted functionality of apps
  - e.g., “not more than 5 SMS are sent per hour”
- editor allows users to instantiate predefined policy templates
- uses static pre-analysis of **FlowDroid**
Investigated Apps so far

Initially: Self-developed case studies
- simplified apps with the core functionality of existing apps
- 8 case studies in total

<table>
<thead>
<tr>
<th>App</th>
<th>Functionality</th>
<th>Security Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bookmark Navigator</td>
<td>Display browser bookmarks.</td>
<td>Browsing history is leaked.</td>
</tr>
<tr>
<td>Minute Man</td>
<td>Break off calls to limit costs.</td>
<td>Called telephone numbers are leaked.</td>
</tr>
<tr>
<td>Distance Tracker</td>
<td>Record jogging distance.</td>
<td>Location of the device is leaked.</td>
</tr>
</tbody>
</table>

Recently: Third-party apps
- so far: analyzed 48 apps from the F-Droid app store
- static analysis: found problematic flow of information in some apps
- dynamic enforcement: successful instrumentation of all apps
- model-driven development: found some apps promising complex informal security guarantees
FlowDroid soon in widespread productive use

FlowDroid currently being adopted for use by one of the world’s largest app-store providers

Will go live in fall

More on FlowDroid at:
https://github.com/secure-software-engineering/soot-infoflow
Conclusion

Goals:
- ensure confidentiality of sensitive data
- limited uses of resources
- enable users to enforce their individual security requirements

Approach:
- develop an app store based on RS$^3$ tools
- provide interfaces for specifying requirements

Prototype: The RS$^3$ Certifying App Store

Outlook:
- provide security guarantees for and detect security problems in third-party apps

Live Demo Possible!
Selected Relevant Publications


