15th International School on Foundations of Software Security and Design
August 31st, 2015, Bertinoro

Certifying the Security of Android Applications with **Cassandra**

Steffen Lortz, Heiko Mantel, David Schneider, Artem Starostin, Timo Bähr, Alexandra Weber

Modeling and Analysis of Information Systems (MAIS), TU Darmstadt, Germany

partially based on

- a short talk by David Schneider
  at the 36th IEEE Symposium on Security and Privacy
- a talk by Steffen Lortz
  at the 4th ACM Workshop on Security and Privacy in Smartphones and Mobile Devices

MAIS
Modeling and Analysis of Information Systems

Motivation

privacy of sensitive data is a problem on Android mobile devices

- they store and collect plenty of sensitive information
- they are (by design) connected to the Internet
- leakage of sensitive data has often been reported in prior work

there are existing security mechanisms for Android, e.g.,

- the permission system
- malware scanning by Google
- other analysis tools from the literature

all of these suffer from (at least) one of two essential problems:

Problem 1: The provided security guarantees are unclear.

Problem 2: The mechanisms are not available to end users.

Our Solution: Cassandra

Certifying App Store for Android

allows user to certify that apps comply with his privacy requirements

- before installation
- using static information flow analysis

our novel contributions:

- a **semantically justified** information flow analysis for Android applications with a soundness proof
  - addresses Problem 1
- a tool making the analysis **available to end users** on their mobile devices
  - addresses Problem 2

Semantically Justified Certification

formal foundation of the analysis

- formal specification of Dalvik bytecode operational semantics
- notion of security defined as a noninterference property

analysis specified by a security type system

- on the bytecode level

$$\frac{M[p] = \texttt{binop}\ x_a, x_b, x_c, bop \quad \mathsf{rda}' = \mathsf{rda}[x_a \mapsto \mathsf{rda}(x_b) \sqcup \mathsf{rda}(x_c) \sqcup se(p)]}{M, region_{P,M}, \mathsf{fda}, \mathsf{mda}, ret, se \vdash p : \mathsf{rda} \to \mathsf{rda}'}$$

$$\frac{M[p] = \texttt{new-instance}\ x_a, c \quad \mathsf{rda}' = \mathsf{rda}[x_a \mapsto se(p)]}{M, region_{P,M}, \mathsf{fda}, \mathsf{mda}, ret, se \vdash p : \mathsf{rda} \to \mathsf{rda}'}$$

$$\frac{M[p] = \texttt{iget}\ x_a, x_b, f \quad \mathsf{rda}' = \mathsf{rda}[x_a \mapsto \mathsf{rda}(x_b) \sqcup \mathsf{fda}(f) \sqcup se(p)]}{M, region_{P,M}, \mathsf{fda}, \mathsf{mda}, ret, se \vdash p : \mathsf{rda} \to \mathsf{rda}'}$$

$$\frac{M[p] = \texttt{iput}\ x_a, x_b, f \quad \mathsf{rda}(x_a) \sqcup \mathsf{rda}(x_b) \sqcup se(p) \sqsubseteq \mathsf{fda}(f)}{M, region_{P,M}, \mathsf{fda}, \mathsf{mda}, ret, se \vdash p : \mathsf{rda} \to \mathsf{rda}}$$

type system proven to be sound

- all typable applications are secure

To our knowledge, this is the first static information flow analysis for Android to be proven sound!

MAIS Modeling and Analysis of Information Systems

The Setting

our work is based on work by Barthe et al. for Java bytecode [Barthe, Picardie & Rezk, '06]

however:

- Dalvik bytecode differs from Java bytecode (register- vs. stack-based)
- previous work only focused on a small subset of instructions
- to provide guarantees for real apps, we need to consider more instructions

challenge:

- find a level of abstraction keeping the size of the model small while being suitable for reasoning about information flow

Formal Model of Dalvik Bytecode

formal syntax of Dalvik bytecode

- group patterns of instruction…
  - …that exist for optimization reasons
    (e.g., from bit-lengths of constants, types of arguments, …)
  - …that are parametric in some way
    (e.g., binary operations, …)

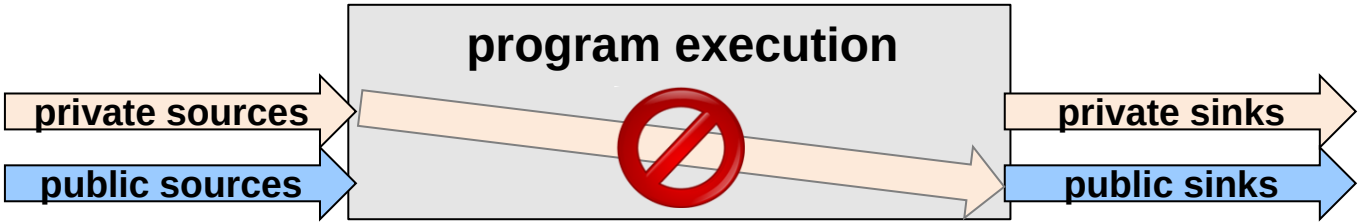
Abstract instruction	Concrete instruction
`const` x_a, n	`const`, `const/4`, `const/16`, `const/high16`
`binop` x_a, x_b, x_c, op	`add-int`, `sub-int`, `mul-int` (+ 13 more)

  - 55 abstract instructions capture 211 of 217 concrete instructions

Security Property

noninterference [Goguen & Meseguer, ‘82]

- the public output of a program must not depend on its secret input
- an observer does not learn secret input by observing the program



two-level lattice of *security domains* $(D, \sqsubseteq)$

- D encodes the levels of confidentiality
- $\sqsubseteq$ encodes how information may flow between domains

indistinguishability relations $\sim_p$

- relate pairs of program configurations that look the same to an observer who can access only public information (depending on the *policy* p)

Security Property

Definition: Security of Methods

A method M of a program P is *secure* w.r.t. a policy p iff any two indistinguishable initial states produce indistinguishable final states.

$$\begin{array}{ccc} \langle 0, \mathrm{r}_1, \mathrm{h}_1 \rangle & \Downarrow_{P,M} & \langle \mathrm{v}_1, h_1' \rangle \\ \sim_p \Big| \quad \Big| \sim_p & & \sim_p \vdots \quad \vdots \sim_p \\ \langle 0, \mathrm{r}_2, \mathrm{h}_2 \rangle & \Downarrow_{P,M} & \langle \mathrm{v}_2, h_2' \rangle \end{array}$$

intuition:

- if only the secret input of a program differs, public output is the same
- hence: the public output does not depend on secret input

Definition: Security of Programs

A program P is *secure* w.r.t. a policy p iff all its entry points are secure w.r.t. p.

Security Type System

the information flow analysis is formalized as a security type system

- typing judgments of the following form:

$$M, region_{P,M}, \mathsf{fda}, \mathsf{mda}, ret, se \vdash n : \mathsf{rda} \rightarrow \mathsf{rda}'$$

- types the instruction at program point n in method M
  - rda and rda' assign security domains to registers before and after execution, respectively

example of a typing rule:

$$\frac{M[p] = \texttt{binop}\ x_a, x_b, x_c, bop \qquad \mathsf{rda}' = \mathsf{rda}[x_a \mapsto \mathsf{rda}(x_b) \sqcup \mathsf{rda}(x_c) \sqcup se(p)]}{M, region_{P,M}, \mathsf{fda}, \mathsf{mda}, ret, se \vdash n : \mathsf{rda} \rightarrow \mathsf{rda}'}$$

Soundness of the Type System

Theorem: Soundness

For all programs P and all policies p

if P is typable with respect to p,

then P is secure with respect to p.

Solution to Problem 1: The provided security guarantee is clear.

we have…

- …a precise, formal definition of security
- …a proven guarantee that the analysis enforces this notion of security

Availability to End Users

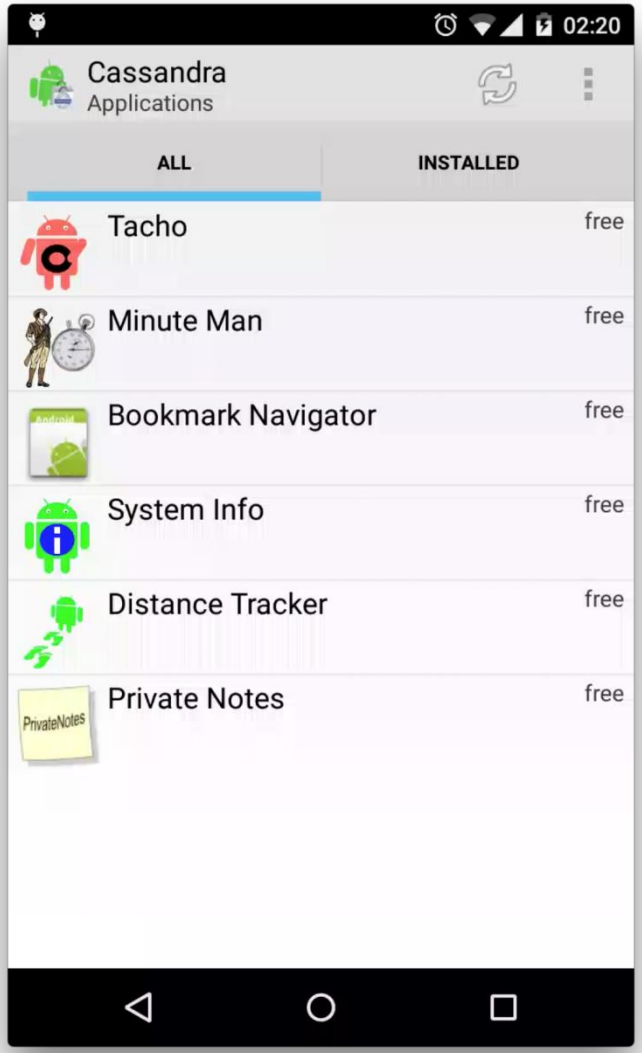
Problem 2: The mechanisms are not available to end users.

goal: make analysis available to end users on their mobile devices

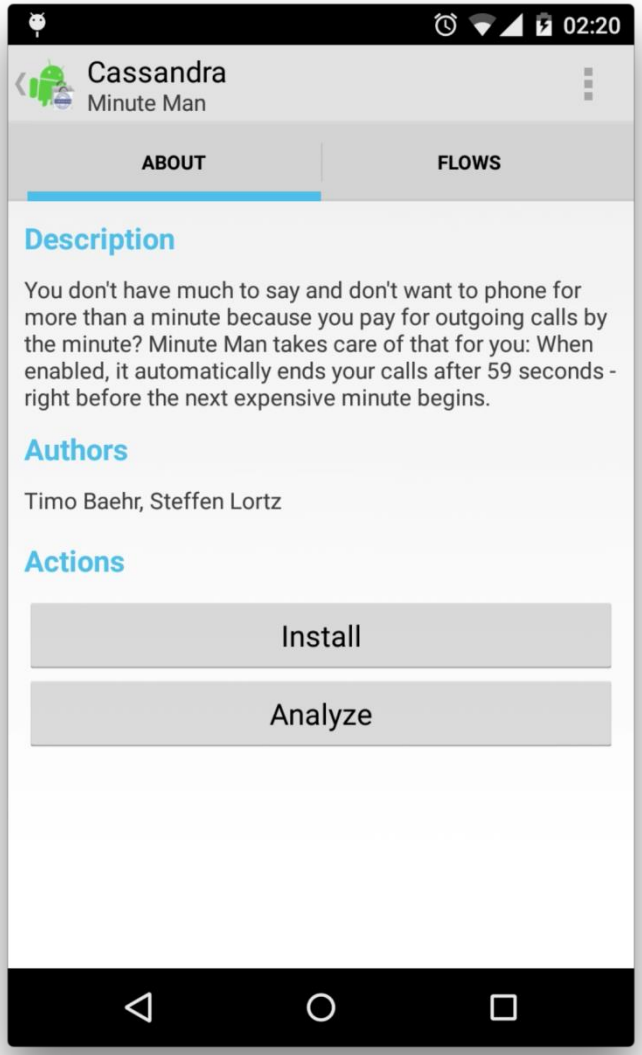
challenges:

- users have different security requirements
  - how can a non-expert user express his requirements in a security policy?
- mobile devices have limited resources
  - certifying applications on the device is costly

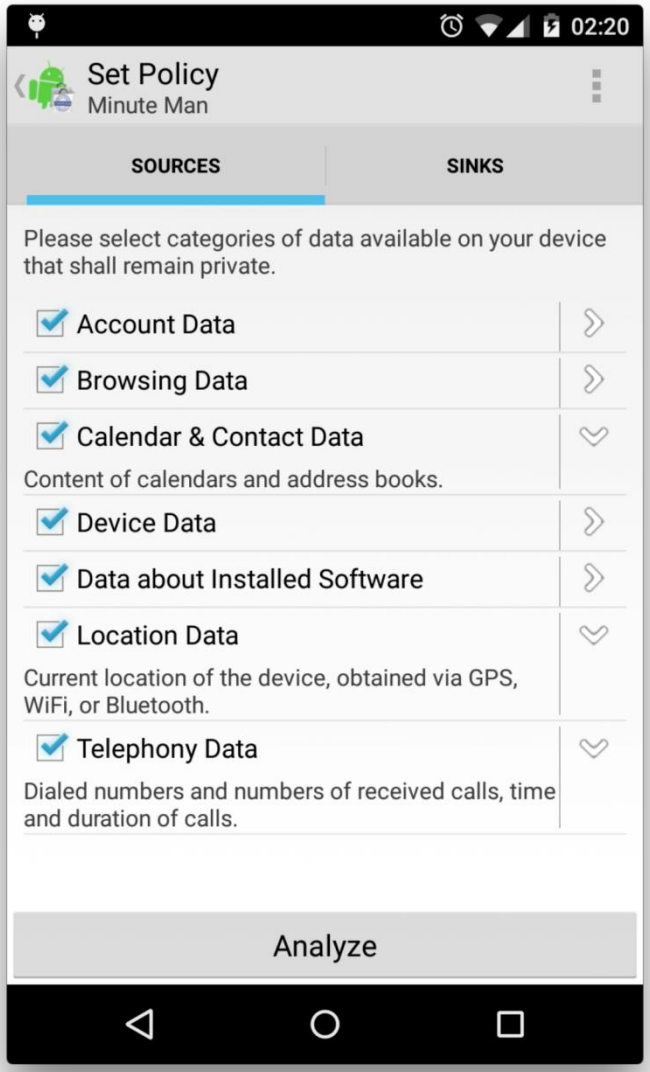
Cassandra in Action



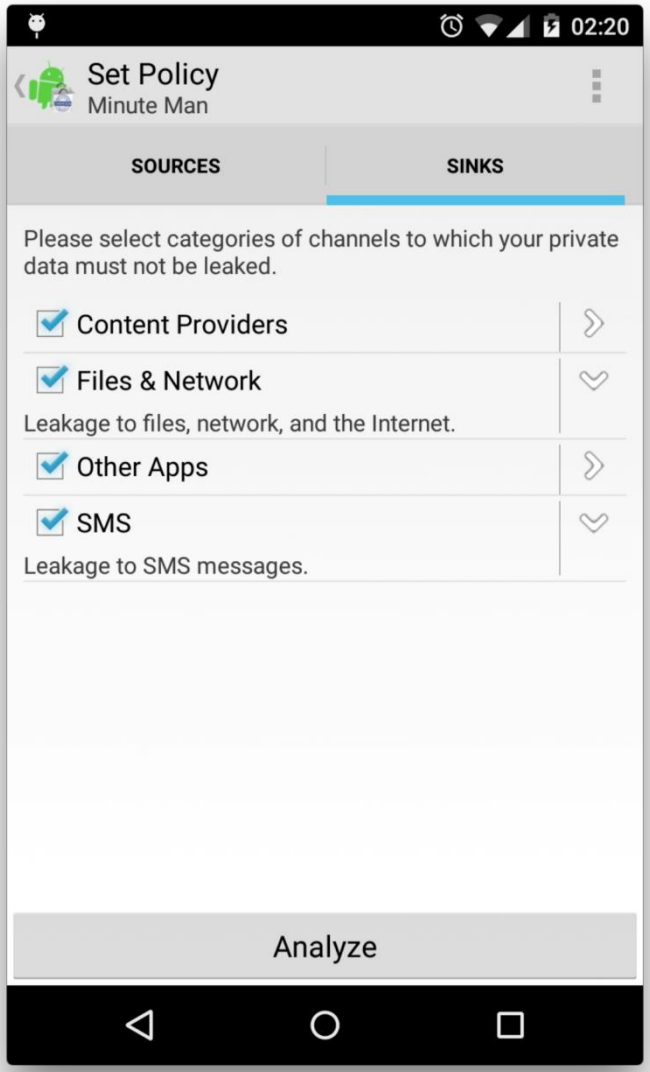
Cassandra in Action



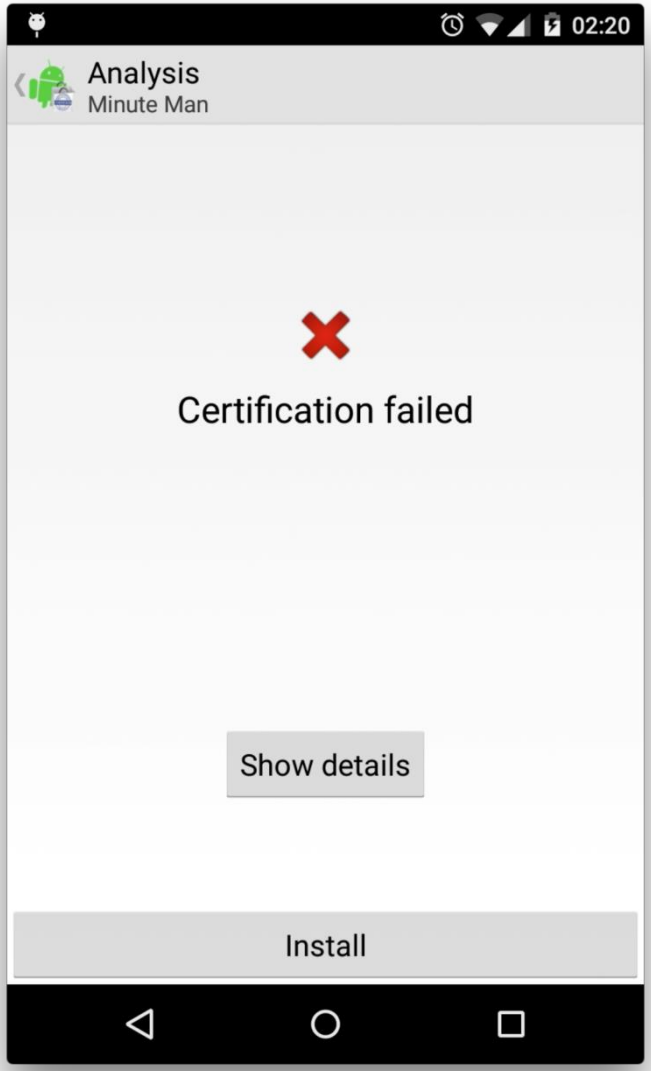
Cassandra in Action



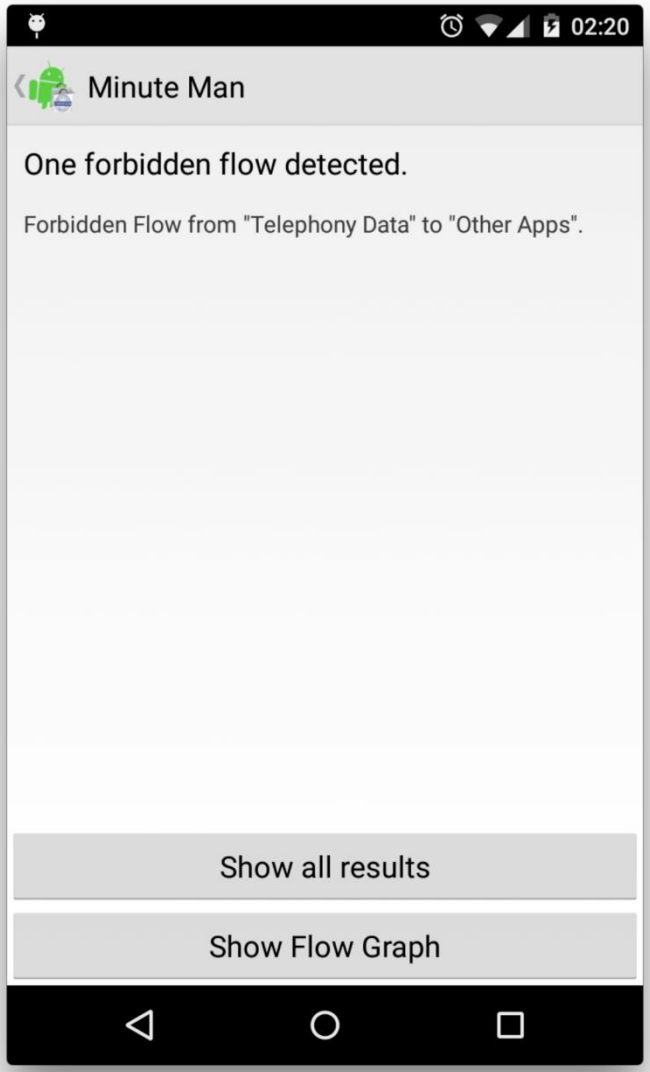
Cassandra in Action



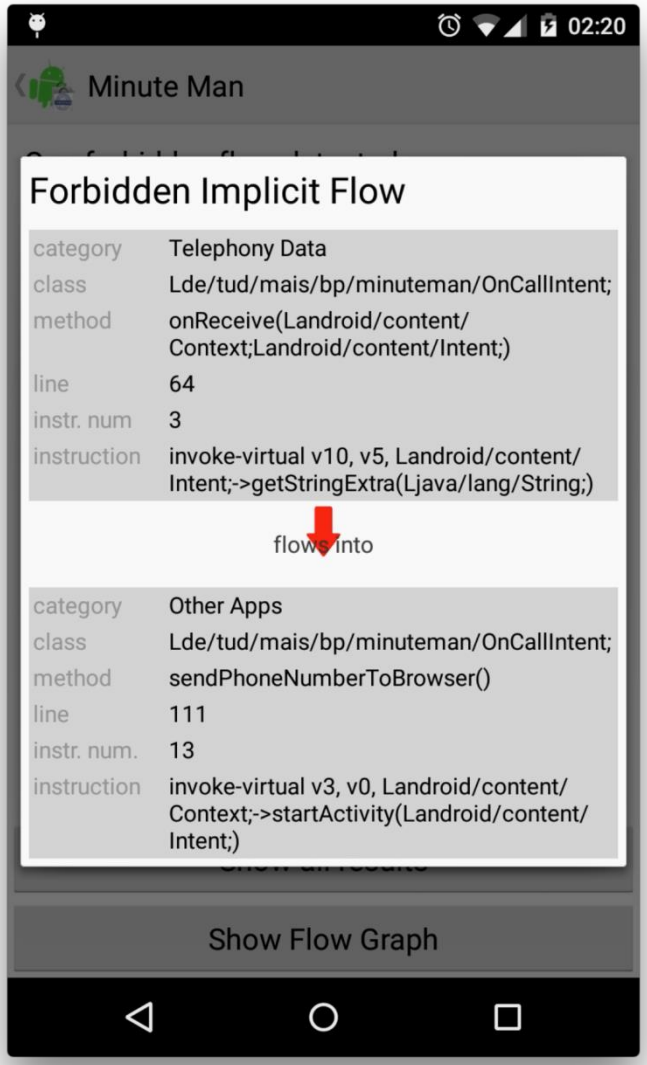
Cassandra in Action



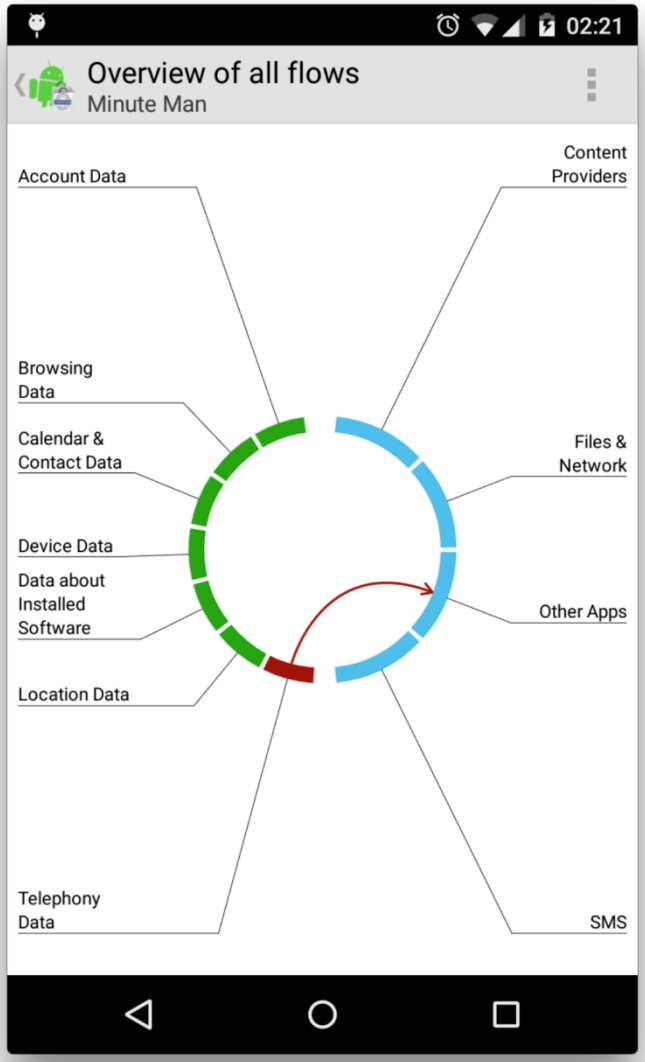
Cassandra in Action



Cassandra in Action



Cassandra in Action



MAIS
Modeling and Analysis of Information Systems

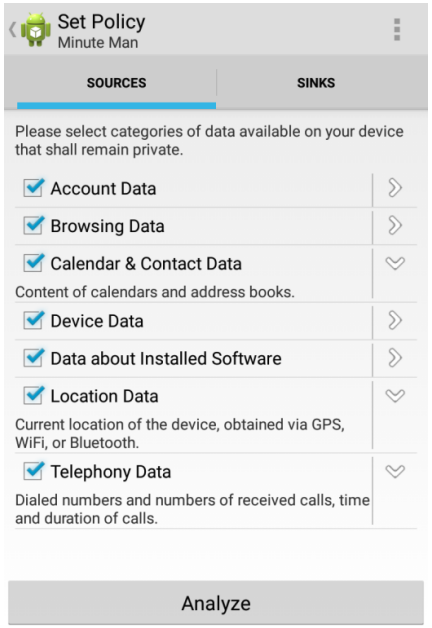
Certifying Applications

input to the certification:

- a partial security typing determined from the user's choice of categories

output of the certification:

- a complete and valid security typing of the app, if possible
- hence, certification is type inference
- the complete typing is a *security certificate*



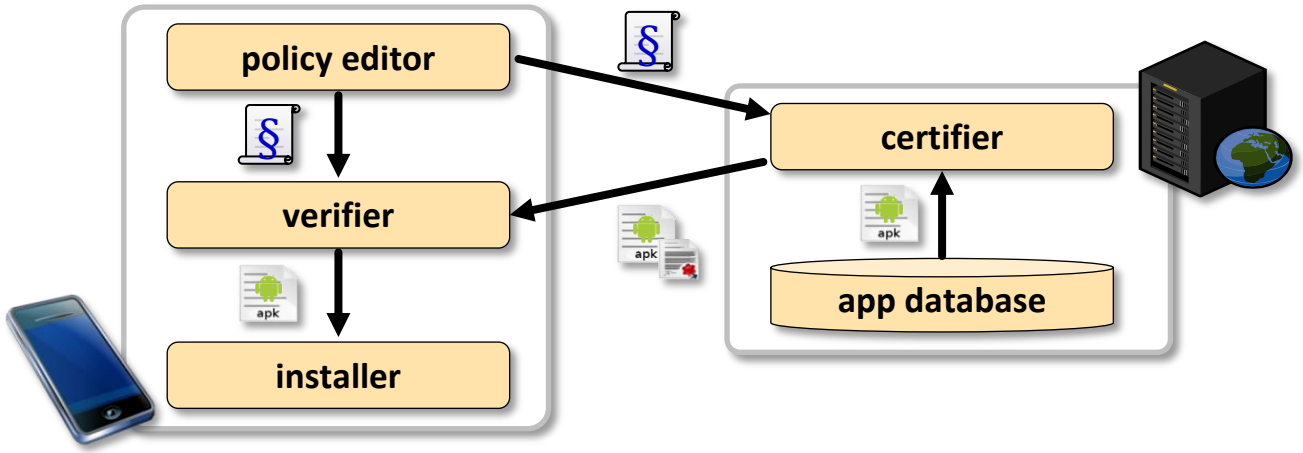
can exceed the resources of a mobile device

- limited computing power: may take too long to be convenient
- limited power: may drain the battery

hence: certify applications on the server that distributes apps

Architecture

we use *proof-carrying code* [Necula, '97]



- compute the security certificate for an app and policy on the server
  - complex type inference
- transmit the certificate along with the application to the mobile device
- verify the certificate on the mobile device
  - quick and simple type check
- cache certificates to increase response time

Main Contributions

in summary, our contributions are:

- a **formal operational semantics** for a large subset of the Dalvik bytecode instruction set
- a **semantically justified** static information flow analysis for Android applications with a soundness proof

Solution to Problem 1: The provided security guarantee is clear.

- an architecture and a tool making the analysis **available to end users** on their mobile devices

Solution to Problem 2: The solution is available to end users.

Cassandra: Present and Future

used as the basis of the **RS³ Certifying App Store**

- integrates different security technologies into an app store
- presented by a poster at S&P ’15

ongoing:

- experimental evaluation with open source apps from F-Droid

next goal:

- certification of third-party apps

to get more info about Cassandra:

- visit: http://www.mais.informatik.tu-darmstadt.de/cassandra.html
- read our paper:

  S. Lortz, H. Mantel, A. Starostin, T. Bähr, D. Schneider, and A. Weber.
  Cassandra: Towards a Certifying App Store for Android. In Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, 2014.

Cassandra

Thank you for your attention!

This work was supported by the DFG in the Priority Program
“Reliably Secure Software Systems” (RS³).

http://www.reliably-secure-software-systems.de