Privacy in the Mobile Age

• June 2010: “iPad app transmits login data in the clear”
• Aug. 2010: “Leaky Apps”
• Oct. 2010: “TaintDroid detects sneaky Android Apps”
• Dec. 2010: “Your Apps are Watching You”
• Feb. 2011: “Snooping Apps”
• Apr. 2011: “iPhone Stored Location in Test Even if Disabled”
• May 2011: “Personal Firewall for Android”
• Aug. 2011: “Accelerometer used to log smartphone keystrokes”
Privacy today

- Dec. 2011: Facebook glitch gave access to other users’ private pictures
- Feb. 2012: “Path uploads your entire iPhone address book to its servers”
- June 2012: “Researchers devise hack that sneaks Android malware into Google market”
- June 2012: “LinkedIn iOS app gathers and transmits user calendar data”
- October 2012: “PlaceRaider: Virtual Theft in Physical Spaces with Smartphones”
- April 2013: „Snapchat allows to share self-destructing photos“
Your apps are watching you - Pandora

Application receives personal information

Pandora: personalized online radio for music streaming

Application violates user’s privacy by forwarding personal information to third parties

A Typical Android App

What it does

Permissions it needs

- take pictures and videos
- coarse (network-based) location
- fine (GPS) location
- full Internet access
- read phone state and identity
- control flashlight
- view network state
- prevent tablet from sleeping
- prevent phone from sleeping
- disable or modify status bar
- modify/delete USB storage contents
- modify/delete SD card contents

Flashlight XL Deluxe Bright Edition™
• Information flow control for apps, services, ...
• necessary for...
  – private information (eg. CreditCardNumber)
  – derived information from private information
    (eg. CreditCardNumber + 42, rebmuNdraCtiderC)
Big Picture of Secure App Deployment

Development with **IFlow**
- Model the app
- Define IF-Properties
- Prove IF-Properties

**App**
- IF Properties
- Proof

Secure App-Store
- IF Properties adequate?
- App satisfies IF Properties?

IF Properties adequate?

IFlow - Developing Systems with Secure Information Flow
Goal of IFlow

IFlow integrates

• a model driven approach  
  [Seehusen, Alghathbar, Jürjens, Basin, Breu, Reif, ...]
• formally verified IFC properties  
  [Meseguer, Rushby, Sands, Mantel, Reif, ...]
• language based type systems for IFC  
  [Denning, Volpano, Myers, Sabelfeld, Simonet, Snelting, ...]
• (with applications to mobile platforms)  
  [Enck, Martinelli, Aktug, Sannella, Barthe, ...]

Result: a Methodology for developing information flow secure mobile applications
Example: Travel Planner App

- Travel planner app on mobile
- Developed by travel agency
- Access to offers via agency
- Booking directly with airline/hotel
- Agency gets commission
- Integration of calendar and credit card app on mobile

Credit card details (What) are sent only after confirmation (When) and only to the airline/hotel (Who).

Private calendar entries (What) are not visible to the Travel Planner App (Who).
Calendar App: Add appointment

IFlow - Developing Systems with Secure Information Flow
Travel Planner App

Select Appointment

Important Business
25.02.2011

Hilton in London
20.01.2011

RS3-Kickoff
20.02.2011

Flight to London
20.01.2011

Flight to Munich
20.01.2011

Travel details and price

Lufthansa
Departure: 9:30  Arrival: 12:45
Price: 830 Euro

Bellagio
Price: 1200 Euro

Total: 2480 Euro

Book&Pay  Back to flight

Back to hotel  Cancel
IFLow Development Process

1. Model application including security policies for information flow control
2. Model application specific security properties
3. Verify properties formally
4. Transform model into code skeletons and run IFC type check
5. Manually fill in empty parts with syntactical restrictions (e.g. no declassification)

→ Formally Verified Guarantees for Information Flow
public OK bookFlightOffer(BookFlightOffer inmsg) {
    int id = inmsg.id;
    CreditCardData ccd_decl = inmsg.ccd;
    Manual.processBooking(id, ccd_decl);
    return new OK();
}
Formal verification

UML Model + application-specific Properties $P$

what, when, where, who

generate + program

IFC language

type check: Java code $\vdash$ NIP

labels, principals, declassify,

transform

Formal Model FM

assume: FM $\vdash$ NIP

prove: FM $\vdash$ NIP $\rightarrow$ $P$

refine

ASMs, temporal logic, algebraic specs

Java code $\vdash$ $P$

Android++
What do we need?

- Platform independent model:  
  *UML + Profile + Labels + Methodology*

- Formal model:  
  *Abstract State Machines + Temporal Logic + algebraic specifications + formalization of noninterference*

- Platform specific model:  
  *Decentralized Label Model + lattice + principles*

- Programming Language with IFC type system:  
  *Java and Information Flow (JIF) or Joana*
Modeling the Travel Planner

Components

IF properties

IF lattice
Modeling the Travel Planner: Communication

```
1: Start() -> GetFlightOffers
    ref

ref
ChooseFlight

ref
ReleaseCCDetails

ref
BookFlight

2: Ok()
```
ReleaseOfCreditCardDetails

When: confirm кредитCardDetails, airline: confirmed
   {Customer->}

Where: declassify creditCardDetails from {Customer->} to {Customer->Airline}

What: creditCardDetails

Who: TravelPlanner
Information flow theory

• Many model-based noninterference models exist
  [Goguen/Meseguer82, Sutherland86, McCullough87, McLean94, Rushby92, Mantel03, v.d.Meyden07]

• Van der Meyden
  – allows intransitive policies
  – „structured state“
    → allows us to talk about variables

• Proof with „Unwinding conditions“
Formal verification with KIV
Summary

**IFlow integrates**
- a model driven approach
- formally verified IFC properties
- language based type systems for IFC