A certified reference validation mechanism for the permission model of Android

Gustavo Betarte Juan Campo Felipe Gorostiaga Carlos Luna

InCo, Facultad de Ingeniería, Universidad de la República, Uruguay.

IMDEA Software Institute, Spain.

October 16, 2017

Table of contents

Table of contents

Table of contents

Table of contents

Summary

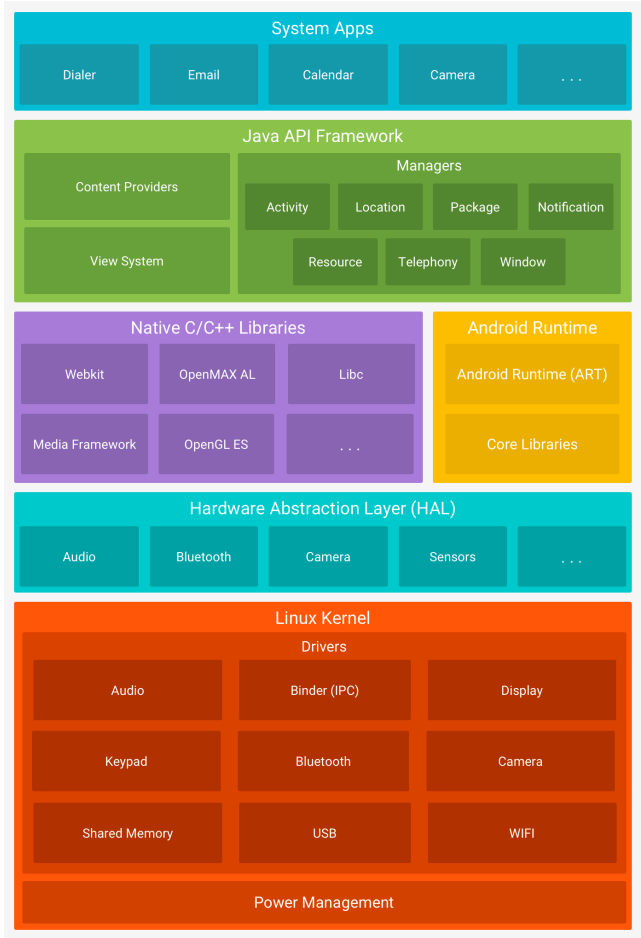
Why Android?

- Present in more than a billion mobile devices.
- Target of many attacks.
- Informal and incomplete documentation.

Summary

Introduction to Android

- Open-source operating system for mobile devices.
- Developed by Google and the Open Handset Alliance (OHA)



Introduction to Android

- Two kinds of applications:
  1. System applications shipped with the Android distribution
     Eg. Clock, Contacts
  2. User applications developed by third parties
     Eg. WhatsApp, Facebook

Introduction to Android

- Two kinds of applications:
  1. System applications shipped with the Android distribution
     Eg. Clock, Contacts
  2. User applications developed by third parties
     Eg. WhatsApp, Facebook
- Both kinds of applications have access to the device's resources/services, as well as to other applications' resources.

Application components

- **Activities**
  - They comprise the application screens
  - They handle the user's interaction with the application
- **Content Providers**
  - They handle data sharing between applications
  - Interface between data and external applications
- **Services**
- **Broadcast Receivers**

Application components

- **Activities**
  - They comprise the application screens
  - They handle the user's interaction with the application
- **Content Providers**
  - They handle data sharing between applications
  - Interface between data and external applications
- **Services**
- **Broadcast Receivers**

Application components

- **Activities**
  - They comprise the application screens
  - They handle the user's interaction with the application
- **Content Providers**
  - They handle data sharing between applications
  - Interface between data and external applications
- **Services**
- **Broadcast Receivers**

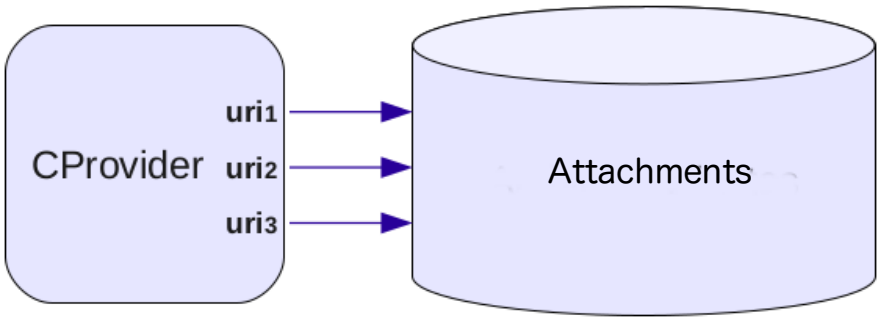
Application components: an example

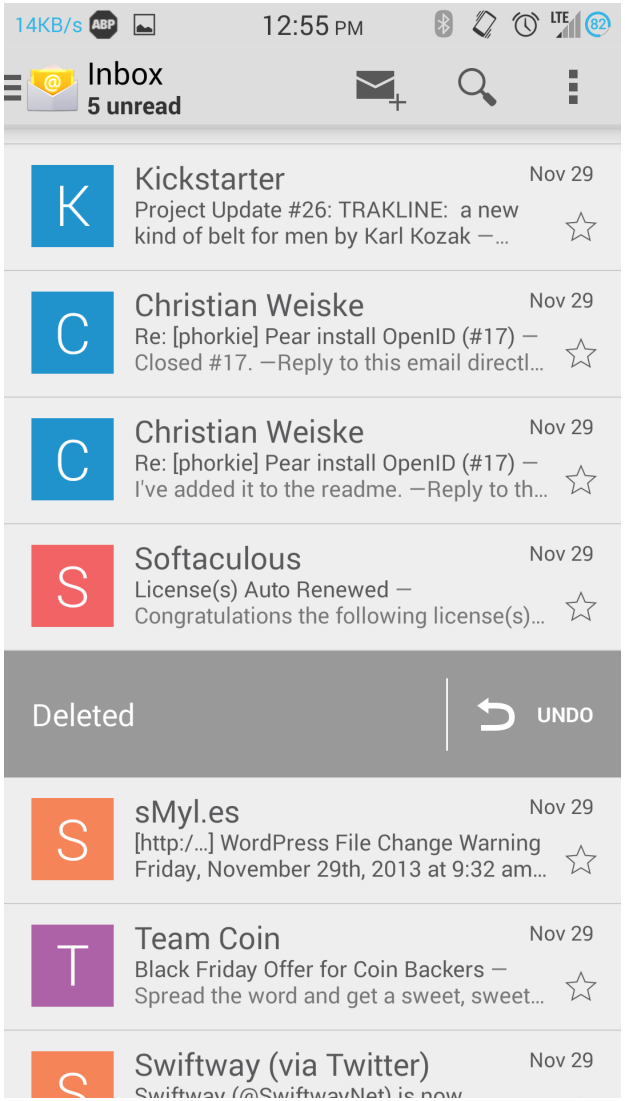
Activities

- Inbox
- New mail

Content Providers

- Attachments





Comunication between Components

- Access to Content Providers:
  - Queries to *URI*s
- Access to any other component:
  - Intents

Android's security model

Access to device and external applications must be regulated to keep:

- Data integrity and confidentiality
- Costs control by the user
- System's proper functioning
- ...

Principle Of Least Privilege

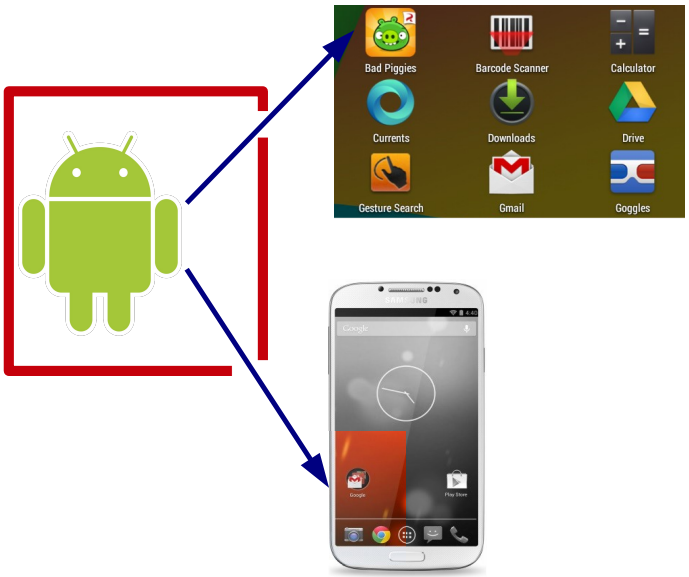
Principle Of Least Privilege

- **Application sandbox**

Principle Of Least Privilege

- **Application sandbox**

- **Permission system**



AndroidManifest

- XML file which every Android application must provide
- It includes static declarations like:
  1. Requested permissions
  2. Custom permissions
  3. Application's components
  4. ...

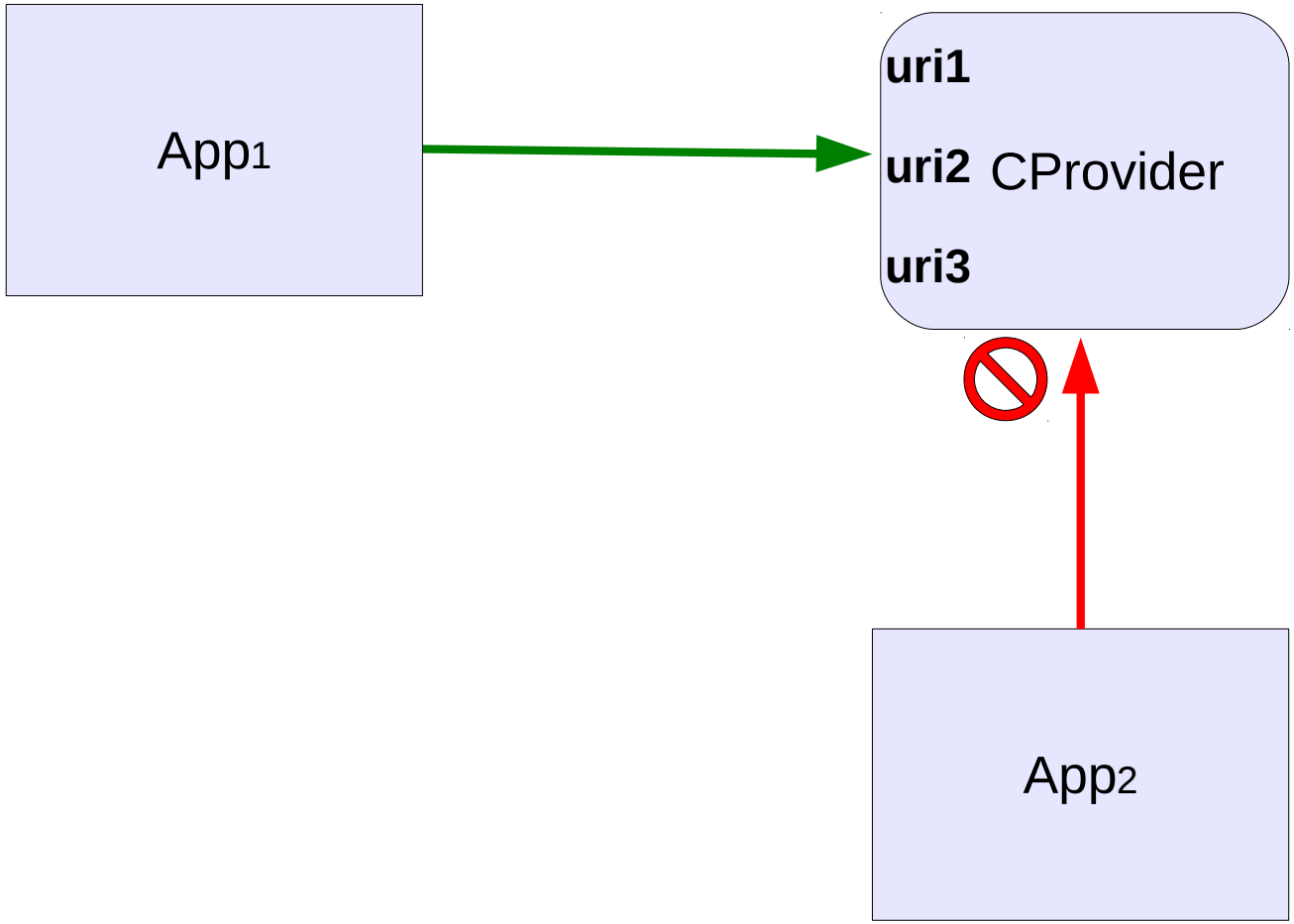
AndroidManifest: Example

```
<manifest package="com.cpexample" ... >

    ⋮

    <uses-permission android:name="android.permission.SEND_SMS" />

    <application
        android:permission="android.permission.SET_WALLPAPER" ... >

        <activity
            android:name="com.cpexample.MainActivity"
            android:permission="android.permission.CALL_PHONE" ... >
        </activity>

        <provider android:name="com.cpexample.MiProvider"
            android:permission="android.permission.SEND_SMS" ... >

        </provider>

        ⋮

    </application>

</manifest>
```

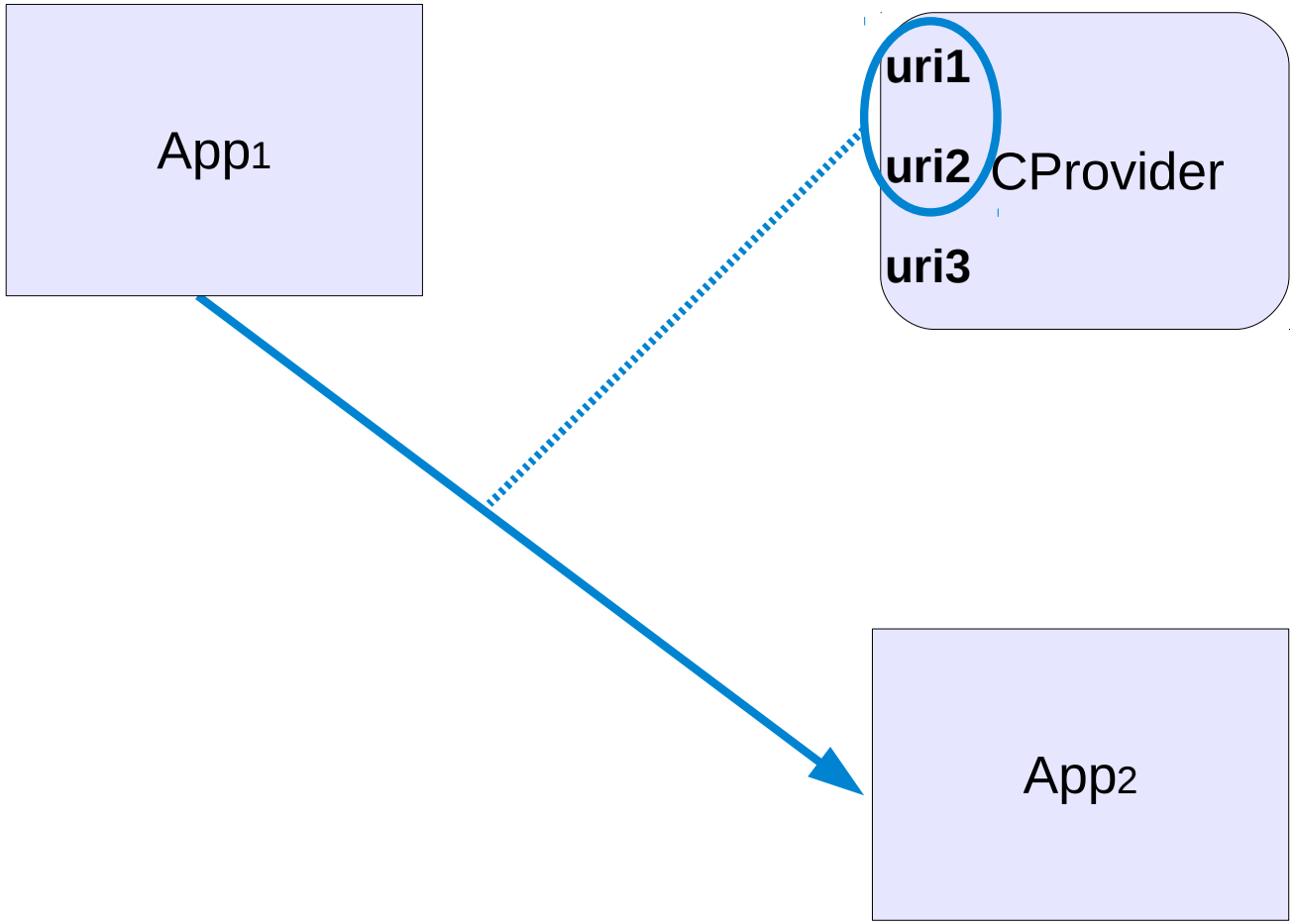
Permission delegation

- Permission delegation between components
- Permissions granted until revocation
- Two delegation mechanisms:
  - Pending intents
  - URI permissions

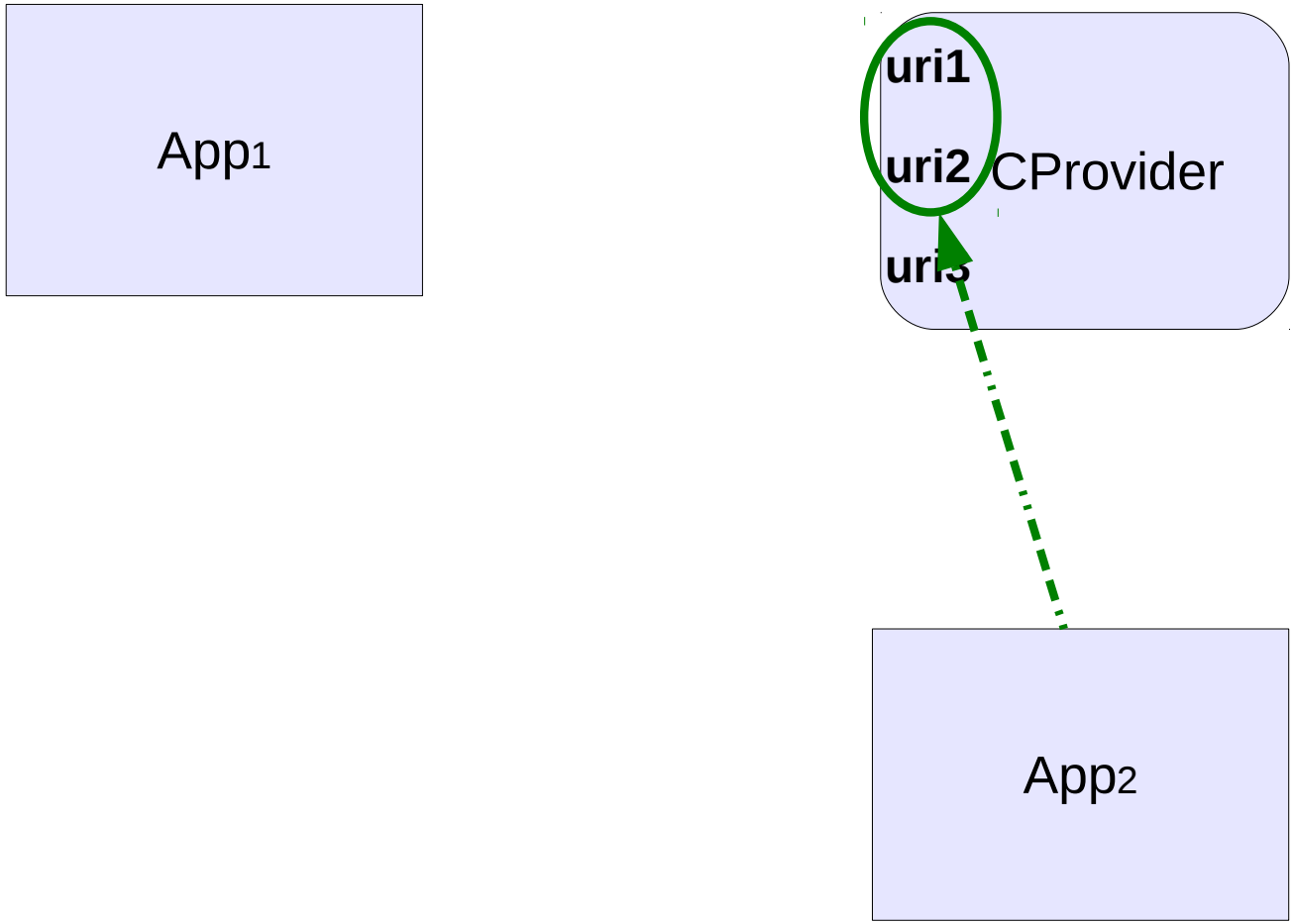
Permission delegation: URI permissions



Permission delegation: URI permissions



Permission delegation: URI permissions



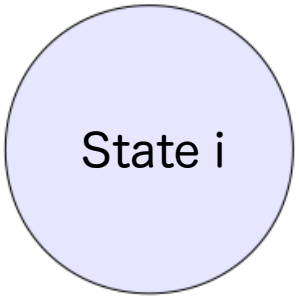
Summary

General characteristics

- Formalization of Android's security system
- Developed using the proof assistant Coq
- Specially focused on:
  - Permission system
  - Interaction between components and the system
- High order specification based on state machines

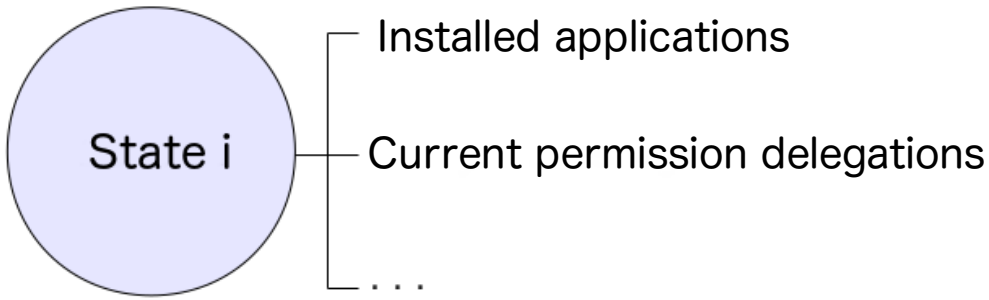
General characteristics

- Formalization of Android's security system
- Developed using the proof assistant Coq
- Specially focused on:
  - Permission system
  - Interaction between components and the system
- High order specification based on state machines



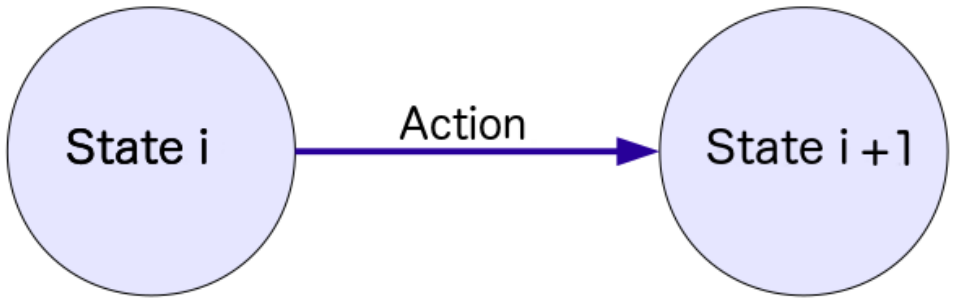
General characteristics

- Formalization of Android's security system
- Developed using the proof assistant Coq
- Specially focused on:
  - Permission system
  - Interaction between components and the system
- High order specification based on state machines



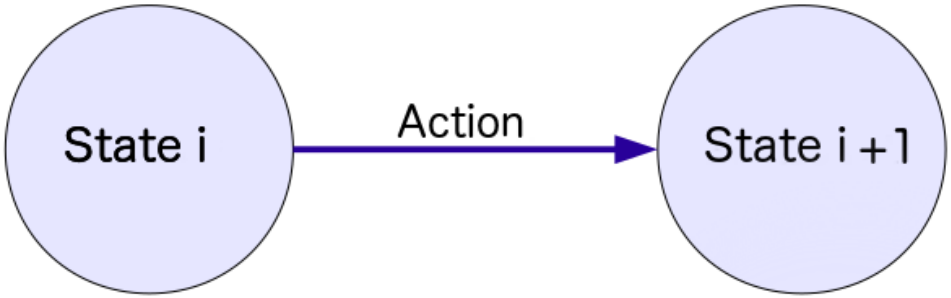
General characteristics

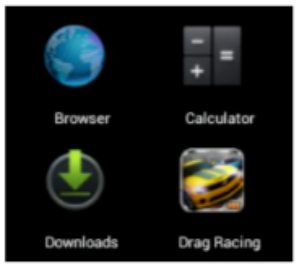
- Formalization of Android's security system
- Developed using the proof assistant Coq
- Specially focused on:
  - Permission system
  - Interaction between components and the system
- High order specification based on state machines



General characteristics

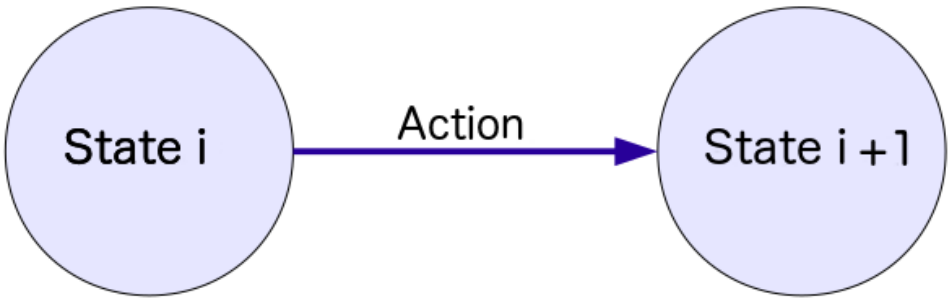
- Formalization of Android's security system
- Developed using the proof assistant Coq
- Specially focused on:
  - Permission system
  - Interaction between components and the system
- High order specification based on state machines

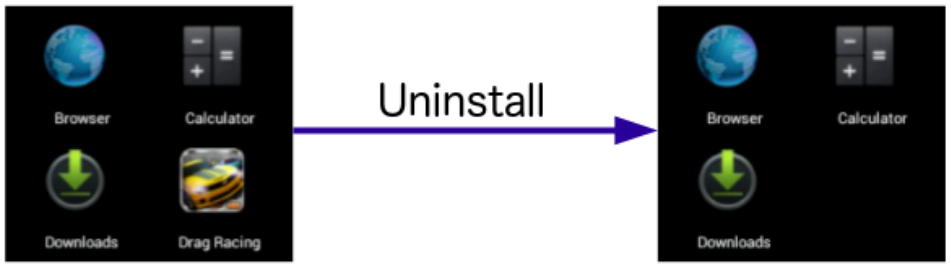




General characteristics

- Formalization of Android's security system
- Developed using the proof assistant Coq
- Specially focused on:
  - Permission system
  - Interaction between components and the system
- High order specification based on state machines





System state

$$
\begin{array}{lll}
\mathit{InstApps} & ::= & \{\mathit{AppId}\} \\
\mathit{GrantedGroups} & ::= & \{\mathit{AppId} \times \{\mathit{PermGrp}\}\} \\
\mathit{AppsPerms} & ::= & \{\mathit{AppId} \times \{\mathit{Perm}\}\} \\
\mathit{CompInsRunning} & ::= & \{\mathit{CompInstance}\} \\
\mathit{OpType} & ::= & \mathit{read} \mid \mathit{write} \mid \mathit{rw} \\
\mathit{DelPPerms} & ::= & \{\mathit{AppId} \times \mathit{ContProv} \times \mathit{Uri} \times \mathit{OpType}\} \\
\mathit{DelTPerms} & ::= & \{\mathit{iComp} \times \mathit{ContProv} \times \mathit{Uri} \times \mathit{OpType}\} \\
\mathit{AppsResCont} & ::= & \{\mathit{AppId} \times \mathit{Res} \times \mathit{ResVal}\} \\
\mathit{SentIntents} & ::= & \{\mathit{iComp} \times \mathit{Intent}\} \\
\mathit{AppsManifest} & ::= & \{\mathit{AppId} \times \mathit{Manifest}\} \\
\mathit{AppsCert} & ::= & \{\mathit{AppId} \times \mathit{Cert}\} \\
\mathit{AppsDefPerms} & ::= & \{\mathit{AppId} \times \{\mathit{Perm}\}\} \\
\mathit{ImageApps} & ::= & \{\mathit{App}\} \\
\mathit{AndroidState} & ::= & \mathit{InstApps} \times \mathit{GrantedGroups} \times \mathit{AppsPerms} \times \mathit{DelTPerms} \times \\
 & & \mathit{CompInsRunning} \times \mathit{DelPPerms} \times \mathit{AppsResCont} \times \mathit{SentIntents} \times \\
 & & \mathit{AppsManifest} \times \mathit{AppsCert} \times \mathit{AppsDefPerms} \times \mathit{ImageApps}
\end{array}
$$

A state is *valid* if application ids are unique ...

Some actions

Some actions

- App **installation** (`install`)
- App **uninstallation** (`uninstall`)

Some actions

- App **installation** (`install`)
- App **uninstallation** (`uninstall`)
- **Start** the execution of a component (`start`)
- **Stop** the execution of a component (`stop`)

Some actions

- App **installation** (`install`)
- App **uninstallation** (`uninstall`)
- **Start** the execution of a component (`start`)
- **Stop** the execution of a component (`stop`)
- **Read** from a content provider (`read`)
- **Write** to a content provider (`write`)

Some actions

- App **installation** (`install`)
- App **uninstallation** (`uninstall`)
- **Start** the execution of a component (`start`)
- **Stop** the execution of a component (`stop`)
- **Read** from a content provider (`read`)
- **Write** to a content provider (`write`)
- **Temporal grant** of permissions (`grantT`)
- **Permanent grant** of permissions (`grantP`)

Some actions

- App **installation** (`install`)
- App **uninstallation** (`uninstall`)
- **Start** the execution of a component (`start`)
- **Stop** the execution of a component (`stop`)
- **Read** from a content provider (`read`)
- **Write** to a content provider (`write`)
- **Temporal grant** of permissions (`grantT`)
- **Permanent grant** of permissions (`grantP`)
- **Revokation** of granted permissions (`revoke`)

Some actions

- App **installation** (`install`)
- App **uninstallation** (`uninstall`)
- **Start** the execution of a component (`start`)
- **Stop** the execution of a component (`stop`)
- **Read** from a content provider (`read`)
- **Write** to a content provider (`write`)
- **Temporal grant** of permissions (`grantT`)
- **Permanent grant** of permissions (`grantP`)
- **Revokation** of granted permissions (`revoke`)
- System API **call** (`call`)
- .....

Some actions

- App **installation** (`install`)
- App **uninstallation** (`uninstall`)
- **Start** the execution of a component (`start`)
- **Stop** the execution of a component (`stop`)
- **Read** from a content provider (`read`)
- **Write** to a content provider (`write`)
- **Temporal grant** of permissions (`grantT`)
- **Permanent grant** of permissions (`grantP`)
- **Revokation** of granted permissions (`revoke`)
- System API **call** (`call`)
- .....

Semantic expressed by the means of pre and postconditions

Semantics

Precondition of action `install`

$$
\begin{aligned}
&Pre(s, \texttt{install}\ app\ m\ c\ lRes) \stackrel{\text{def}}{=} \\
&\quad \neg isAppInstalled(app, s) \ \wedge \ \neg has\_duplicates\_cmp(m) \ \wedge \\
&\quad \forall c : \text{Comp}, c \in cmp(m) \ \rightarrow \ c \notin cmpInState(s) \ \wedge \\
&\quad \neg hasDuplicates\_perm(m) \ \wedge \ authPerms(m, s) \ \wedge \\
&\quad \forall c : \text{Comp}, c \in cmp(m) \ \rightarrow \\
&\quad cmpDeclareIntentFilterCorrectly(c)
\end{aligned}
$$

Semantics

Precondition of action `install`

$$
\begin{aligned}
&\mathit{Pre}(s, \texttt{install}\ \mathit{app}\ m\ c\ \mathit{lRes}) \stackrel{\text{def}}{=} \\
&\quad \neg \mathit{isAppInstalled}(\mathit{app}, s)\ \wedge\ \neg \mathit{has\_duplicates\_cmp}(m)\ \wedge \\
&\quad \forall c : \mathrm{Comp}, c \in \mathit{cmp}(m)\ \rightarrow\ c \notin \mathit{cmpInState}(s)\ \wedge \\
&\quad \neg \mathit{hasDuplicates\_perm}(m)\ \wedge\ \mathit{authPerms}(m, s)\ \wedge \\
&\quad \forall c : \mathrm{Comp}, c \in \mathit{cmp}(m)\ \rightarrow \\
&\quad \mathit{cmpDeclareIntentFilterCorrectly}(c)
\end{aligned}
$$

Postcondition of action `install`

$$
\begin{aligned}
&\mathit{Post}(s, \texttt{install}\ \mathit{app}\ m\ c\ \mathit{lRes}, s') \stackrel{\text{def}}{=} \\
&\quad \mathit{addManifest}(m, \mathit{app}, s, s')\ \wedge\ \mathit{addCert}(c, \mathit{app}, s, s')\ \wedge \\
&\quad \mathit{addDefPerms}(\mathit{app}, m, s, s')\ \wedge\ \mathit{addApp}(\mathit{app}, s, s')\ \wedge \\
&\quad \mathit{addRes}(\mathit{app}, \mathit{lRes}, s, s')\ \wedge\ \mathit{initializePermLists}(\mathit{app}, s, s')\ \wedge \\
&\quad \mathit{sameOtherFields\_install}(s, s')
\end{aligned}
$$

Actions execution

The state transition is represented with the relation noted $\hookrightarrow$:

$$\frac{validState\ s \qquad Pre\ s\ act \qquad Post\ s\ act\ s'}{s \xhookrightarrow{act,ok} s'}$$

$$\frac{validState\ s \qquad ErrorMsg\ s\ act\ err}{s \xhookrightarrow{act,err} s}$$

Actions execution

The state transition is represented with the relation noted $\hookrightarrow$:

$$\frac{validState\ s \qquad Pre\ s\ act \qquad Post\ s\ act\ s'}{s \xhookrightarrow{act,ok} s'}$$

$$\frac{validState\ s \qquad ErrorMsg\ s\ act\ err}{s \xhookrightarrow{act,err} s}$$

$$\boxed{s_i \xhookrightarrow{act,r} s_{i+1}}$$

Actions execution

The state transition is represented with the relation noted $\hookrightarrow$:

$$\frac{validState\ s \qquad Pre\ s\ act \qquad Post\ s\ act\ s'}{s \xhookrightarrow{act,ok} s'}$$

$$\frac{validState\ s \qquad ErrorMsg\ s\ act\ err}{s \xhookrightarrow{act,err} s}$$

$$\boxed{s_i \xhookrightarrow{act,r} s_{i+1}}$$

$$s_0 \xhookrightarrow{act_1,r_1} s_1 \xhookrightarrow{act_2,r_2} \cdots \xhookrightarrow{act_{n-1},r_{n-1}} s_{n-1} \xhookrightarrow{act_n,r_n} s_n$$

Actions execution

The state transition is represented with the relation noted $\hookrightarrow$:

$$\frac{validState\ s \qquad Pre\ s\ act \qquad Post\ s\ act\ s'}{s \xhookrightarrow{act,ok} s'}$$

$$\frac{validState\ s \qquad ErrorMsg\ s\ act\ err}{s \xhookrightarrow{act,err} s}$$

$$\boxed{s_i \xhookrightarrow{act,r} s_{i+1}}$$

$$s_0 \xhookrightarrow{act_1,r_1} s_1 \xhookrightarrow{act_2,r_2} \cdots \xhookrightarrow{act_{n-1},r_{n-1}} s_{n-1} \xhookrightarrow{act_n,r_n} s_n$$

Lemma (Validity is invariant)

$\forall\ (s\ s' : AndroidST)(a : Action)(r : Response),$
$s \xhookrightarrow{a,r} s' \rightarrow valid\_state(s')$

Summary

Security properties

Basic properties

Eg: state validity invariance

Security properties

Basic properties

Eg: state validity invariance

Desired properties

Eg: Principle of least privilege

Security properties

Basic properties

Eg: state validity invariance

Desired properties

Eg: Principle of least privilege

Undesired properties

Eg: privilege escalation

Security properties

Basic properties

Eg: state validity invariance

Desired properties

Eg: Principle of least privilege

Undesired properties

Eg: privilege escalation

Mitigating properties

Eg: mitigate *eavesdropping*, *intent spoofing*

Properties concerning the security model of Android 6

- No fine control over grouped permissions:
  Android's permission system is not granular enough for granting a proper subset of the set of permissions that belong to a group.

Properties concerning the security model of Android 6

- No fine control over grouped permissions:
  Android's permission system is not granular enough for granting a proper subset of the set of permissions that belong to a group.
- Implicit individual permission granting:
  In contrast to what happened in previous Android versions, Applications may obtain privileges that were never granted to it.

Properties concerning the security model of Android 6

- No fine control over grouped permissions:
  Android's permission system is not granular enough for granting a proper subset of the set of permissions that belong to a group.
- Implicit individual permission granting:
  In contrast to what happened in previous Android versions, Applications may obtain privileges that were never granted to it.
- Internet access implicitly and irrevocably allowed:
  If the execution of an Android API call only requires permissions of level normal, it is enough for an application to list them as used on its manifest file to be allowed to perform such call.

Reasoning over the certified reference validation mechanism

- Dangerous permissions must be explicitly granted:
  A non-grouped dangerous permission can only be explicitly granted to an application.

Reasoning over the certified reference validation mechanism

- Dangerous permissions must be explicitly granted:
  A non-grouped dangerous permission can only be explicitly granted to an application.
- Revoked permissions must be regranted:
  If an application used to have a permission that was later revoked, only regranting it will allow the application to have it again.

Reasoning over the certified reference validation mechanism

- Dangerous permissions must be explicitly granted:
  A non-grouped dangerous permission can only be explicitly granted to an application.
- Revoked permissions must be regranted:
  If an application used to have a permission that was later revoked, only regranting it will allow the application to have it again.
- The right to start an external component is revocable:
  A running component may have the right of starting another one on a certain state, but may not be able to do so at a later time.

Reasoning over the certified reference validation mechanism

- Dangerous permissions must be explicitly granted:
  A non-grouped dangerous permission can only be explicitly granted to an application.
- Revoked permissions must be regranted:
  If an application used to have a permission that was later revoked, only regranting it will allow the application to have it again.
- The right to start an external component is revocable:
  A running component may have the right of starting another one on a certain state, but may not be able to do so at a later time.
- Delegated permissions are not recursively revoked:
  In Android 6 if a permission p is revoked for an application app not necessarily shall be revoked for the applications to which app delegated p.

Mitigating properties: Eavesdropping

- Unauthorized information monitoring.
- In Android: when sensible information is sent as part of public broadcast messages.

Mitigating properties: Eavesdropping

- Unauthorized information monitoring.
- In Android: when sensible information is sent as part of public broadcast messages.

Lema

If a component c which belongs to applicaiton a sends an intent of type broadcast protected by a permission of type *signature* or *signature or system*, then if a non-system application a' wasn't signed with the same permission as a, then it can't receive it.

Development of a certified implementation

Certified implementation

- Implementation of Coq functions for each of the specified actions.

Development of a certified implementation

Certified implementation

- Implementation of Coq functions for each of the specified actions.
- Proof of soundness: functions implement the execution relation for each action.

Development of a certified implementation

Certified implementation

- Implementation of Coq functions for each of the specified actions.
- Proof of soundness: functions implement the execution relation for each action.
- Extraction of a Haskell program (command dispatcher) correct by construction.

The implementation

The *step* function and execution of `install` action

Definition $step(s, a)$ $:=$
  match a **with**
    $\mid \ldots \rightarrow \ldots$
    $\mid$ `install` $app\ m\ c\ lRes \rightarrow install\_safe(app, m, c, lRes, s)$
    $\mid \ldots \rightarrow \ldots$
  end.

Definition $install\_safe(app, m, c, lRes, s) : Result :=$
  match $install\_pre(app, m, c, lRes, s)$ **with**
    $\mid Some\ ec \rightarrow \{error(ec), s\}$
    $\mid None \rightarrow \{ok, install\_post(app, m, c, lRes, s)\}$
  end.

Soundness

Theorem (Soundness of Android security system implementation)

$$\forall\ (s : AndroidST)\ (a : Action), valid\_state(s) \rightarrow\ s \xrightarrow{a, step(s,a).resp} step(s, a).st$$

Soundness

Theorem (Soundness of Android security system implementation)

$$\forall\ (s : AndroidST)\ (a : Action), valid\_state(s) \rightarrow\ s \xrightarrow{a, step(s,a).resp} step(s, a).st$$

Lemma (Soundness of valid execution)

$$\forall\ (s : AndroidST)\ (a : Action), valid\_state(s) \rightarrow\ Pre(s, a)\ \rightarrow$$
$$s \xrightarrow{a, ok} step(s, a).st\ \wedge\ step(s, a).resp = ok$$

Lemma (Soundness of error execution)

$$\forall\ (s : AndroidST)\ (a : Action), valid\_state(s) \rightarrow\ \neg Pre(s, a) \rightarrow$$
$$\exists\ (ec : ErrorCode),\ step(s, a).st = s \wedge step(s, a).resp = error(ec) \wedge$$
$$ErrorMsg(s, a, ec)$$

Use of certified implementation

Haskell code

Applying the program extraction mechanism provided by `Coq` we have derived a certified `Haskell` prototype of the reference validation mechanism (command dispatcher).

Use of certified implementation

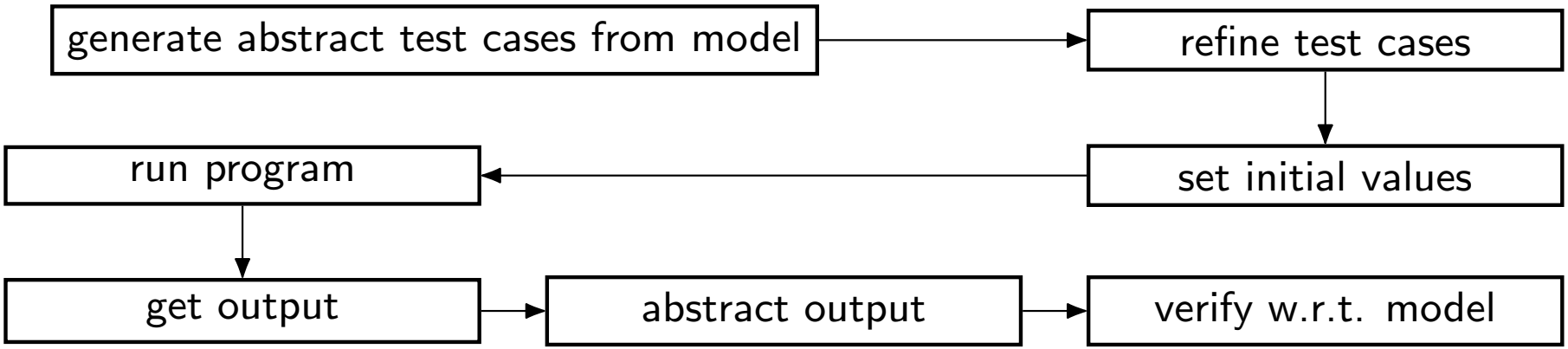
Haskell code

Applying the program extraction mechanism provided by `Coq` we have derived a certified `Haskell` prototype of the reference validation mechanism (command dispatcher).

Oracle

Automatic generation of test cases for a real Android system based on the model using the techniques of *model-based testing*, incorporating the use of the *oracle*.

A model based testing process



Questions?

Questions?

Thank you!

Thank you!