SeMA: A Design Methodology for Building Secure Android Apps

Joydeep Mitra  Venkatesh-Prasad Ranganath
Department of Computer Science
Kansas State University, USA

International Workshop on Advances in Mobile App Analysis (A-Mobile 2019)
San Diego, USA
November 11, 2019
Context

• Storyboards are used to capture the UI+UX of an app
• Security is crucial to the UX of a mobile app
• Current UX design process of an app is limited in terms of security reasoning
• Can reasoning about security be baked into the design process of an app?
What is mobile app storyboarding?

A storyboard is a sequence of images that serves as a specification of the user observed behavior in terms of screens and transitions between screens.
Limitations of Current Mobile App Storyboarding Approaches/Tools

- Inability to specify non-UI behavior
- Inability to enable collaboration between app designers and app developers
- Inability to reason about non-functional properties such as security

We propose a methodology (SeMA) based on storyboarding to enable the specification and verification of security properties of Android apps at design time.
Proposed Methodology

• App designer specifies the app’s storyboard
• App designer and developer collaborate to iteratively refine the storyboard by adding non-UI related behavior (e.g., constraints on when transitions will be triggered)
• After every iteration verify if the storyboard satisfies pre-defined security properties
• Finally, generate property preserving code
• Developer extends generated code with business logic
Illustrative Example: Initial Storyboard

- **Messenger**
  - Send
  - Add

- **Contacts**
  - Phone
  - Add

- **SaveStatus**
  - Status = “Saved”

- **MsgStatus**
  - Status = “MsgSent”
Illustrative Example: Storyboard with UI Constraints

- **Messenger**
  - Send
  - Add

- **Contacts**
  - Phone
  - Add

- **SaveStatus**
  - Status = "Saved"

- **MsgStatus**
  - Status = "MsgSent"

Actions:
- Add.click
- Save.click
- Send.click
Illustrative Example: **Storyboard with Non-UI Constraints**

- **Messenger**
  - Send
  - Add

- **Contacts**
  - Phone
  - Add

- **SaveStatus**
  - Status = “Saved”

**Flow:***
- Add.click from **Messenger** to **Contacts**
- Save.click and EXT_STORE.write (”contacts.txt”, Phone.val)
- Send.click and SMS.send(“Please help”, EXT_STORE.read(“contacts.txt”))
- Status = “MsgSent” from **MsgStatus** to **SaveStatus**
Illustrative Example: Security Analysis of the Storyboard

- **Messenger**
  - Send
  - Add

- **Contacts**
  - Phone
    - Add
  - Save.click and EXT_STORE.write(“contacts.txt”, Phone.val)

- **SaveStatus**
  - Status = “Saved”

- **MsgStatus**
  - Status = “MsgSent”

- **Sensitive data leak**
- **Data injection**
Realizing SeMA for Android [PoC/Ongoing]

- Extend existing Storyboard tools (e.g. Navigation graphs) to enable the specification of non-UI behavior
- Define security properties based on known vulnerabilities
- Build the analysis framework to verify pre-defined security properties on the storyboard
- Build the code generation algorithm to translate storyboards to Java/Kotlin
- Enable the methodology in Android Studio
Realizing SeMA for Android Platform

Use JetPack’s Navigation Graph for storyboarding
Realizing SeMA for Android Platform

```xml
<navigation xmlns:android="http://schemas.android.com/apk/res-auto"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    app:startDestination="@id/messngerFrag"
    android:id="@+id/main_nav">
    <fragment android:id="@+id/messngerFrag">
        <fragment android:id="@+id/contactFrag"
            tools:layout="@layout/contacts">
            <widget wid="@id/phoneNum"/>
            <action android:id="@+id/goToStatus"
                app:destination="@id/statusFrag"
                widgetOn="@id/save"
                gesture="click"/>
        </fragment>
    </fragment>
    <fragment android:id="@+id/statusFrag"
        tools:layout="@layout/savestatus"/>
</navigation>
```
Realizing SeMA for Android Platform

```xml
<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
            xmlns:app="http://schemas.android.com/apk/res-auto"
            xmlns:tools="http://schemas.android.com/tools"
            app:startDestination="@id/messengerFrag"
            android:id="@+id/main_nav">
    <fragment android:id="@+id/messengerFrag"
              tools:layout="@layout/messenger">
        <action android:id="@+id/goToContacts"
                app:destination="@id/contactFrag"
                widgetOn="@id/add"
                gesture="click"/>
        <action android:id="@+id/goToMsg"
                app:destination="@id/msgFrag"
                widgetOn="@id/send"
                gesture="click"/>
    </fragment>
</navigation>
```

Extend navigation graph with **UI constraints**
Realizing SeMA for Android Platform

Extending navigation graph with non-UI constraints
Realizing SeMA for Android Platform

```
<constraint>
  <fun android:name="sendTextMsg" resource="SMS" method="send">
    <parm arg="Help!"/>
    <parm arg="fun/getSavedContacts">
      <fun android:name="getSavedContacts" resource="EXT_STORE" method="get">
        <parm arg="contacts.txt"/>
      </fun>
    </parm>
  </fun>
</constraint>
```

ps/amobile/app/src/main/res/navigation/main_nav.xml:21: Error: parameter used in sendTextMsg might be malicious [DataInjectionScreen] ="sendTextMsg" resource="SMS" method="send"

Extend navigation graph with Security Analysis
Realizing SeMA for Android Platform

Extend navigation graph with Security Analysis
Challenges

• Enabling storyboards to capture non-UI behavioral constraints in a non-intrusive way  [PoC/Ongoing]
• Making the analysis context-aware  [Future Work]
• Checking richer security properties (e.g. temporality)  [Future Work]
• Ensuring preservation of security properties  [Future Work]
Takeaways

A design methodology to enable automated reasoning and verification of security properties of Android apps

• Builds on storyboarding
• Tackles different classes of security properties
• Can be realized with existing Android app development tools
• Facilitates automated reasoning and verification