How Android’s UI Security is Undermined by Accessibility

Anatoli Kalysch (Speaker), Davide Bove, and Tilo Müller

November 30, 2018

Friedrich-Alexander-Universität Erlangen-Nürnberg
Department of Computer Science
IT Security Infrastructures Lab
Software Security Research Group
whoami

- PhD Student at FAU
- Research Focus on Mobile Security and Program Analysis
  - Malware Analysis
  - App Analysis Environments
  - UI Security
  - App Reverse Engineering
- Also tutoring courses in Reverse Engineering and IT Forensics
Outline

Of Androids and their A11y Services

A11y Capabilities in Context
  Working as Intended?
  Probably not Working as Intended

A11y as an Attack Vector
  Are App Developers Aware?
  Countering A11y-based and UI redressing Attacks

Take Aways

Demo
Outline

Of Androids and their A11y Services

A11y Capabilities in Context
  Working as Intended?
  Probably not Working as Intended

A11y as an Attack Vector
  Are App Developers Aware?
  Countering A11y-based and UI redressing Attacks

Take Aways

Demo
UI Security

- Android features strong app separation concepts, e.g.,
  - Sandboxing
  - Binder-assisted window management
  - Permission-based concepts

- UI offers a user-centered perspective on data and apps

- Flaws in the UI allow to mitigate these separation concepts and leak data or overtake the UI [3, 2].
UI Security in an A11y Context

- A11y services have a major impact on UI security due to them:
  - being notified about every UI change, and
  - being able to take action for the user

- Actions can be automated and performed a lot faster:
  - e.g., granting permissions and generating user input [2],
  - starting screen recordings or even
  - enabling a new IME and registering it as the new default

- Since many security measures rely on user confirmations this creates immense potential for abuse
Secure Flag
Secure Flag
User Dialog for Screen Recordings

**DU Recorder** will start capturing everything that's displayed on your screen.

☐ Don't show again

CANCEL START NOW
Register your own IME

![Image of Android keyboard settings with options for Gboard, Google voice typing, English (United States), Smoke + Mirrors]
A11y TextView Sniffing

- A11y services are notified about UI changes, e.g., when the user generates input such as text
- Every input character generates several a11y events which can be problematic for password / sensitive information fields
- Which is why all contents of password fields are censored with the ‘dot’ character to not leak any information ...
- ... except the last character which is visible for 2.5 seconds after entry!
Missing Settings Synchronization

• Using Unstructured Supplementary Service Data (USSD) codes allows you to change phone, network, and carrier options

• Some of these options are hidden to the normal user others can be interacted with from the settings app

• Changing options through USSD codes does not affect the values displayed in the setting app

• An a11y service can actually interact with settings apps and input USSD codes through the phone application
Misleading Capabilities

Use Smoke + Mirrors?

Smoke + Mirrors needs to:

- **Observe your actions**
  Receive notifications when you’re interacting with an app.

- **Retrieve window content**
  Inspect the content of a window you’re interacting with.
Misleading Capabilities

Use TalkBack?

TalkBack needs to:

- **Observe your actions**
  Receive notifications when you’re interacting with an app.

- **Retrieve window content**
  Inspect the content of a window you’re interacting with.

- **Turn on Explore by Touch**
  Tapped items will be spoken aloud and the screen can be explored using gestures.

- **Observe text you type**
  Includes personal data such as credit card numbers and passwords.

- **Control display magnification**
  Control the display’s zoom level and positioning.

- **Fingerprint gestures**
  Can capture gestures performed on the device
Use Smoke + Mirrors?

CANCEL  OK
Outline

Of Androids and their A11y Services

A11y Capabilities in Context
  Working as Intended?
  Probably not Working as Intended

A11y as an Attack Vector
  Are App Developers Aware?
  Countering A11y-based and UI redressing Attacks

Take Aways

Demo
Distribution of A11y Services in App Stores

Figure: Number of a11y services per app category.
## App Logins Under Attack

<table>
<thead>
<tr>
<th>Category</th>
<th>Number of apps with a login</th>
<th>Percentage of logins vulnerable against</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>A11y Events</td>
</tr>
<tr>
<td>Business</td>
<td>116</td>
<td>100%</td>
</tr>
<tr>
<td>Communication</td>
<td>47</td>
<td>100%</td>
</tr>
<tr>
<td>Dating</td>
<td>63</td>
<td>100%</td>
</tr>
<tr>
<td>Entertainment</td>
<td>58</td>
<td>100%</td>
</tr>
<tr>
<td>Finance</td>
<td>172</td>
<td>84.9%</td>
</tr>
<tr>
<td>Games</td>
<td>104</td>
<td>95.2%</td>
</tr>
<tr>
<td>Health</td>
<td>57</td>
<td>98.3%</td>
</tr>
<tr>
<td>Shopping</td>
<td>42</td>
<td>95.2%</td>
</tr>
<tr>
<td>Social</td>
<td>99</td>
<td>100%</td>
</tr>
<tr>
<td>Travel</td>
<td>45</td>
<td>97.7%</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td><strong>803</strong></td>
<td><strong>95.6%</strong></td>
</tr>
</tbody>
</table>

**Table:** Out of 1100 apps 803 had a login screen, most of them being vulnerable.
Figure: Employed security mechanisms per category.
Countermeasures

- A11y event filtering or sanitization
- Behavioral Listeners
- Window Punching
- In-App Keyboards
- Fingerprint API
A11y Event Filtering

- public boolean dispatchPopulateAccessibilityEvent(AccessibilityEvent event)
- public void onPopulateAccessibilityEvent(AccessibilityEvent event)
- public void onInitializeAccessibilityEvent(AccessibilityEvent event)
- public void onInitializeAccessibilityNodeInfo(AccessibilityNodeInfo accessibilityNodeInfo)
Behavioral Listeners

- We can abuse the limitations of a11y services as a countermeasure
- This however excludes people using them from using the app
- A11y services can only simulate ‘click’ events
  - No TouchDown or TouchUp event generated
  - Switching to the corresponding Listeners
Window Punching

OverlayDetection

Email

Password (stealing)

SIGN IN OR REGISTER
Vetting Attacks against Countermeasures

<table>
<thead>
<tr>
<th>Attack</th>
<th>Vulnerable Android Versions</th>
<th>Possible Countermeasures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.0</td>
<td>7.0</td>
</tr>
<tr>
<td>A11y Event Sniffing</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>A11y Screen Recording</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>A11y-enabled Malicious IME</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>A11y-based Ad Hijacking [2]</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Overlay and a11y assisted password stealing [2]</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Keyboard App Hijacking [2]</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Full App Passthrough / Clickable Overlays [?]</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial App Clickable Overlays [?]</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Keystroke Inference [2]</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table: A11y and overlay-based attacks presented here and in previous work by different authors.
Take Aways

- A11y Services can sniff passwords upon activation
  - If they are additionally allowed to take actions for the user even silent IME installs and screen recordings become possible
  - Device tampering is possible as well

- Application developers are not aware of this threat
  - 99.25% of apps with a login on Google Play were vulnerable to credential leakages
  - Currently deployed protection mechanisms do not offer adequate protection

- A11y Services and UI attack scenarios are a viable threat as recent ‘advances’ in malware have shown [4][1]
Disclosure Process

- Vulnerabilities and bugs reported through bug and security reports
  - 7 reports all together
  - Highest rating among them is ‘Low’ most were rated ‘NSBC’
    (Non-Security Bulletin Class -> probably won’t get a fix anytime soon)

- Reports were submitted as between December 2017 and March 2018

- Selected app developers from the categories finance and healthcare were notified about the bugs in their login fields in August

- Notification of all developers planned until the end of 2018
Thank you.

Questions?

Contact
Website: https://www.cs1.tf.fau.de/person/anatoli-kalysch/
Email: anatoli.kalysch@fau.de
Twitter: @aka_kaly

PoC snippets and selected PoC countermeasure projects available (starting December 1st) at:

https://github.com/anatolikalysch/roots_a11y
References


