ANALYZING INTER-APPLICATION COMMUNICATION IN ANDROID

Erika Chin
Adrienne Porter Felt
Kate Greenwood
David Wagner
UC Berkeley
INTER-APPLICATION COMMUNICATION

- Yelp App
- Maps App
- Dialer App
- Malicious App

Eavesdropping Attacks
Injection Attacks
ORGANIZATION

- Android communication model
- Security analysis of Android
- ComDroid
- Analysis of third-party applications
- Recommendations
ANDROID OVERVIEW

- **Intents** = Android IPC
- Applications are divided into **components**
- Intents can be sent between components
- Intents can be used for intra- and inter-application communication
Explicit Intents

Only the specified destination receives this message
Implicit Intents

Yelp

Implicit Intent
Action: VIEW

Handles Action: VIEW

Map App

Handles Action: DISPLAYTIME

Clock App
Implicit Intent

Implicit Intent Action: VIEW

Handles Action: VIEW

Yelp

Map App

Handles Action: VIEW

Browser App
SECURITY ANALYSIS OF ANDROID
COMMON DEVELOPER PATTERN: UNIQUE ACTION STRINGS

IMDb App

Handles Actions:
willUpdateShowtimes,
showtimesNoLocationError

Showtime Search

Implicit Intent Action:
willUpdateShowtimes

Results UI
Date & Location

Thursday, June 23

Current Location

New This Week

Bad Teacher (2011)
Rated R, 1 hr 32 mins, 6.3/10
Showtimes from Century Richmond
Hilltop 16, Century San Francisco Centre
9 and XD, and 1 other...

Cars 2 (2011)
Rated G, 1 hr 53 mins, 6.9/10
Showtimes from AMC Bay Street 16, AMC
Bay Street 16, and 2 others
COMMON DEVELOPER PATTERN: UNIQUE ACTION STRINGS

IMDb App

Handles Actions:
willUpdateShowtimes,
showtimesNoLocationError

Showtime Search

Implicit Intent Action:
willUpdateShowtimes

Results UI
ATTACK #1: EAVESDROPPING

IMDb App

Showtime Search

Implicit Intent Action: willUpdateShowtimes

Eavesdropping App

Handles Action: willUpdateShowtimes, showtimesNoLocationError

Sending Implicit Intents makes communication public
ATTACK #2: INTENT SPOOFING

Malicious Injection App

Malicious Component

Action: showtimesNoLocationError

IMDb App

Handles Action: willUpdateShowtimes, showtimesNoLocationError

Results UI

Receiving Implicit Intents makes the component public
Typical case

Attack case
ATTACK #3: MAN IN THE MIDDLE

**IMDb App**

- Handles Action: `willUpdateShowtimes`, `showtimesNoLocation` Error
- Action: `willUpdateShowtimes`

**Man-in-the-Middle App**

- Handles Action: `willUpdateShowtimes`, `showtimesNoLocationError`
- Action: `showtimesNoLocationError`
ATTACK #4: SYSTEM INTENT SPOOFING

- Background – System Broadcast
  - Event notifications sent by the system
  - Some can only be sent by the system

- Receivers become accessible to all applications when listening for system broadcast
**SYSTEM BROADCAST**

- **System Notifier**
  - **Action:** *BootCompleted*

- **App 1**
  - Component
  - **Handles Action:** *BootCompleted*

- **App 2**
  - Component
  - **Handles Action:** *BootCompleted*

- **App 3**
  - Component
  - **Handles Action:** *BootCompleted*
**System Intent Spoofing: Failed Attack**

- **Malicious App**
  - Malicious Component
  - Action: *BootCompleted*

- **App 1**
  - Handles Action: *BootCompleted*
**SYSTEM INTENT SPOOFING: SUCCESSFUL ATTACK**

**Malicious App**

- Malicious Component
- To: App1.Component

**App 1**

- Handles Action: *BootCompleted*
- Component
REAL WORLD EXAMPLE: ICE APP

- ICE App: Allows doctors access to medical information on phones

- Contains a component that listens for the `BootCompleted` system broadcast

- On receipt of the Intent, it exits the application and locks the screen
REAL WORLD EXAMPLE: ICE
ComDroid analyzes applications to detect Intent-based attack surfaces
EVALUATION

- Manually verified ComDroid’s warnings for 20 applications

- 60% of applications examined have at least 1 exploitable IPC vulnerability

<table>
<thead>
<tr>
<th>Type</th>
<th># of Warnings</th>
<th># of Apps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severe Vulnerability</td>
<td>34</td>
<td>12</td>
</tr>
<tr>
<td>Bad Practice</td>
<td>16</td>
<td>6</td>
</tr>
<tr>
<td>Spurious Warning</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>
RECOMMENDATIONS

- Treat inter- and intra-application communication as different cases

- Prevent public internal communication
  - 21% of severe vulnerabilities
  - 63% of bugs due to bad practice

- Verify system broadcasts
  - 6% of severe vulnerabilities
  - 13% of bugs due to bad practice

- Can be fixed by either developers or platform
RELATED WORK

- Enck et al. – introduces information leakage through Broadcast Intents and information injection into Receivers

- Burns – discusses other common developers’ errors
**CONCLUSION**

- Applications may be vulnerable to other applications through Android Intent communication

- Many developers misuse Intents or do not realize the consequences of their program design

- 60% of applications examined had at least 1 vulnerability

- ComDroid tool to be publically accessible soon at www.comdroid.org
Thank you!

Any questions?