Systematic Detection of Capability Leaks in Stock Android Smartphones

Michael Grace  Yajin Zhou  Zhi Wang  Xuxian Jiang

North Carolina State University
“I have always wished that my computer would be as easy to use as my telephone. My wish has come true. I no longer know how to use my telephone.”

– Bjarne Stroustrup (designer of C++)
Phones: the PCs of the Future?

• Smartphone shipments increased 42% between 3Q 2010 and 3Q 2011 (Gartner, 11/15/2011)

• More smartphones shipping than personal computers (IDC, 2/7/2011)
  • New markets: first computer = smartphone
Smartphone ≠ Handheld PC

- Unique abilities specific to the form factor
  - Many sensors: “context-aware”
  - Dialup ➔ always on ➔ always with you

- Resource constrained

- Different vendor relationships and primacy
Related Work

• Problems with Permissions
  • e.g., Kirin [Enck et al., CCS ‘09], Soundcomber [Schlegel et al., NDSS ‘11], Stowaway [Felt et al., CCS ‘11], Guess Who’s Texting You [Schrittwieser et al., NDSS ‘12]...

• Information Leak Detection
  • e.g., PiOS [Egele et al., NDSS ‘11], TaintDroid [Enck et al., OSDI ‘10]...

• Phone Defenses
  • e.g., MockDroid [Beresford et al., HotMobile ‘11], TISSA [Zhou et al., TRUST ‘11], AppFence [Hornyack et al., CCS ‘11], Permission Re-Delegation [Felt et al., USENIX Security ‘11], QUIRE [Dietz et al., USENIX Security ‘11], XManDroid [Bugiel et al., NDSS ‘12], MoCFI [Davi et al., NDSS ‘12]...

• Market Issues
  • e.g., DroidMOSS [Zhou et al., CODASPY ‘12], DroidRanger [Zhou et al., NDSS ‘12]...
Firmware and Fragmentation

• A conspicuous gap in the body of work!

• Not like on desktops, or other smartphone platforms

• Research Goal: Determine the impact firmware customizations have on security and privacy
Android Capabilities

- Platform defines some APIs
- APIs *may* require capabilities (called permissions)
- Applications can define APIs the same way
- What happens when an application defines a new API based on a restricted old one?
  - That’s up to the author!
Capability Leaks

- **Capability Leak**: A situation where an app can gain access to a restricted API without requesting proper permission.

- **Explicit Capability Leak**: Broadening access to a restricted API by exposing it via another API.

- **Implicit Capability Leak**: Inheriting permissions from other applications.
Explicit Capability Leaks

Outside Caller (no permissions)

Leaking API (needs no permissions, has permission P)

Restricted API (needs permission P)
Detecting Capability Leaks

- Android SDK gives us no tools!

- Function composition
  - Capability leak: \( g(x) = f(x) + \text{some other stuff} \)

- Intuitive algorithm:
  1. Find interesting (dangerous) APIs \((f(x))\)
  2. Find new API definitions \((g(x))\)
  3. Link them!
System Overview

- Preloaded Apps
- Framework Classes
- Possible Path Identification
- Infeasible Path Pruning
- Leak Report
- Android Framework Knowledge

Woodpecker
Possible Path Identification

1. Construct a control-flow graph

2. Find all paths from an IPC entry point to an API of interest
Possible Path Identification: Challenges

- Object references
  - Class hierarchy used to conservatively resolve references

- Extensive use of callbacks
  - Use framework knowledge to stitch together callbacks
Infeasible Path Pruning

- *Many* potential paths exist
  - Most are either impossible or uninteresting

- Must prune these uninteresting paths
  - Branch conditions need an understanding of program data-flow
  - Explicit permission checks are “infeasible paths”

- Our approach: *Symbolic Path Simulation*
Symbolic Path Simulation

If \( X = 0 \)

\[ X = 0 \]

Then

\[ X = 0 \]

Else

\[ X \neq 0 \]
Implementation

• Based on the baksmali decompiler (1.2.6)

• Covers 13 permissions, controlling:
  • Phone information
  • Location API
  • Phone dialing
  • Sending text messages
  • Camera/microphone
  • Rebooting/shutting down the device
  • Installing/removing apps
  • Factory reset
Evaluation
Explicit Capability Leaks Found

<table>
<thead>
<tr>
<th>Permission</th>
<th>Legend</th>
<th>HTC EVO 4G</th>
<th>Wildfire S</th>
<th>Motorola</th>
<th>Samsung</th>
<th>Google</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coarse Location</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Fine Location</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Call Phone</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Call Privileged</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Camera</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Delete Packages</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Install Packages</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Master Clear</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Read Phone State</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Reboot</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Record Audio</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Send SMS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Shutdown</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
Demo
## Implicit Capability Leaks Found

<table>
<thead>
<tr>
<th>Permission</th>
<th>Legend</th>
<th>HTC EVO 4G</th>
<th>Wildfire S</th>
<th>Motorola DROID</th>
<th>Motorola DROID X</th>
<th>Samsung Epic 4G</th>
<th>Google Nexus One</th>
<th>Google Nexus S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coarse Location</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fine Location</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call Phone</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call Privileged</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Camera</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delete Packages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Install Packages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Master Clear</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read Phone State</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reboot</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Record Audio</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Send SMS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shutdown</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Performance Measurement

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Model</th>
<th>Time</th>
<th># Apps</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTC</td>
<td>Legend</td>
<td>3366.63s</td>
<td>125</td>
</tr>
<tr>
<td></td>
<td>EVO 4G</td>
<td>4175.03s</td>
<td>160</td>
</tr>
<tr>
<td></td>
<td>Wildfire S</td>
<td>3894.37s</td>
<td>144</td>
</tr>
<tr>
<td>Motorola</td>
<td>DROID</td>
<td>2138.38s</td>
<td>76</td>
</tr>
<tr>
<td></td>
<td>DROID X</td>
<td>3311.94s</td>
<td>161</td>
</tr>
<tr>
<td>Samsung</td>
<td>Epic 4G</td>
<td>3732.56s</td>
<td>138</td>
</tr>
<tr>
<td>Google</td>
<td>Nexus One</td>
<td>2059.47s</td>
<td>76</td>
</tr>
<tr>
<td></td>
<td>Nexus S</td>
<td>1815.71s</td>
<td>72</td>
</tr>
</tbody>
</table>
Discussion

• Accuracy
  • False negatives: native code, undocumented extensions
  • False positives: conservative analysis

• Threads and Time
  • Instruction interleaving, shared state
  • Example: callback handling
Conclusions

• Capability leaks present a tangible threat to security and privacy on existing Android smartphones

• We present a system, Woodpecker, to detect these capability leaks
Thank you!
Implicit Capability Leaks

- Applications don’t have permissions, *user identifiers (UIDs)* do.

---

**App A**

- **Manifest**
  - Permission MASTER_CLEAR
  - sharedUserId X

- **Code**
  - masterClear()