AirBag: Boosting Smartphone Resistance to Malware Infection

Chiachih Wu, Yajin Zhou
Hunal Patel, Xuxian Jiang
NC State University

Zhenkai Liang
National University of Singapore

Presented By:
William Hollingsworth
Problem Definition

http://www.theguardian.com/technology/2012/may/16/android-smartphone-market-50-percent
Problem Definition

- Mobile malware increasingly common
- Three solution types
  - Server-side
    - Analyze apps in Marketplace and as they come in
Problem Definition

- Mobile malware increasingly common
- Three solution types
  - Client-side
    - Traditional Anti-Malware programs
    - Repackage apps to enforce access control
    - Extend permissions systems
Problem Definition

- Mobile malware increasingly common
- Three solution types
  - Virtualization-based
    - Multiple virtual phones
    - Multi-user support
Problem Definition

- Want: app-centric, lightweight virtualization
- AirBag:
  - Untrusted apps disallowed from direct interaction with Android
  - App Isolation Runtime (AIR)
    - Incognito
    - Profiling
    - Normal
Outline

- Problem Definition
- System Design
  - Threat Model
  - Enabling Techniques
- Implementation (AirBag)
- Evaluation
- Limitations
- Conclusion
System Design: Current

- Trusted App
- Native Android Runtime
- Linux OS Kernel (w/ Android Extension)
- Untrusted App

User Kernel
System Design: Threat Model

- Users will install malicious applications
  - Not necessarily intended
- Assume a trusted phone OS (TCB)
System Design: Goals

1. Reliably isolate untrusted apps
   a) **Challenge**: Open design of Android
2. Provide a safer user experience
3. Incur minimal overhead
System Design: Proposed

User

Kernel

Native Android Runtime

Linux OS Kernel (w/ Android Extention)

Trusted App

Trusted App

Untrusted App

Context-Aware Device Virtualization

Decoupled AIR

AirBag
Outline

- Problem Definition
- System Design
  - Threat Model
  - Enabling Techniques
- Implementation (AirBag)
- Evaluation
- Limitations
- Conclusion
Enabling Techniques

- Decoupled App Isolation Runtime (AIR)
- Namespace/Filesysteem Isolation
- Context-Aware Device Virtualization
Enabling Techniques

- **Decoupled App Isolation Runtime (AIR)**
  - Normally all apps share the same runtime
  - AIR provides an independent runtime
    - Separate implementation of Android framework
    - Return faked sensitive information

- **Namespace/Filesystem Isolation**

- **Context-Aware Device Virtualization**
Enabling Techniques

- Decoupled App Isolation Runtime (AIR)
- Namespace/Filesystem Isolation
- Context-Aware Device Virtualization
Enabling Techniques

- **Decoupled App Isolation Runtime (AIR)**
- **Namespace/Filesysterm Isolation**
  - Prevent communication between runtimes
  - Accomplished using a cgroup
- **Context-Aware Device Virtualization**
Enabling Techniques

- Decoupled App Isolation Runtime (AIR)
- Namespace/Filesysteem Isolation
- Context-Aware Device Virtualization
Enabling Techniques

- Decoupled App Isolation Runtime (AIR)
- Namespace/Filesystem Isolation
- Context-Aware Device Virtualization
  - Contention for hardware resources
    - Ex: SurfaceFlinger
  - Allow access only to the active runtime
Enabling Techniques

- Decoupled App Isolation Runtime (AIR)
- Namespace/Filesyststem Isolation
- Context-Aware Device Virtualization
Outline

• Problem Definition
• System Design
  – Threat Model
  – Enabling Techniques
• Implementation (AirBag)
• Evaluation
• Limitations
• Conclusion
Implementation

Fig. 4. Seamless Integration of AirBag
Implementation

- To launch the app stub:
  - Prepare a separate filesystem root
  - Run `airbag_init`
  - Create network device
    - Forward network and phone requests
  - Determine namespace via:
    ```
task_struct->nsproxy->current
```
Implementation

(a) A screenshot of HippoSMS-infected video browser

(b) A pop-up alert on background SMS behavior
## Implementation

<table>
<thead>
<tr>
<th>Hardware Device</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>Audio Playback and Capture</td>
</tr>
<tr>
<td>Framebuffer</td>
<td>Display Output</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics Processor</td>
</tr>
<tr>
<td>Input</td>
<td>Touchscreen and Buttons</td>
</tr>
<tr>
<td>IPC</td>
<td>Binder IPC Framework</td>
</tr>
<tr>
<td>Networking</td>
<td>WiFi Network Interface</td>
</tr>
<tr>
<td>pmem</td>
<td>Physical Memory Allocator</td>
</tr>
<tr>
<td>Power</td>
<td>Power Management (Suspend/Resume)</td>
</tr>
<tr>
<td>RTC</td>
<td>Real Time Clock</td>
</tr>
<tr>
<td>Sensors</td>
<td>Temperature, Accelerometer, GPS</td>
</tr>
<tr>
<td>Telephony</td>
<td>Cellular Radio (GSM, CDMA)</td>
</tr>
</tbody>
</table>
## Implementation

<table>
<thead>
<tr>
<th>Hardware Device</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>Audio Playback and Capture</td>
</tr>
<tr>
<td>Framebuffer</td>
<td>Display Output</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics Processor</td>
</tr>
<tr>
<td>Input</td>
<td>Touchscreen and Buttons</td>
</tr>
<tr>
<td>IPC</td>
<td>Binder IPC Framework</td>
</tr>
<tr>
<td>Networking</td>
<td>WiFi Network Interface</td>
</tr>
<tr>
<td>pmem</td>
<td>Physical Memory Allocator</td>
</tr>
<tr>
<td>Power</td>
<td>Power Management (Suspend/Resume)</td>
</tr>
<tr>
<td>RTC</td>
<td>Real Time Clock</td>
</tr>
<tr>
<td>Sensors</td>
<td>Temperature, Accelerometer, GPS</td>
</tr>
<tr>
<td>Telephony</td>
<td>Cellular Radio (GSM, CDMA)</td>
</tr>
</tbody>
</table>
Implementation

- Telephony support is partly dependent on vendor
  - Service daemon: rild
  - Vendor library: libhtc_ril.so
  - Java class: com.android.internal.telephony.RIL
Implementation

Fig. 3. Telephony Virtualization in AirBag
Implementation

• To update the screen, allocate a separate framebuffer

• Driver reads framebuffer matching current namespace
Implementation
Outline

● Problem Definition
● System Design
  – Threat Model
  – Enabling Techniques
● Implementation (AirBag)
● Evaluation
● Limitations
● Conclusion
Evaluation: Effectiveness

GoldDream

- Reads received SMS messages
- Uploads them to a remote server

(a) Faked phone number is being accessed

(b) Faked phone number is being uploaded
Evaluation: Effectiveness

HippoSMS

- Sends messages to premium-rate numbers
Evaluation: Performance

<table>
<thead>
<tr>
<th>Benchmark Name</th>
<th>Version</th>
<th>Workload Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AnTuTu Benchmark [5]</td>
<td>2.8.3</td>
<td>Combination</td>
</tr>
<tr>
<td>BrowserMark [7]</td>
<td>2.0</td>
<td>CPU/IO</td>
</tr>
<tr>
<td>NenaMark2 [11]</td>
<td>2.3</td>
<td>GPU</td>
</tr>
<tr>
<td>Neocore [12]</td>
<td>1.9.35</td>
<td>GPU</td>
</tr>
<tr>
<td>SunSpider [15]</td>
<td>0.9.1</td>
<td>CPU/IO</td>
</tr>
</tbody>
</table>
Evaluation: Performance

Fig. 8. Performance Measurement of AirBag on Google Nexus One, Nexus 7, and Samsung Galaxy S III
Evaluation: Performance

Fig. 9. AnTuTu Measurement Results
Evaluation: Power/Memory

- Fully-charged device (Nexus 7)
  - 24 hours, no workload
    - **Stock**: 91%
    - **AirBag**: 89%
Evaluation: Power/Memory

• Memory footprint
  - 4 hours, no workload
    • Stock: 59.31%
    • AirBag: 60.87%
  - 4 hours, repeated audio
    • Stock: 60.25%
    • AirBag: 63.70%
Outline

- Problem Definition
- System Design
  - Threat Model
  - Enabling Techniques
- Implementation (AirBag)
- Evaluation
- Limitations
- Conclusion
Limitations

- Apps cannot migrate between each runtime
Limitations

• Apps cannot migrate between each runtime
• No incoming calls/messages in AIR
Limitations

- Apps cannot migrate between each runtime
- No incoming calls/messages in AIR

http://www.imdb.com/title/tt0479968/
Limitations

• Apps cannot migrate between each runtime
• No incoming calls/messages in AIR
• One runtime for all untrusted apps
• Malicious app may detect sand-boxing
Outline

- Problem Definition
- System Design
  - Threat Model
  - Enabling Techniques
- Implementation (AirBag)
- Evaluation
- Limitations
- Conclusion
Conclusion

- AirBag
  - Lightweight OS-level virtualization for Android
  - Provides a separate application runtime
  - Prevent leakage of sensitive information