Dr. Android and Mr. Hide: 
Fine-grained Permissions in Android Applications

Jinseong Jeon*, Kristopher K. Micinski*, Jeffrey A. Vaughan#, 
Ari Fogel+, Nikhilesh Reddy+, Jeffrey S. Foster*, and Todd Millstein+.

* University of Maryland, College Park

# LogicBlox (work performed while the author was at UCLA)

+ University of California, Los Angeles
Permissions on Android

- Permissions associated with resources and OS features
  - Internet, GPS, telephony, ...

- Permissions granted at install time
  - once granted, apps can use such permissions any way they want

- Example: GasBuddy app
  - Internet
    - to access to gasbuddy.com
  - Location
    - to find cheapest gas near users
  - Phone state
    - to access phone’s unique ID
Proposal: Finer-grained Permissions

- Hypothesis: permissions are too broad
  - GasBuddy has access to entire Internet
  - But, only needs access to gasbuddy.com

- Our goal: add finer-grained permissions
  - Help *app developers*
    - improve apps’ robustness against security exploits
    - attest to apps’ proper usage of permissions
  - Help *users*
    - understand *how* apps use permissions
  - Add permissions without changing platform
Example Finer-grained Permissions

- Internet ➔ InternetURL\(d\)
  - InternetURL(gasbuddy_com)

- GPS ➔ LocationBlock
  - 150m resolution (city block)

- Phone state ➔ PhoneState\(p\)
  - PhoneState(UniqueID)
Our Tool Chain

- **RefineDroid** infers how permissions are used in the app
- **Dr. Android** retrofits the app with fine-grained permissions
- **Mr. Hide** enforces fine-grained permissions at runtime
# Fine-grained Permissions in Mr. Hide

<table>
<thead>
<tr>
<th>Android Permissions</th>
<th>Full Capabilities</th>
<th>Fine-grained...</th>
<th>...in Mr. Hide</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERNET</td>
<td>any sites</td>
<td>particular domains</td>
<td>InternetURL(d)</td>
</tr>
<tr>
<td>LOCATION</td>
<td>exact location</td>
<td>block resolution</td>
<td>LocationBlock</td>
</tr>
<tr>
<td>READ_PHONE_STATE</td>
<td>all kinds</td>
<td>only (e.g.) IMEI</td>
<td>ReadPhoneState(p)</td>
</tr>
<tr>
<td>READ_CONTACT</td>
<td>all records</td>
<td>specific fields</td>
<td>ContactCol(c)</td>
</tr>
<tr>
<td>WRITE_SETTINGS</td>
<td>all kinds</td>
<td>only (e.g.) ringtone</td>
<td>WriteSettings(s)</td>
</tr>
</tbody>
</table>
RefineDroid

- infers how permissions are used
- string analysis to search URL-like strings
- constant propagation analysis to determine *key parameters* to privileged APIs
- e.g. for system settings:

```java
Uri uri = Uri.parse("my_ringtone.mp3");
RingtoneManager.setActualDefaultRingtoneUri(
    this, RingtoneManager.TYPE_ALARM, uri);
String path = uri.toString();
Settings.System.putString(
    getContentResolver(),
    Settings.System.RINGTONE, path);
```
Permission Usage in the Wild

- 750 apps: 24 free apps in each Google Play category
- some (e.g. 44 for write settings) apps are over-privileged
- use far narrower capabilities than available
- a few popular items are used frequently
Mr. Hide

the Hide interface to the droid environment
- services
  - interact with a client app and resources
- client-side library
  - a drop-in replacement for sensitive APIs
**Dalvik Rewriter for Android**
- injects hidelib.dex
- modifies the app’s bytecode to use Mr. Hide
- removes Android perms. and adds Mr. Hide perms.
We also looked at whether we could use better constant propagation to eliminate false positives, rather than the simplistic approach used currently. We found that it would require developing a much more sophisticated constant propagation system to achieve this goal. However, we believe it may not be worth the effort.

While we may investigate this in future work, we have found that the majority of false positives can be prevented by rewriting apps to use different permissions, particularly in the context of access from different countries based on the IP address the device is using. We found that false negatives are more problematic, as they can prevent a rewritten app from executing as expected. For example, Amazon uses a large set of domains which deal with access from different countries, and we found that the majority of these domains are for third party ads or similar purposes.

This is particularly useful in the context of rewriting, as RefineDroid has a relatively low rate of false negatives and a higher rate of false positives. The low rate of false negatives serves as evidence that policies produced by RefineDroid may not be as refined as we would like.

Contacts report this information for when RefineDroid fails to discover a fine-grained permission by one or more fine-grained permissions, indicating policies that cannot obviously be removed, and which exercise our fine-grained permissions, as shown in Table 1. Figure 10 shows the fine-grained permissions for 14 apps from Google Play.

### 14 case studies
- which exercise our fine-grained permissions
- 31 out of 35 usages of perms. are replaced
- not always appropriate: e.g. *sftp* feature of ASTRO

Table 1: Fine-grained permissions for 14 apps from Google Play

<table>
<thead>
<tr>
<th>Permission</th>
<th>Amazon</th>
<th>Angry Birds</th>
<th>Angry Birds Rio</th>
<th>ASTRO</th>
<th>Baby Monitor</th>
<th>GasBuddy</th>
<th>Horoscope</th>
<th>Shazam</th>
<th>Google Sky Map</th>
<th>Task Killer</th>
<th>Brightest Flashlight</th>
<th>Ultimate Flashlight</th>
<th>Qrdroid</th>
<th>Radar Now!</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERNET</td>
<td>● ● ● ●</td>
<td>○ ● ● ● ● ●</td>
<td>○ ● ● ● ● ●</td>
<td>● ● ●</td>
<td>● ● ● ● ● ●</td>
<td>● ● ●</td>
<td></td>
<td></td>
<td>● ● ● ● ● ●</td>
<td>● ● ● ● ●</td>
<td></td>
<td>● ● ● ● ● ●</td>
<td>● ● ● ● ●</td>
<td></td>
</tr>
<tr>
<td>READCONTACTS</td>
<td>● ● ● ●</td>
<td>○ ● ● ● ● ●</td>
<td>○ ● ● ● ● ●</td>
<td>● ● ●</td>
<td>● ● ●</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACCESS_ LOCATION</td>
<td>● ● ● ●</td>
<td>○ ● ● ● ● ●</td>
<td>○ ● ● ● ● ●</td>
<td>● ● ●</td>
<td>● ● ●</td>
<td></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>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE_SETTINGS</td>
<td>●</td>
<td>●</td>
<td>○</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>○</td>
<td>●</td>
<td>●</td>
<td>○</td>
<td>●</td>
</tr>
<tr>
<td>InternetURL(·)</td>
<td>✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>ContactCol(·)</td>
<td>✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>LocationBlock</td>
<td>✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>ReadPhoneState(UniqueId)</td>
<td>✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>WriteSettings(Ringtone)</td>
<td>✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
</tbody>
</table>

built-in Android perm. ●: can be replaced ○: cannot be replaced ✗: over-privileged

fine-grained perm. ✓: needed, and being used
Transformation Time

<table>
<thead>
<tr>
<th>Name</th>
<th># Ins</th>
<th># Chg</th>
<th>Tm (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon</td>
<td>114,691</td>
<td>174</td>
<td>17.86</td>
</tr>
<tr>
<td>Angry Birds</td>
<td>79,311</td>
<td>760</td>
<td>11.44</td>
</tr>
<tr>
<td>Angry Birds Rio</td>
<td>173,441</td>
<td>968</td>
<td>21.92</td>
</tr>
<tr>
<td>ASTRO</td>
<td>149,911</td>
<td>695</td>
<td>18.30</td>
</tr>
<tr>
<td>Baby Monitor</td>
<td>12,378</td>
<td>1</td>
<td>3.81</td>
</tr>
<tr>
<td>Gas Buddy</td>
<td>67,514</td>
<td>222</td>
<td>11.81</td>
</tr>
<tr>
<td>Horoscope</td>
<td>92,441</td>
<td>829</td>
<td>12.73</td>
</tr>
<tr>
<td>Shazam</td>
<td>259,643</td>
<td>778</td>
<td>30.67</td>
</tr>
<tr>
<td>Google Sky Map</td>
<td>33,355</td>
<td>193</td>
<td>8.38</td>
</tr>
<tr>
<td>Task Killer</td>
<td>9,696</td>
<td>76</td>
<td>6.10</td>
</tr>
<tr>
<td>Brightest Flashlight</td>
<td>174,159</td>
<td>1,265</td>
<td>18.94</td>
</tr>
<tr>
<td>Ultimate Flashlight</td>
<td>46,878</td>
<td>464</td>
<td>8.26</td>
</tr>
<tr>
<td>Qrdroid</td>
<td>105,400</td>
<td>11</td>
<td>9.05</td>
</tr>
<tr>
<td>Radar Now!</td>
<td>26,706</td>
<td>121</td>
<td>7.66</td>
</tr>
</tbody>
</table>

- small changes are enough
  - thanks to Mr. Hide’s clean drop-in replacements
- fast enough for offline use
Correctness of Transformation

- Applied RefineDroid-Dr.Android-Mr.Hide and then run the apps manually
- Rewritten apps pass the Dalvik verifier
- Almost all activities of apps function normally, with no observable changes

Limitations
- small behavioral differences due to WebView Mr. Hide didn’t support
- performance overhead due to IPC
  - but not that significant in practice
Conclusion

- Fine-grained Permissions
  - RefineDroid, a static analyzer to infer...
  - Mr. Hide, clean interfaces to enforce...
  - Dr. Android, a bytecode rewriter to retrofit...

- Broader uses
  - can be enhanced to check suspicious perm. usage
  - other security policies
    - inserting mock data, changing policies at run-time, etc.
  - general support for modifying Dalvik bytecode

- Hiring postdoc
Android Popularity

- **Android**, the most popular smartphone platform
- 64.1% market share (2Q12)\(^1\), 675K apps, 25B installs\(^2\)

\(^1\)http://www.gartner.com/it/page.jsp?id=2120015
\(^2\)http://officialandroid.blogspot.com/2012/09/google-play-hits-25-billion-downloads.html
(1) **RefineDroid**: light-weight analysis to infer the app’s perm. usages

(2) **Dr. Android**: bytecode transformer to retrofit the app w/ fine-grained perm.

(3) **Mr. Hide**: easy-to-replace interfaces to enforce fine-grained perm.
<table>
<thead>
<tr>
<th><strong>1 Outside Resources</strong></th>
<th><strong>InternetURL(d)</strong></th>
<th><strong>2 Structured User Info.</strong></th>
<th><strong>3 Sensors</strong></th>
<th><strong>LocationBlock</strong></th>
<th><strong>4 System State/Settings</strong></th>
<th><strong>ReadPhoneState(p)</strong></th>
<th><strong>WriteSettings(s)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INTERNET, SMS, BLUETOOTH, NFC, EXTERNAL_STORAGE,</strong> ...</td>
<td><strong>whitelist or blacklist</strong></td>
<td><strong>CALANDER, CONTACT, ACCOUNTS, LOGS,</strong> ...</td>
<td><strong>LOCATION, CAMEAR, AUDIO,</strong> ...</td>
<td><strong>reducing the fidelity (e.g., blur)</strong></td>
<td>*<em>BOOKMARKS, SETTINGS, <em>_STATES,</em></em> ...</td>
<td><strong>subset</strong></td>
<td></td>
</tr>
</tbody>
</table>
RefineDroid

A static analysis tool
- infers which fine-grained permissions are used
  - InternetURL(d), ContactCol(c), ReadPhoneState(p), WriteSettings(s)
- via simple analyses
  - string analysis to search URL-like strings
  - constant propagation analysis to determine key parameters to privileged APIs

<table>
<thead>
<tr>
<th>(a) Contacts</th>
<th>(b) Phone state</th>
<th>(c) System settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cursor c = getContentResolver() .query(uri, projections, selection, ...); int index = c.getColumnIndex(ContactsContract.PhoneLookup.NUMBER); String id = c.getString(index);</td>
<td>TelephonyManager tm = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE); String id = tm.getDeviceId(); tm.listen(new PhoneStateListener() { ... }, PhoneStateListener.LISTEN_CALL_STATE);</td>
<td>Uri uri = Uri.parse(&quot;my_ringtone.mp3&quot;); RingtoneManager.setActualDefaultRingtoneUri(this, RingtoneManager.TYPE_ALARM, uri); String path = uri.toString(); Settings.System.putString(getContentResolver(), Settings.System.RINGTONE, path);</td>
</tr>
</tbody>
</table>
Over-privileged

- Apps use far fewer capabilities than available
few popular items are used

<table>
<thead>
<tr>
<th>Domain</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>google.com</td>
<td>457</td>
</tr>
<tr>
<td>admob.com</td>
<td>324</td>
</tr>
<tr>
<td>gstatic.com</td>
<td>311</td>
</tr>
<tr>
<td>facebook.com</td>
<td>256</td>
</tr>
<tr>
<td>android.com</td>
<td>207</td>
</tr>
</tbody>
</table>

- **Apps using domain**

- **Column popularity**

- **Settings popularity**

- **Phone state popularity**
Mr. Hide

- the _Hide_ interface to the _droid_ environment
  - services
    - interact with a client app and resources
    - strong guarantee of permission enforcement
  - client-side library
    - a drop-in replacement for sensitive APIs
binding to Mr. Hide services

- most hidelib methods are synchronous
  - to be compatible to built-in Android APIs
  - using RPCs via binders
- but, service binding is asynchronous!
- inserted a splash activity, while waiting for connections
Mr. Hide (Implementation)

- uses Android’s permission framework
  - able to define our own set of permissions
  - parameterized permissions via a permission tree
    - hidelib.permission.net
    - hidelib.permission.net.google_com

- binding to Mr. Hide services
  - once bound, synchronous RPCs are enabled
  - inserted a splash screen while waiting for connections
  - currently, only launcher activity is supported

- permission-specific
  - proxies for system resource managers, etc.
**Dalvik Rewriter for Android**
- modifies the app’s bytecode to use Mr. Hide
- injects hidelib.dex
- removes Android perms. and adds Mr. Hide perms.
Rewriting Manifest
- trivial to manipulate XML-formatted document

Rewriting bytecode
- merging two DEX files
  - all pools should be duplicate-free and sorted
- replacing references to Android classes with corresponding hidelib proxies
- inserting service binding steps

permission-specific
- finding code patterns to retrieve instances of system resource managers and then returning Mr. Hide’s managers in lieu of the platform’s, etc.
Based on our study we believe it may not be worth the effort of developing a much more sophisticated constant propagation to eliminate false positives, rather than the simpler and faster approach we use currently. We found that it would result in the same set of reachable domains, but only one time as much analysis needed. For example, Amazon uses a large set of domains which deal with access from different parts of the application. We found that these false positives can prevent a rewritten app from executing successfully. We examined the false positives in detail, and found that the majority are domains for third party ads or similar. This is particularly useful in the context of rewriting, as Dr. Android and Mr. Hide prevent this by removing the original permissions. Contacts report this information for accounts, which is not always the case.

We also looked at whether we could use better constant propagation and observed that the majority of the apps we examined had a high rate of false positives. The low rate of false negative was generally reachable, but not always within the test settings. For example, Amazon uses a higher rate of false positives. The low rate of false negatives can prevent a rewritten app from executing successfully. When RefineDroid fails to discover a fine-grained permission that an app needs to run. Columns grouped under heading #Chg Tm (s) indicate policies that cannot obviously be removed, and the user is asked to review the decision.

<table>
<thead>
<tr>
<th>Name</th>
<th>Apk (KB)</th>
<th>Dex (KB)</th>
<th># Ins</th>
<th># Chg</th>
<th>Tm (s)</th>
<th>Domains</th>
<th>Contacts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon</td>
<td>1,607</td>
<td>2,288</td>
<td>114,691</td>
<td>174</td>
<td>17.86</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>Angry Birds</td>
<td>993</td>
<td>15,018</td>
<td>79,311</td>
<td>760</td>
<td>11.44</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td>Angry Birds Rio</td>
<td>2,081</td>
<td>22,716</td>
<td>173,441</td>
<td>968</td>
<td>21.92</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>ASTRO</td>
<td>1,428</td>
<td>2,348</td>
<td>149,911</td>
<td>695</td>
<td>18.30</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Baby Monitor</td>
<td>163</td>
<td>781</td>
<td>12,378</td>
<td>1</td>
<td>3.81</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Gas Buddy</td>
<td>781</td>
<td>1,269</td>
<td>67,514</td>
<td>222</td>
<td>11.81</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Horoscope</td>
<td>844</td>
<td>3,731</td>
<td>92,441</td>
<td>829</td>
<td>12.73</td>
<td>17</td>
<td>16</td>
</tr>
<tr>
<td>Shazam</td>
<td>2,641</td>
<td>3,904</td>
<td>259,643</td>
<td>778</td>
<td>30.67</td>
<td>20</td>
<td>17</td>
</tr>
<tr>
<td>Google Sky Map</td>
<td>459</td>
<td>2,212</td>
<td>33,355</td>
<td>193</td>
<td>8.38</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Task Killer</td>
<td>129</td>
<td>99</td>
<td>9,696</td>
<td>76</td>
<td>6.10</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Brightest Flashlight</td>
<td>1,870</td>
<td>1,756</td>
<td>174,159</td>
<td>1,265</td>
<td>18.94</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>Ultimate Flashlight</td>
<td>485</td>
<td>1,287</td>
<td>46,878</td>
<td>464</td>
<td>8.26</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>Qrdroid</td>
<td>922</td>
<td>3,802</td>
<td>105,400</td>
<td>11</td>
<td>9.05</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>Radar Now!</td>
<td>379</td>
<td>569</td>
<td>26,706</td>
<td>121</td>
<td>7.66</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

- row rate of false negative
- near-upper bound on app’s behavior
- high rate of false positive
- generally reachable, but not within the test settings