Security in Android Applications

Master Thesis

Pascal Gadient

Software Composition Group
University of Bern
Switzerland
Agenda

- Introduction
- What are security code smells?
- How prevalent are they?
- Why identifying security smells is helpful?
- Conclusion
Software is Everywhere
We Love Apps

MAYBE IF WE TELL PEOPLE
THE BRAIN IS AN APP

THEY WILL START USING IT
Mobile Device Addiction
Mobile Security is Vital
Software Insecurity Thrives

Security in Android Applications
Security Code Smell

Definition:
Symptoms in the code that indicate the prospect of security and privacy vulnerabilities.
Research Goals

- **RQ$_1$**: What are the security code smells in Android apps?
- **RQ$_2$**: How prevalent are security smells in benign apps?
- **RQ$_3$**: To which extent identifying such smells facilitates detecting security vulnerabilities?
Literature Review

Keyword Search

Inspection of citations and cited papers

Reading of abstracts and introductions

Agreement by discussion

Collection of worklisted papers

Security in Android Applications
What are the security code smells in Android apps?
Insufficient Attack Protection

➤ Unreliable Information Sources
➤ Untrustworthy / Outdated Libraries
➤ Native Code
➤ Open to Piggybacking
➤ Unnecessary Permissions
Security Invalidation

- Weak Crypto Algorithm or Configuration
- Improper Certificate Use
- Unacknowledged Distribution
Broken Access Control

- Insecure Inter-Component Communication
- Unprotected System Sockets
- Custom Scheme Channel
Sensitive Data Exposure

- Insecure Storage
- Exposed Identifiers
Lax Input Validation

- Unverified JavaScript Code
- Dynamic Code Loading
- SQL Injection
How prevalent are security smells in apps?
Scope of the Study

» Random apps from AndroZoo

» Corpora size:

  46,000 apps
  440 GB

» Lightweight analysis

» 10 out of 28 smells analysed
Subjects of the Study

Apktool

Small

XML

Web parser

-smali

-iget-object

-v7, p0, Linfo/sm...
Prevalence of Smells

# of different smells apps suffer

- 0, 9%
- 1, 10%
- 2, 22%
- 3, 16%
- 4, 13%
- 5, 11%
- 6, 11%
- 7, 6%
- 8, 2%
Prevalence of Smells

# of different smells apps suffer

- 2, 22%
- 3, 16%
- 4, 13%
- 5, 11%
- 6, 11%
- 7, 6%
- 8, 2%
- 0, 9%
- 1, 10%
Prevalence of Smells

# of different smells apps suffer

3, 16%
4, 13%
5, 11%
6, 11%
7, 6%
8, 2%
0, 9%
1, 10%
2, 22%
Distribution of Each Smell

- Debuggable Release: 1%
- Exposed Clipboard: 10%
- Header Attachment: 11%
- Insecure Network Protocol: 12%
- Broken WebView's Sandbox: 33%
- Improper Certificate Validation: 36%
- Unique Hardware Identifier: 41%
- Custom Scheme Channel: 44%
- Dynamic Code Loading: 61%
- XSS-like Code Injection: 85%
Distribution of Each Smell

<table>
<thead>
<tr>
<th>Smell</th>
<th># of apps affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debuggable Release</td>
<td>1%</td>
</tr>
<tr>
<td>Exposed Clipboard</td>
<td>10%</td>
</tr>
<tr>
<td>Header Attachment</td>
<td>11%</td>
</tr>
<tr>
<td>Insecure Network Protocol</td>
<td>12%</td>
</tr>
<tr>
<td>Broken WebView's Sandbox</td>
<td>33%</td>
</tr>
<tr>
<td>Improper Certificate Validation</td>
<td>36%</td>
</tr>
<tr>
<td>Unique Hardware Identifier</td>
<td>41%</td>
</tr>
<tr>
<td>Custom Scheme Channel</td>
<td>44%</td>
</tr>
<tr>
<td>Dynamic Code Loading</td>
<td>61%</td>
</tr>
<tr>
<td>XSS-like Code Injection</td>
<td>85%</td>
</tr>
</tbody>
</table>
Distribution of Each Smell

<table>
<thead>
<tr>
<th>Smell</th>
<th># of apps affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debuggable Release</td>
<td>1%</td>
</tr>
<tr>
<td>Exposed Clipboard</td>
<td>10%</td>
</tr>
<tr>
<td>Header Attachment</td>
<td>11%</td>
</tr>
<tr>
<td>Insecure Network Protocol</td>
<td>12%</td>
</tr>
<tr>
<td>Broken WebView's Sandbox</td>
<td>33%</td>
</tr>
<tr>
<td>Improper Certificate Validation</td>
<td>36%</td>
</tr>
<tr>
<td>Unique Hardware Identifier</td>
<td>41%</td>
</tr>
<tr>
<td>Custom Scheme Channel</td>
<td>44%</td>
</tr>
<tr>
<td>Dynamic Code Loading</td>
<td>61%</td>
</tr>
<tr>
<td>XSS-like Code Injection</td>
<td>85%</td>
</tr>
</tbody>
</table>
Distribution of Each Smell

Security in Android Applications
Distribution of Each Smell in API Levels

API level

% of security smells

0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 21

- Broken WebView's Sandbox
- XSS-like Code Injection
- Dynamic Code Loading
- Improper Certificate Validation
- Insecure Network Protocol
- Exposed Clipboard
- Unique Hardware Identifier
- Header Attachment
- Custom Scheme Channel
- Debuggable Release

Security in Android Applications
Distribution of Each Smell in API Levels

Security in Android Applications
Distribution of Each Smell in API Levels

Percentage of security smells across different API levels.
Distribution of Each Smell in API Levels

- Broken WebView's Sandbox
- XSS-like Code Injection
- Dynamic Code Loading
- Improper Certificate Validation
- Insecure Network Protocol
- Exposed Clipboard
- Unique Hardware Identifier
- Header Attachment
- Custom Scheme Channel
- Debuggable Release

% of security smells vs. API level
The Impact of Number of Downloads

<table>
<thead>
<tr>
<th>Popularity</th>
<th># of security smells</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 50K</td>
<td><img src="image1" alt="Boxplot" /></td>
</tr>
<tr>
<td>50K - 500K</td>
<td><img src="image2" alt="Boxplot" /></td>
</tr>
<tr>
<td>500K - 5M</td>
<td><img src="image3" alt="Boxplot" /></td>
</tr>
<tr>
<td>5M - 50M</td>
<td><img src="image4" alt="Boxplot" /></td>
</tr>
<tr>
<td>50M - 500M</td>
<td><img src="image5" alt="Boxplot" /></td>
</tr>
<tr>
<td>&gt; 500M</td>
<td><img src="image6" alt="Boxplot" /></td>
</tr>
</tbody>
</table>
The Impact of User Ratings

The diagram shows the distribution of security smells across different user rating ranges.

- **1.0 - 1.9**: The boxplot indicates a lower number of security smells, with a median of 2.
- **2.0 - 2.9**: A slight increase in security smells, median is around 3.
- **3.0 - 3.9**: Further increase, median is close to 4.
- **4.0 - 4.9**: A steady rise, median is approximately 5.
- **5**: The highest rating category shows the maximum number of security smells, with a median close to 6.

Each boxplot represents a range of user ratings, with the box showing the interquartile range and the whiskers indicating the spread of data points.
To which extent identifying security smells facilitates detecting vulnerabilities?
Study Design
Result

- Header Attachment
- Improper Certificate Validation
- Insecure Network Protocol
- Exposed Clipboard
- Others
Where Vulnerability Reports Failed?

- **Header Attachment**
  
  *Data sensitivity of headers*

- **Improper Certificate Validation**
  
  *Customised TrustManagers with pinning support*

- **Insecure Network Protocol**
  
  *Local web resources in middleware*

- **Exposed Clipboard**
  
  *Data sensitivity of content*
Summary
Our Contribution

- Increase in security awareness
- Evaluation of security smell distribution
- Lightweight analysis assessment
- In future: In-depth exploration
Thank you for your attention!