Benchmarking Android Security Analysis

A Bachelors Project,
Final Presentation

by Timo Spring
Supervised by Claudio Corrodi
1. Project Overview

What is it about?

Problem
• Millions of android apps
• Hundreds of analysis tools
• Large scale taxonomies classifying them
• Lack of comparison in practice

Project Idea
• Run selected tools on common dataset
• Compare the results from the different tools
1. Project Overview

**Benchmarking concept**

**Small scale qualitative**
- DroidBench dataset (119 apps)
- Common Configuration
- Manually check the validity of the reported leaks

**Large scale quantitative**
- F-Droid dataset (~1.5k apps)
- Automatically analyse number of detections and matchings
2. Tool Selection Process

Focus on vulnerability detection

2. Tool Selection Process

... only few tools obtainable and runnable
### 3. Selected Tools In A Nutshell

*Tools in a nutshell – pretty much the same*

<table>
<thead>
<tr>
<th></th>
<th>COVERT</th>
<th>Flowdroid</th>
<th>IccTA</th>
<th>IC3 (Epicc)</th>
<th>Horndroid</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type:</strong></td>
<td>Static &amp; Formal</td>
<td>Static</td>
<td>Static</td>
<td>Static</td>
<td>Static &amp; Formal</td>
</tr>
<tr>
<td><strong>Artefact:</strong></td>
<td>Manifest</td>
<td>Manifest</td>
<td>Manifest</td>
<td>Manifest</td>
<td>Code</td>
</tr>
<tr>
<td></td>
<td>Code</td>
<td>Layout</td>
<td>Layout</td>
<td></td>
<td>Code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Code</td>
<td>Code</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Sensitivity:</strong></td>
<td>Flow</td>
<td>Flow</td>
<td>Flow</td>
<td>Flow*</td>
<td>Flow*</td>
</tr>
<tr>
<td></td>
<td>Field</td>
<td>Field</td>
<td>Field</td>
<td>Field*</td>
<td>Field*</td>
</tr>
<tr>
<td></td>
<td>Context</td>
<td>Context</td>
<td>Context</td>
<td>Context</td>
<td>Context</td>
</tr>
<tr>
<td><strong>Sources and Sinks</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Uses</strong></td>
<td>Flowdroid</td>
<td>Flowdroid</td>
<td>Flowdroid IC3</td>
<td>Flowdroid</td>
<td></td>
</tr>
</tbody>
</table>

* partially
3. Benchmarking Implementation

*Runs tools and parses output*

Class: `org.cert.sendsms.ButtonListener`
Method: `onClick(android.view.View) void`
Sink Method: `sendMessage(String uid) void`
Detected by: `flowdroid, iccta`

- Easy to extend with new tools (artefact, parser, results)
- Usability
4. Small Scale Analysis

_DroidBench facilitates analysis for true/false positives_

- 119 apps with known data leak vulnerabilities
- 125 leaks (sinks) – indicated in source code
- Enables analysis for true/false positives
4. Evaluation – Small Scale Analysis

**Metrics for comparison**

- **Number of reported vulnerabilities**
  - True / false positives

- **Precision & recall**
  - Compare performance

- **McNamar’s Test**
  - Pairwise comparison (similarity)
4. Evaluation – Small Scale Analysis

Overview of true and false positives

<table>
<thead>
<tr>
<th>Tool</th>
<th>True Positives</th>
<th>False Positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flowdroid</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>Horndroid</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>COVERT</td>
<td>100</td>
<td>25</td>
</tr>
<tr>
<td>IC3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IccTA</td>
<td>100</td>
<td>50</td>
</tr>
</tbody>
</table>
4. Evaluation – Small Scale Analysis

*Flowdroid with highest accuracy*

---

![Bar chart showing True Positives and False Positives for Flowdroid, Horndroid, COVERT, IC3, and IccTA.](chart.png)

- Flowdroid: ~70% True Positives, ~55% False Positives
- Horndroid: ~58% True Positives
- COVERT: ~55% True Positives
- IC3: ~41% True Positives
- IccTA: ~68% True Positives

# DROIDBENCH LEAKS

- Flowdroid: ~70%
- Horndroid: ~58%
- COVERT: ~68%
- IC3: ~55%
- IccTA: ~41%
4. Evaluation – Small Scale Analysis

**COVERT and IC3 under-perform**

<table>
<thead>
<tr>
<th>Tool</th>
<th>True Positives</th>
<th>False Positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flowdroid</td>
<td>~70%</td>
<td>~55%</td>
</tr>
<tr>
<td>Horndroid</td>
<td>~58%</td>
<td>~41%</td>
</tr>
<tr>
<td>COVERT</td>
<td>~55%</td>
<td>~41%</td>
</tr>
<tr>
<td>IC3</td>
<td>~41%</td>
<td>~41%</td>
</tr>
<tr>
<td>IccTA</td>
<td>~68%</td>
<td>~41%</td>
</tr>
</tbody>
</table>

# DROIDBENCH LEAKS
4. Evaluation – Small Scale Analysis

COVERT with highest precision

Precision
~70%
4. Evaluation – Small Scale Analysis

IC3 very unprecise and inaccurate

Precision
~10%
4. Evaluation – Small Scale Analysis

*Flowdroid and Horndroid recall most true positives*

![Graph showing recall and true positives for various tools. Flowdroid and Horndroid have approximately 79% recall.](image)

- **Flowdroid** and **Horndroid** recall most true positives with an approximate 79% recall.
4. Evaluation – Small Scale Analysis

How about our implementation?

- Flowdroid: ~79%
- Benchmarking: ~90%
4. Evaluation – Small Scale Analysis

Agreement effect on probability of correct classification

<table>
<thead>
<tr>
<th></th>
<th>HornDroid</th>
<th>COVERT</th>
<th>IC3</th>
<th>IccTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlowDroid</td>
<td>0.729</td>
<td>0.776</td>
<td>0.653</td>
<td>0.699</td>
</tr>
<tr>
<td>HornDroid</td>
<td>0.696</td>
<td>0.476</td>
<td>0.709</td>
<td></td>
</tr>
<tr>
<td>COVERT</td>
<td></td>
<td>0.479</td>
<td>0.754</td>
<td></td>
</tr>
<tr>
<td>IC3</td>
<td></td>
<td></td>
<td>0.62</td>
<td></td>
</tr>
</tbody>
</table>

Agreement of tools fairly impacts the probability of true classification
4. Evaluation – Small Scale Analysis

Best performing tools are significantly similar

<table>
<thead>
<tr>
<th></th>
<th>HornDroid</th>
<th>COVERT</th>
<th>IC3</th>
<th>IccTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlowDroid</td>
<td>9.94</td>
<td>10.56</td>
<td>35.29</td>
<td>2.77</td>
</tr>
<tr>
<td>HornDroid</td>
<td></td>
<td>0.2</td>
<td>8.53</td>
<td>6.01</td>
</tr>
<tr>
<td>COVERT</td>
<td></td>
<td></td>
<td>26.33</td>
<td>6.97</td>
</tr>
<tr>
<td>IC3</td>
<td></td>
<td></td>
<td></td>
<td>28.99</td>
</tr>
</tbody>
</table>

Statistical significant similarities among tools are observable
4. Evaluation – Small Scale Analysis

*Summarized*

- Flowdroid has best performance among tools
- Benchmarking can leverage base approaches
- Tools with better performance tend to be significantly similar
5. Next steps

*Large Scale Analysis*

- Run analysis on F-Droid data set (~1.5k real world apps)
- Verify number of matchings among tools
- Already detected 669 vulnerabilities for 108 real world apps
  - 10 vulnerabilities are reported by at least two tools

⚠️ Time consuming: A lot of time outs, especially for Horndroid
4. Evaluation – Large Scale Analysis

Analysis are time consuming
4. Evaluation – Large Scale Analysis

*Data leaks are present in real world applications*

![Graph showing data leaks across different tools](graph.png)
6. Lessons Learned

User need for Benchmarking tools

- The availability of artefacts in the Android security domain is poor.
- Similar structure does not mean similar performance.
- Benchmarking can leverage base approaches and increase quality of results.
Benchmarking Android Data Leak Detection Tools

Claudio Corradi, Timo Spring, Mohammad Ghafari, and Oscar Nierstrasz
Software Composition Group, University of Bern, Bern, Switzerland

Abstract Security of mobile application available in virtual stores is a concern because platform providers cannot vet every published application. Consequently, many applications—both benign and malign—exhibit security issues such as leaking of sensitive data. In recent years, researchers have proposed a myriad of techniques and tools to detect such issues. However, it is unclear how these approaches perform compared to each other. The tools are often no longer available, thus comparing different approaches is almost infeasible.

In this work, we review approaches for detecting data leak in Android applications. From an initial list of 87 approaches, only 8 could be obtained and executed, and produced results in the selected domain. We compare these using a set of known vulnerabilities and discuss the overall performance of the tools. We further propose an approach to compare security analysis tools by normalizing their interfaces, which simplifies result reproduction and extension.

Keywords: data leak, Android, benchmarking

1 Introduction

Security of mobile applications is a hot topic in both research and industry. With millions of available applications in virtual stores, platform providers such...
Backup
4. Backup – Formulas

Accuracy = $\frac{TP + TN}{TP + TN + FP + FN}$

Precision = $\frac{TP}{TP + FP}$

Recall = $\frac{TP}{TP + FN}$

McNamar’s Test:

$\chi^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}}$

Confidence Interval: 99%
4. Backup – Small Scale Analysis

Custom Configuration reduces number of reported leaks

- **Common Config**
- **Custom Config**