MONASH University

Towards Engineering High-quality and Secure Mobile Apps for Social Good

Li Li
Senior Lecturer,
Monash University
li.li@monash.edu
http://lilicoding.github.io

@lilicoding

2020 Australian Research Council's Discovery Early Career Researcher Awardee

GROUP OF EIGHT AUSTRALIA

IEEE Connecting Experts

IEEE Region 8 Young Professionals

Monash University

Named after Sir John Monash, a famous Australian contributed to almost every level of Australian life.

Monash University

#57 IN THE WORLD

Times Higher Education World University Rankings 2022

#58 IN THE WORLD

QS World University Rankings 2022

#6 IN THE WORLD

Times Higher Education Golden Age University Rankings 2020

#66 IN THE WORLD

QS Graduate Employability Rankings 2020

#45 IN THE WORLD

National Taiwan University Rankings 2020

#1 IN AUSTRALIA

Reuters' Most Innovative Universities in Asia-Pacific 2019

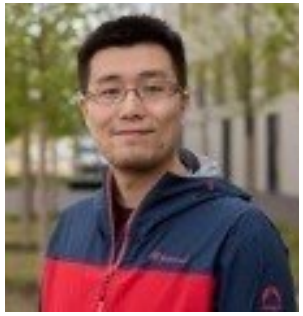
Monash University

Melbourne is today in its 235th day of shutdown since the pandemic began – officially making it the city to have endured the world's longest lockdown.

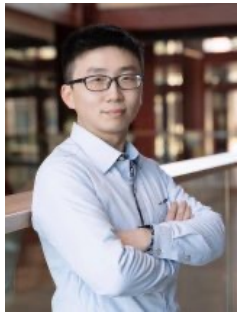
skynews.com.au
Melbourne endures world’s longest lockdown

Collaborative Efforts

Dr. Jun Gao



Dr. Pingfan Kong



Dr. Xiao Chen (Postdoc)

PhD students

- Md. Shamsujjoha
- Xiaoyu Sun
- Yanjie Zhao
- Pei Liu
- Jiawei Wang
- Yue Liu
- Tianming Liu
- Mingyi Zhou
- Yonghui Liu

Monash SMAT team

Luxembourg Trux team, lead by Prof. Jacques Klein

Motivation

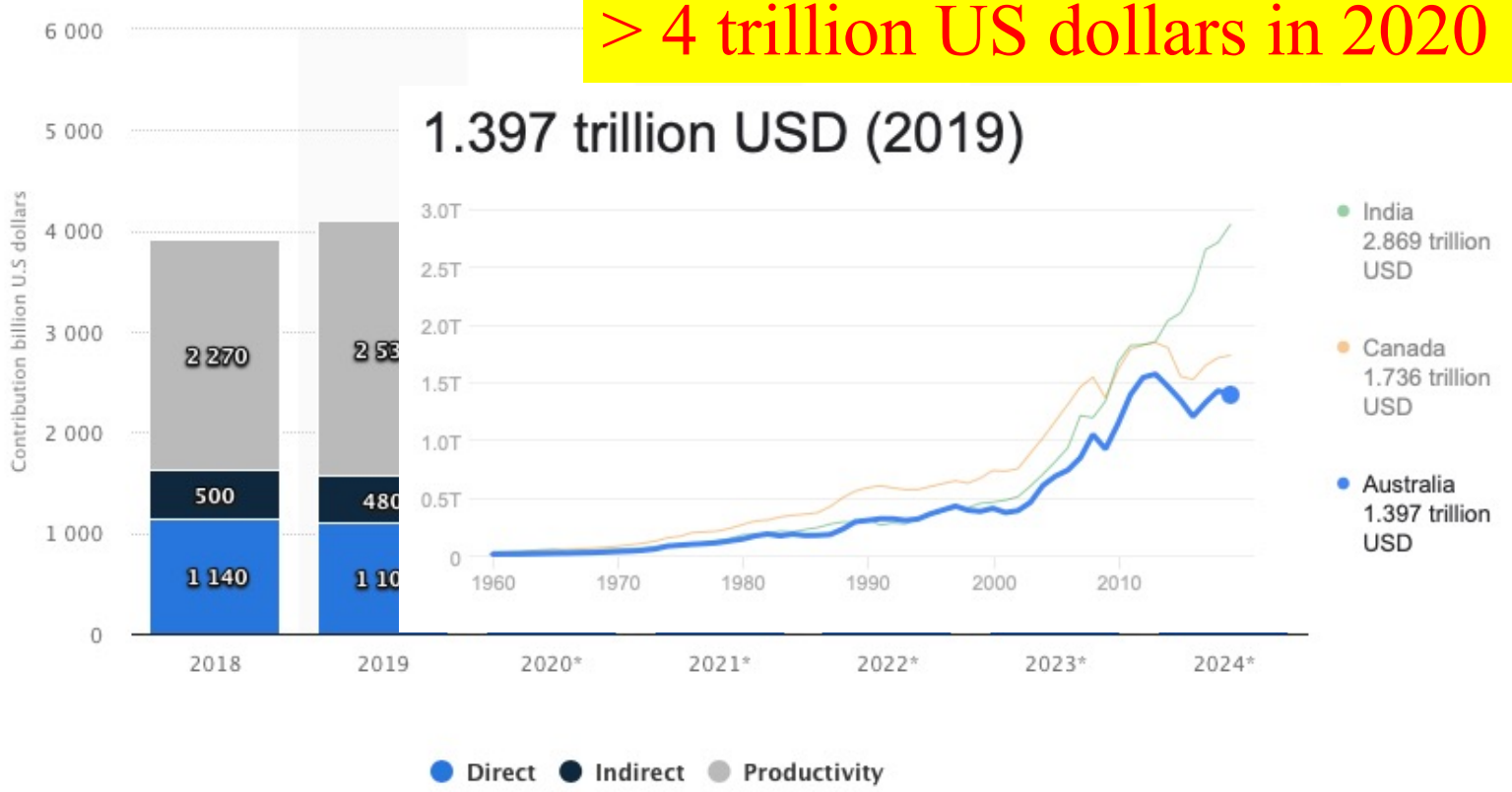
MONASH University

Towards Engineering High-quality and Secure Mobile Apps for Social Good

Motivation

Towards Engineering High-quality and Secure Mobile Apps for Social Good

> 4 trillion US dollars in 2020

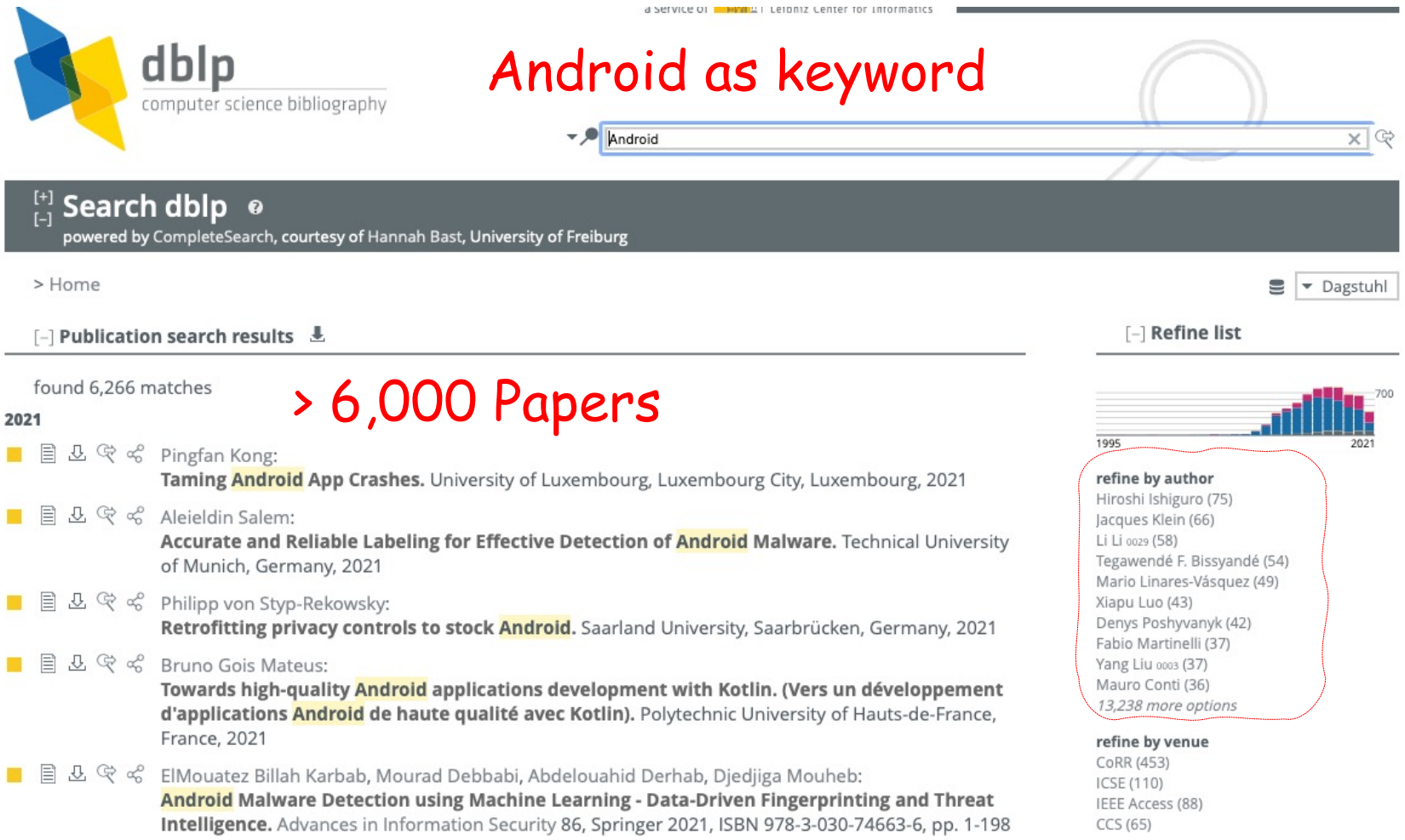


Economic contribution *in billion U.S dollars*

Motivation

MONASH University

Towards Engineering High-quality and Secure Mobile Apps for Social Good

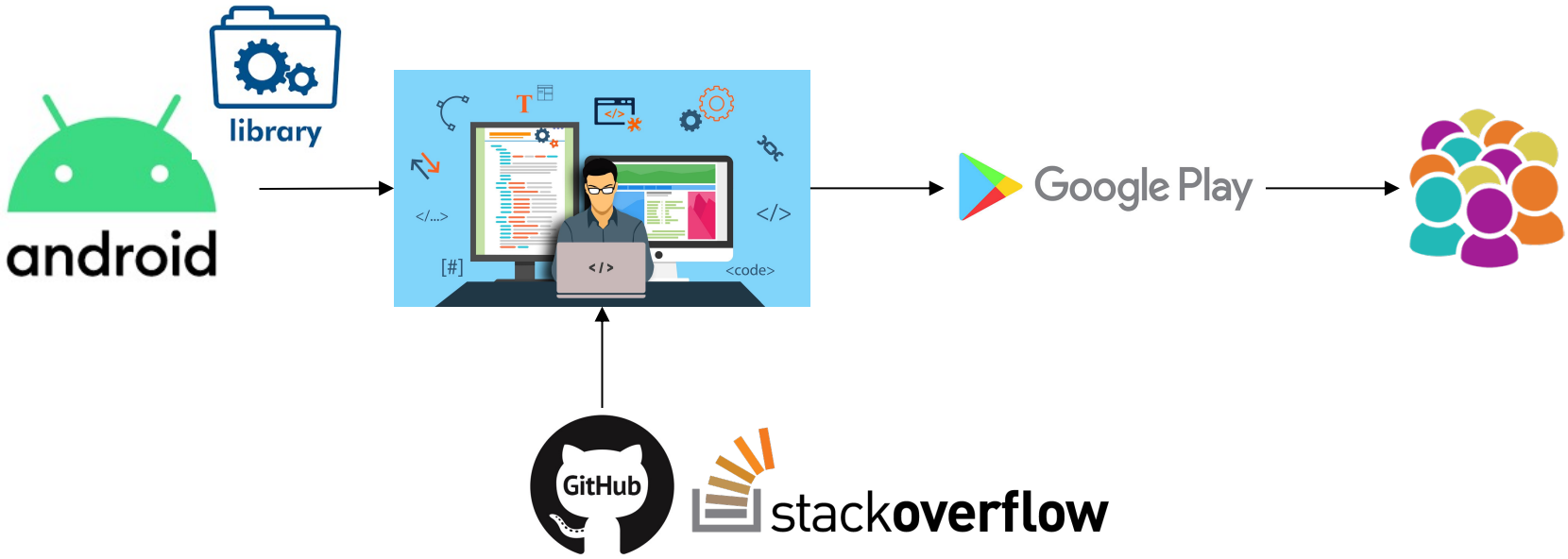


Motivation

MONASH University

Towards Engineering High-quality and Secure Mobile Apps for Social Good

Actions to put human values at the forefront while improving the mobile app ecosystem, including the improvement of app's design, development, testing and maintenance processes.



Motivation

MONASH University

Towards Engineering High-quality and Secure Mobile Apps for Social Good

HUMANISE

HumaniSE Lab

(Human-centric Software Engineering Lab)

Our world-leading research program aims to understand and incorporate the unique and varied aspects of peoples' needs and abilities into software engineering practices. This human-centric approach will lead to new and, most importantly, inclusive software solutions for today's diverse population.

Led by ARC Laureate Professor John Grundy

Motivation

MONASH University

Towards Engineering High-quality and Secure Mobile Apps for Social Good

Australian Government
Australian Research Council

ARC Discovery Project:
Values-oriented Defect Fixing for Mobile Software Applications

A First Look at Human Values-Violation in App Reviews

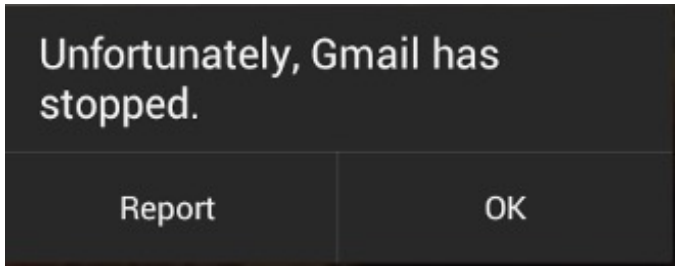
Humphrey O. Obie*, Waqar Hussain*, Xin Xia*, John Grundy*, Li Li*, Burak Turhan * †, Jon Whittle‡, Mojtaba Shahin*
*Monash University, Melbourne, Australia
†University of Oulu, Oulu, Finland
‡CSIRO's Data61, Melbourne, Australia
{humphrey.obie, waqar.hussain, xin.xia, john.grundy, li.li, burak.turhan, mojtaba.shahin}@monash.edu, jon.whittle@data61.csiro.au

Humphrey O. Obie, Waqar Hussain, Xin Xia, John Grundy, Li Li, Burak Turhan, Jon Whittle and Mojtaba Shahin, A First Look at Human Values-Violation in App Reviews, ICSE, SEIS Track, 2021

Outline

MONASH University

Towards Engineering High-quality and Secure Mobile Apps for Social Good



Methodology

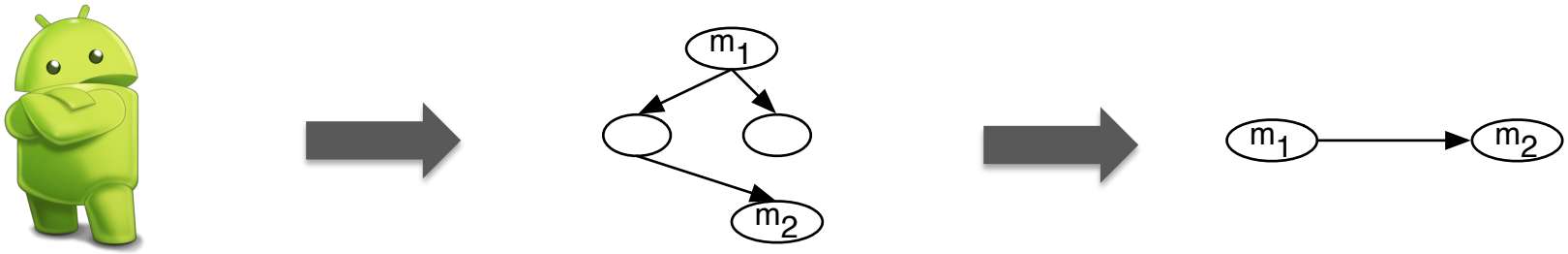
Static Analysis

Dynamic Analysis

Machine/Deep Learning

Mobile Ecosystem

MONASH University



Methodology

Static Analysis

Dynamic Analysis

Machine/Deep Learning

Mobile Ecosystem

MONASH University

Static Analysis of Android Apps: A Systematic Literature Review

Li Li[a,1], Tegawendé F. Bissyandé[a], Mike Papadakis[a], Siegfried Rasthofer[b], Alexandre Bartel[a,2], Damien Octeau[c], Jacques Klein[a], Yves Le Traon[a]

[a] *Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg*
[b] *Fraunhofer SIT, Darmstadt, Germany*
[c] *University of Wisconsin and Pennsylvania State University*

Abstract

Context: Static analysis exploits techniques that parse program source code or bytecode, often traversing program paths to check some program properties. Static analysis approaches have been proposed for different tasks, including for assessing the security of Android apps, detecting app clones, automating test cases generation, or for uncovering non-functional issues related to performance or energy. The literature thus has proposed a large body of works, each of which attempts to tackle one or more of the several challenges that program analysers face when dealing with Android apps.

Objective: We aim to provide a clear view of the state-of-the-art works that statically analyse Android apps, from which we highlight the trends of static analysis approaches, pinpoint where the focus has been put, and enumerate the key aspects where future researches are still needed.

Methodology

Static Analysis

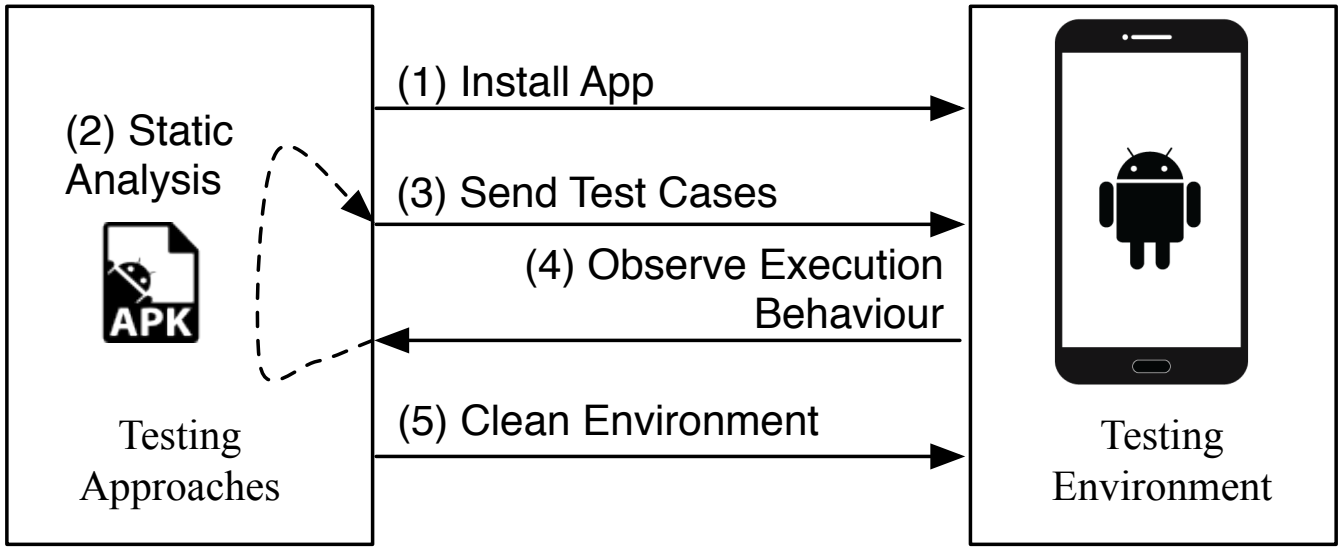
Dynamic Analysis

Machine/Deep Learning

Li Li, Tegawendé F. Bissyandé, Mike Papadakis, Siegfried Rasthofer, Alexandre Bartel, Damien Octeau, Jacques Klein and Yves Le Traon, Static Analysis of Android Apps: A Systematic Literature Review, Information and Software Technology (IST), 2017

Mobile Ecosystem

MONASH University



Methodology

Static Analysis

Dynamic Analysis

Machine/Deep Learning

Mobile Ecosystem

MONASH University

Automated Testing of Android Apps: A Systematic Literature Review

Pingfan Kong, Li Li[ξ], Jun Gao, Kui Liu, Tegawendé F. Bissyandé, Jacques Klein

Abstract—Automated testing of Android apps is essential for app users, app developers and market maintainer communities alike. Given the widespread adoption of Android and the specificities of its development model, the literature has proposed various testing approaches for ensuring that not only functional requirements but also non-functional requirements are satisfied. In this paper, we aim at providing a clear overview of the state-of-the-art works around the topic of Android app testing, in an attempt to highlight the main trends, pinpoint the main methodologies applied and enumerate the challenges faced by the Android testing approaches as well as the directions where the community effort is still needed. To this end, we conduct a Systematic Literature

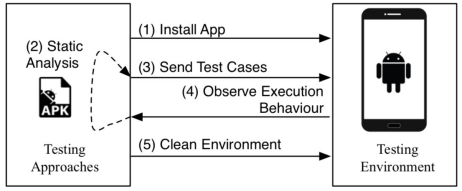


Fig. 1: Process of testing Android apps.

Methodology

Static Analysis

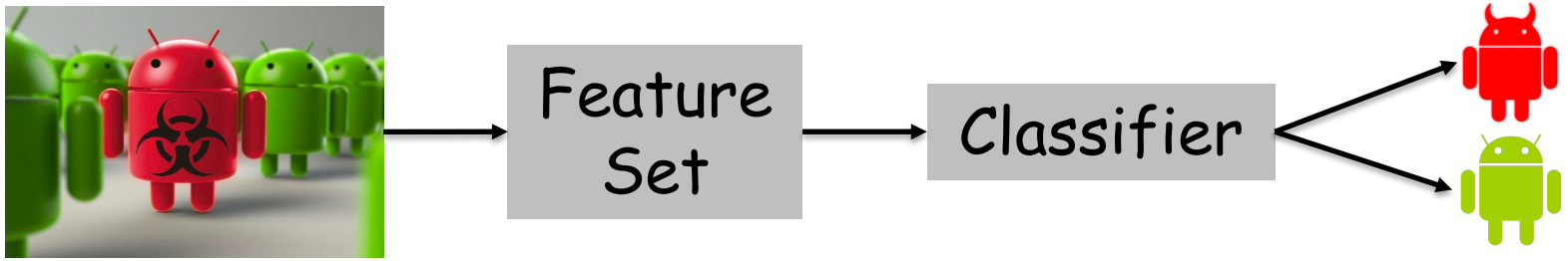
Dynamic Analysis

Machine/Deep Learning

Pingfan Kong, Li Li, Jun Gao, Kui Liu, Tegawendé Bissyandé, Jacques Klein, Automated Testing of Android Apps: A Systematic Literature Review, IEEE Transactions on Reliability (TRel), 2018

Mobile Ecosystem

MONASH University



Methodology

Static Analysis

Dynamic Analysis

Machine/Deep Learning

Mobile Ecosystem

MONASH University

Deep Learning for Android Malware Defenses: a Systematic Literature Review

YUE LIU, CHAKKRIT TANTITHAMTHAVORN, and LI LI, Monash University, Australia
YEPANG LIU, Southern University of Science and Technology, China

Malicious applications (especially in the Android platform) are a serious threat to developers and end-users. Many research efforts have hence been devoted to developing effective approaches to defend Android malware. However, with the explosive growth of Android malware and the continuous advancement of malicious evasion technologies like obfuscation and reflection, android malware defenses based on manual rules or traditional machine learning may not be effective due to limited apriori knowledge. In recent years, a dominant research field of deep learning (DL) with the powerful feature abstraction ability has demonstrated a compelling and promising performance in various fields, like Nature Language processing and image processing. To this end, employing deep learning techniques to thwart the attack of Android malware has recently gained considerable research attention. Yet, there exists no systematic literature review that focuses on deep learning approaches for Android Malware defenses. In this paper, we conducted a systematic literature review to search and analyze how deep learning approaches have been applied in the context of malware defenses in the Android environment. As a result, a total of 104 studies were identified over the period 2014-2020. The results of our investigation show that even though most of these studies still mainly consider DL-based on Android malware detection, 35 primary studies (33.7%) design the defenses approaches based on other scenarios. This review also describes research trends, research focuses, challenges, and future research directions in DL-based Android malware defenses.

Methodology

Static Analysis

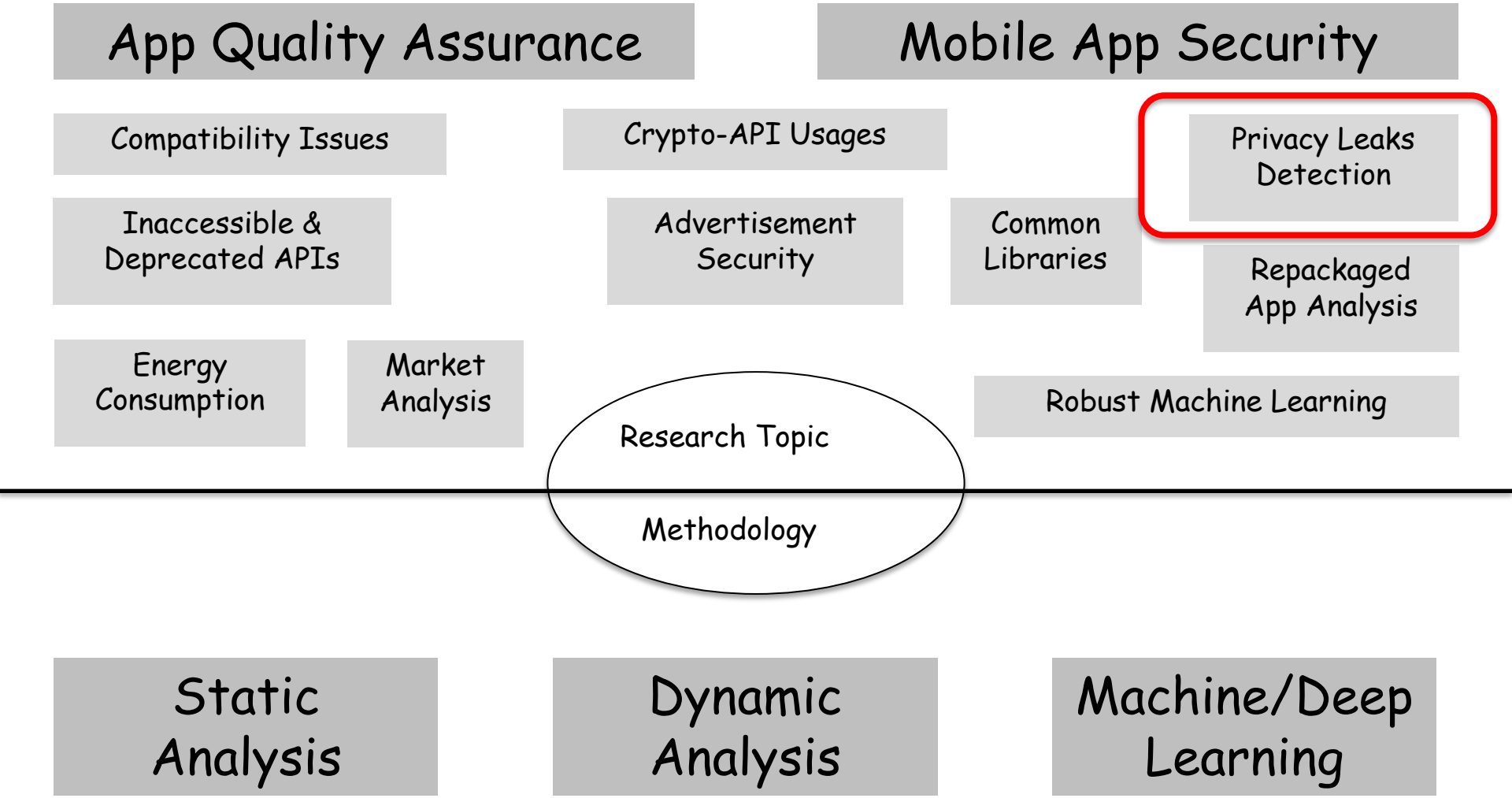
Dynamic Analysis

Machine/Deep Learning

Yue Liu, Chakkrit Tantithamthavorn, Li Li, Yepang Liu, Deep Learning for Android Malware Defenses: a Systematic Literature Review, arXiv preprint arXiv:2103.05292, 2021

Mobile Software Engineering

MONASH University



Privacy Leaks Detection

MONASH University

Prof. Eric Bodden
University of Paderborn

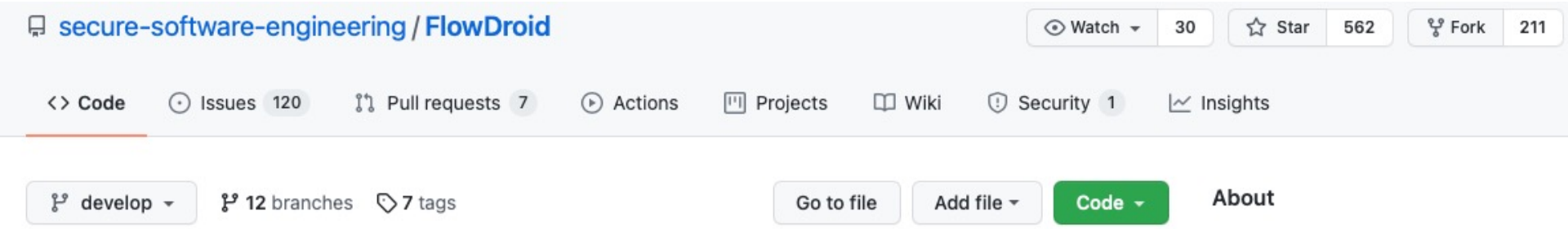
FlowDroid: Precise Context, Flow, Field, Object-sensitive and Lifecycle-aware Taint Analysis for Android Apps

PLDI 2014

Steven Arzt, Siegfried Rasthofer, Christian Fritz, Eric Bodden
EC SPRIDE
Technische Universität Darmstadt
firstName.lastName@ec-spride.de

Alexandre Bartel, Jacques Klein, and Yves Le Traon
Interdisciplinary Centre for Security, Reliability and Trust
University of Luxembourg
firstName.lastName@uni.lu

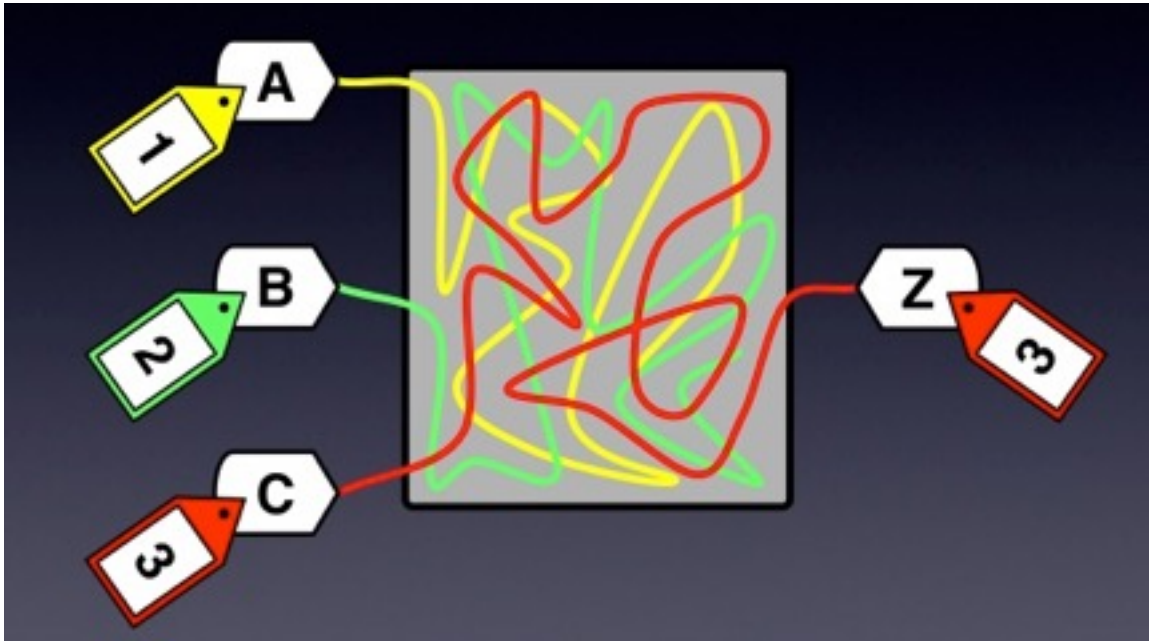
Damien Octeau, Patrick McDaniel
Department of Computer Science and Engineering
Pennsylvania State University
{octeau,mcdaniel}@cse.psu.edu



Privacy Leaks Detection

MONASH University

Uncovering Privacy Leaks via Static Taint Analysis

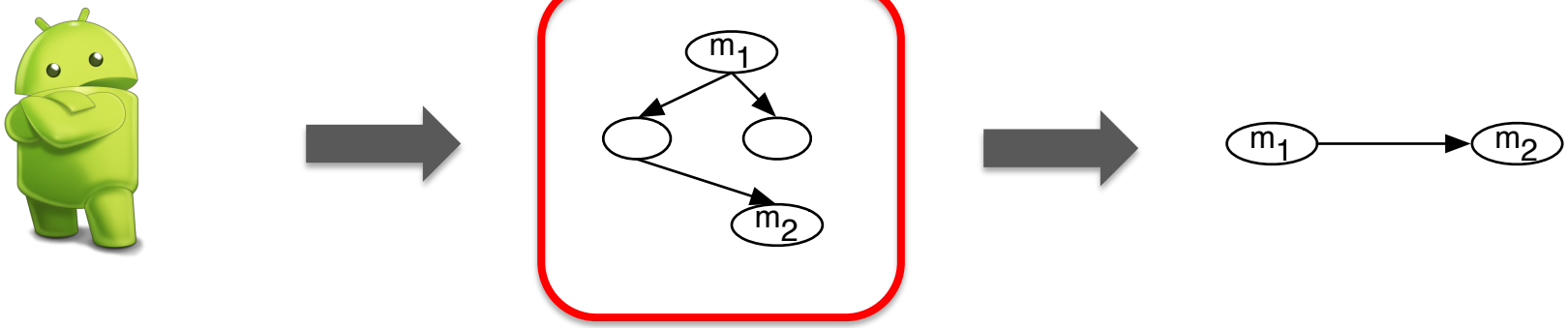


source

sink

Privacy Leaks Detection

MONASH University



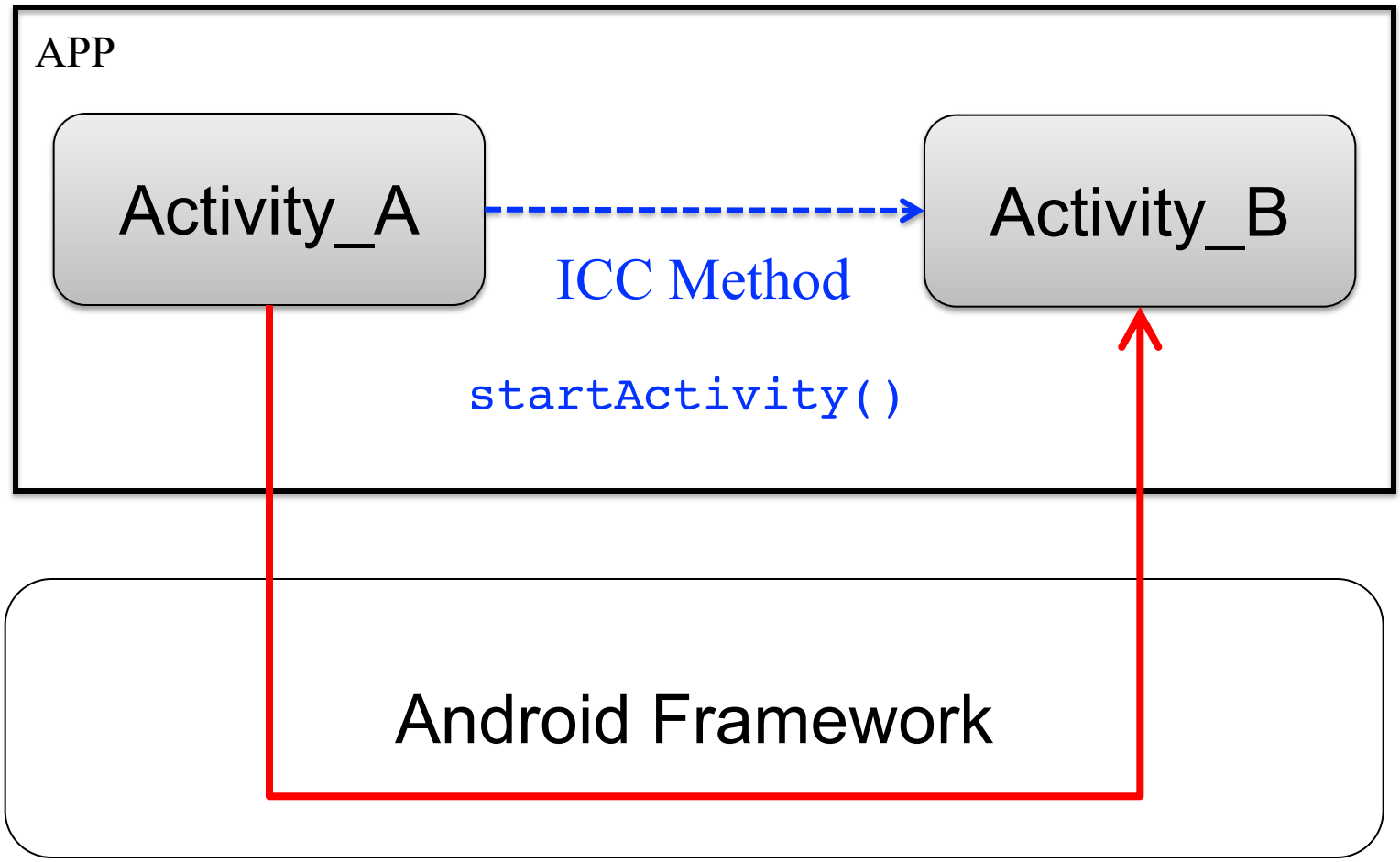
Build Sound Call Graph

Inter-Component Communication (ICC)

Reflection

Privacy Leaks Detection

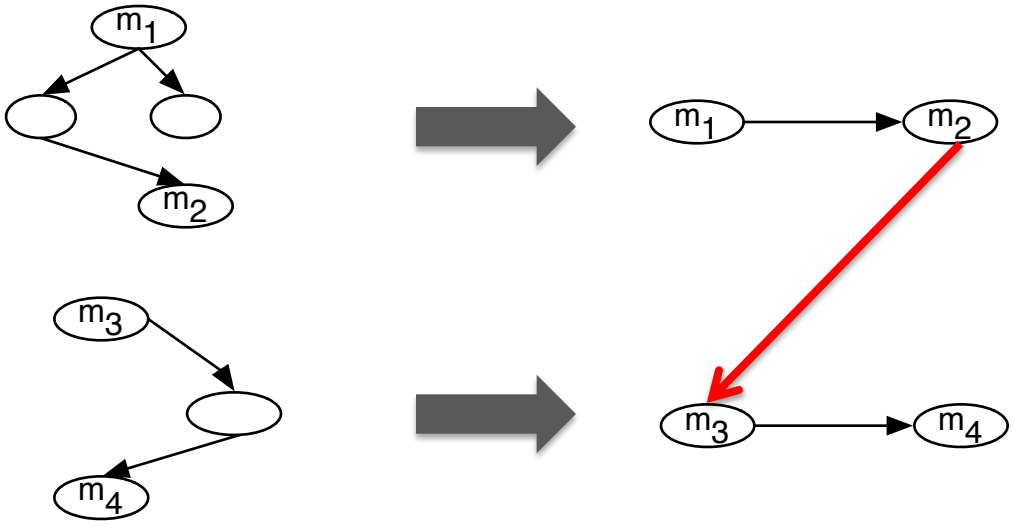
MONASH University



Privacy Leaks Detection

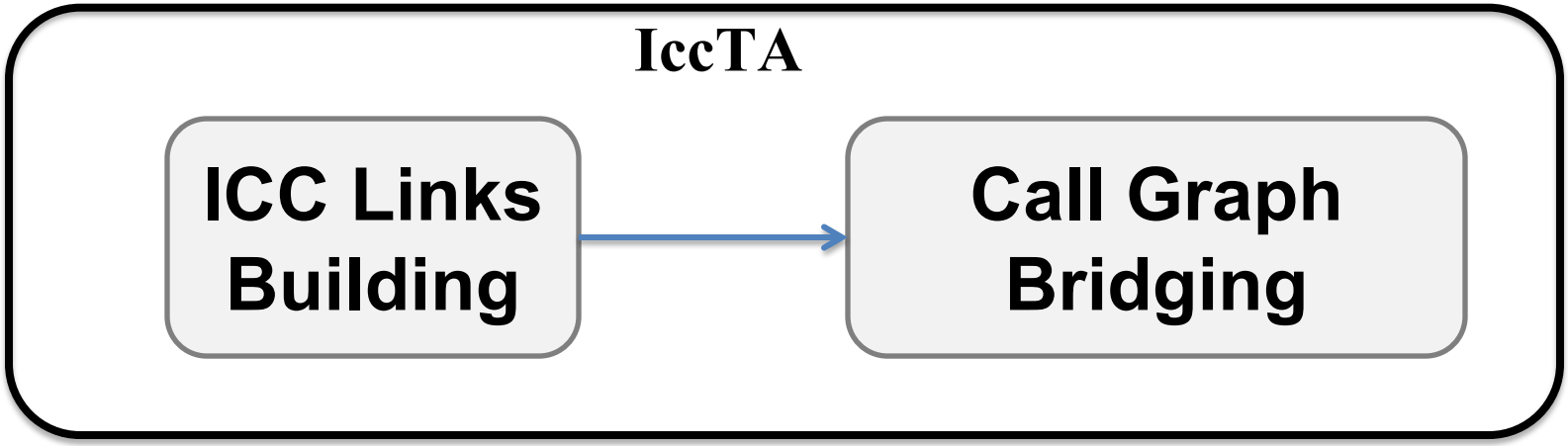
MONASH University

Activity_A

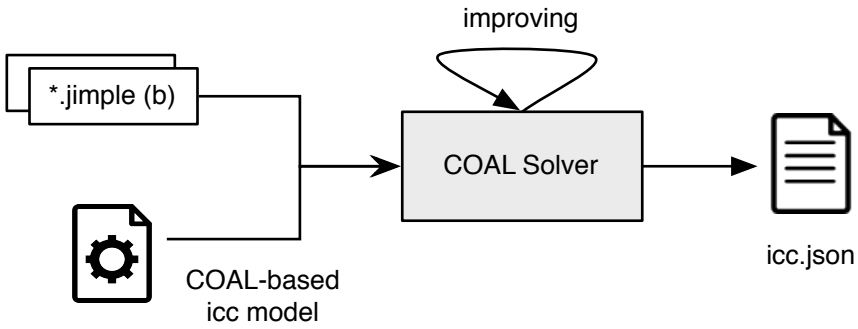


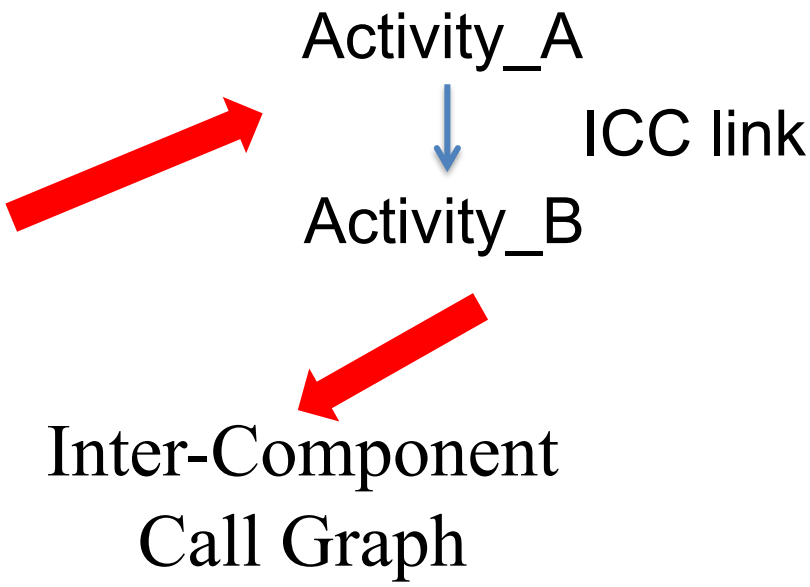
Privacy Leaks Detection

MONASH University



Constant String Propagation

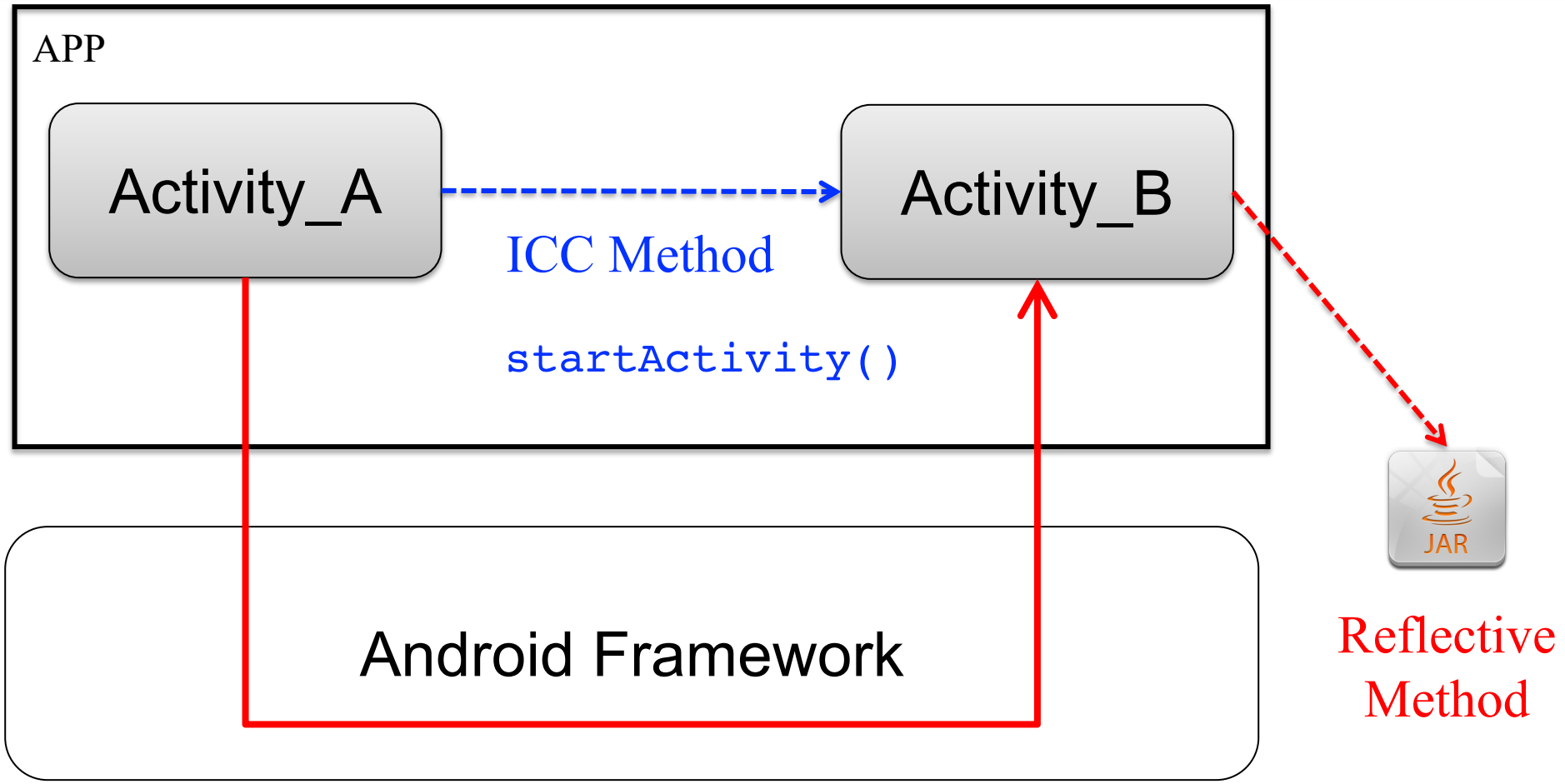




Li Li, Alexandre Bartel, Tegawendé F. Bissyandé Jacques Klein, Yves Le Traon, Steven Arzt, Siegfried Rasthofer, Eric Bodden, Damien Octeau and Patrick McDaniel, IccTA: Detecting Inter-Component Privacy Leaks in Android Apps, ICSE 2015

Privacy Leaks Detection

MONASH University



Privacy Leaks Detection

MONASH University

DroidRA

Constant String Propagation

(1) To infer target values of reflective calls.

Object o = c.newInstance(); → **o = new ReflectiveClass();**

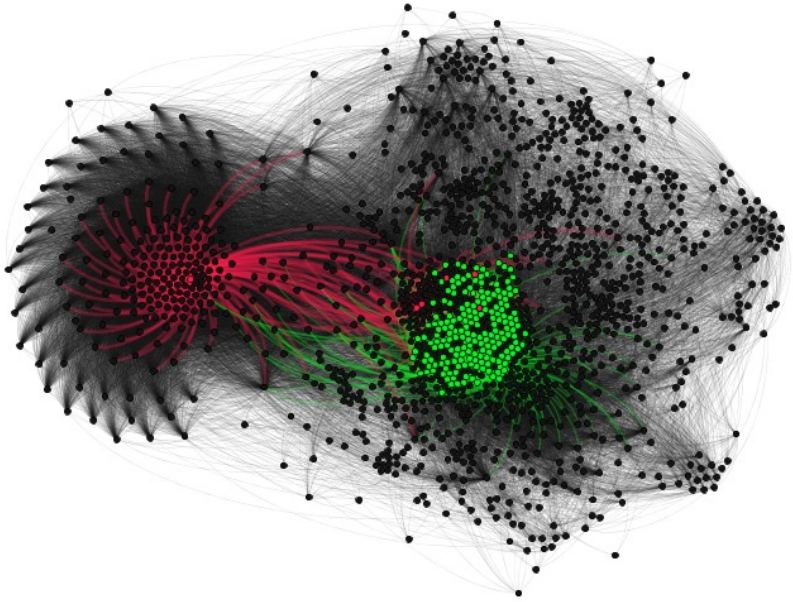
`m.invoke(o, imei);` → **`o.setImei(imei);`**

(2) To replace reflective calls with traditional Java calls.

Privacy Leaks Detection

MONASH University

DroidRA



Case Study:
Org.bl.cadone (CHA)

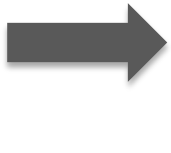
Green: new introduced
Red: permission-protected

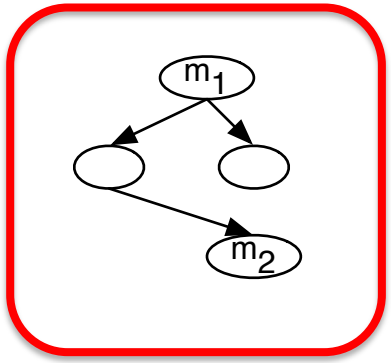
Yanjie Zhao, Li Li, Haoyu Wang, Haipeng Cai, Tegawendé F. Bissyandé, Jacques Klein and John Grundy, On the Impact of Sample Duplication in Machine Learning based Android Malware Detection, ACM TOSEM 2021

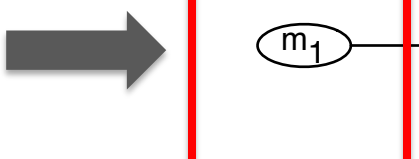
Li Li, Tegawendé F. Bissyandé, Damien Octeau and Jacques Klein, DroidRA: Taming Reflection to Support Whole-Program Analysis of Android Apps, The 2016 International Symposium on Software Testing and Analysis (ISSTA 2016), 2016

Privacy Leaks Detection

MONASH University







Build Sound Call Graph

Sensitive Fields

Inter-Component Communication (ICC)

Reflection

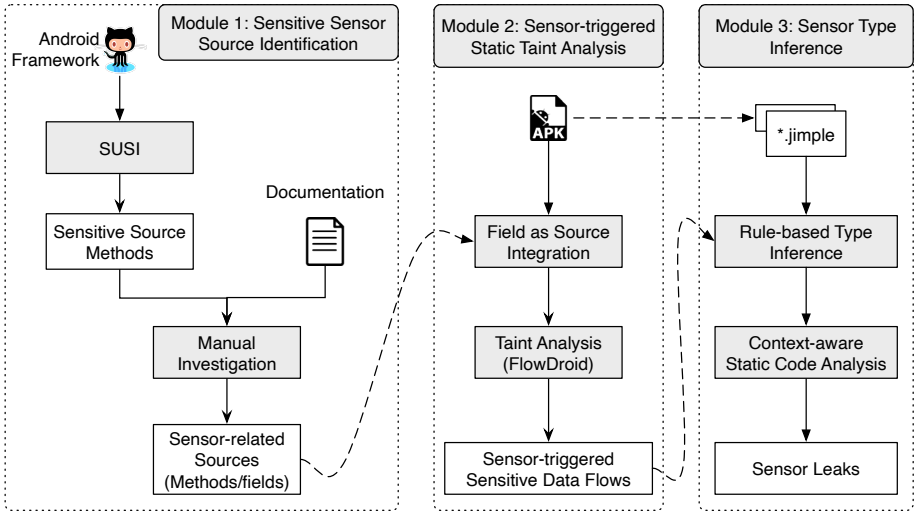
Privacy Leaks Detection

MONASH University

Improve FlowDroid to support Field Sources(develop branch) #385

Merged StevenArzt merged 6 commits into `secure-software-engineering:develop` from `MobileSE:develop` 4 days ago

Conversation 2 Commits 6 Checks 0 Files changed 5



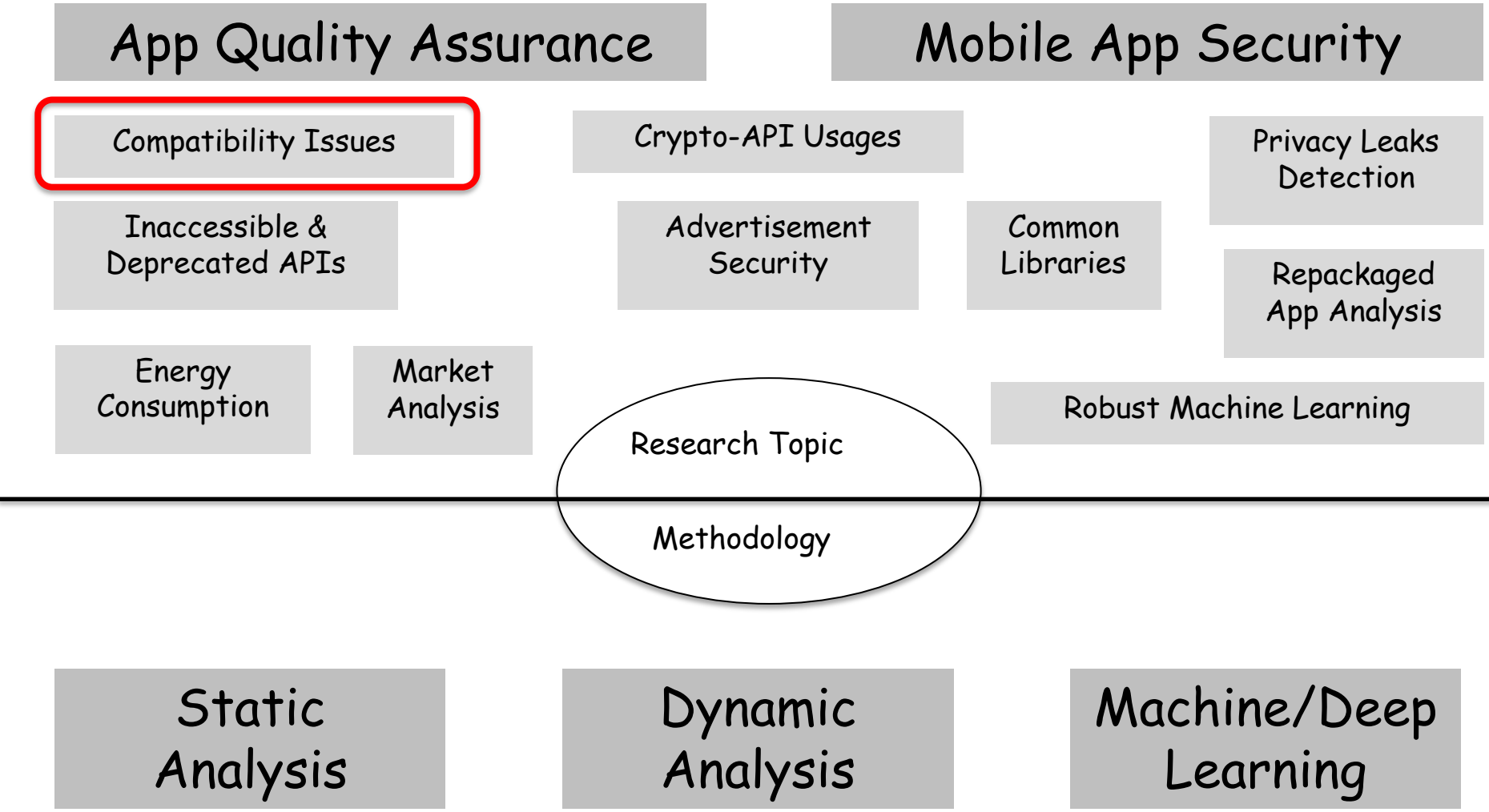
SEEKER

Characterizing Sensor Leaks in Android Apps

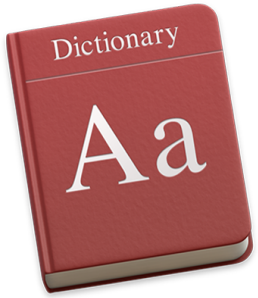
Xiaoyu Sun, Xiao Chen, Kui Liu, Sheng Wen, **Li Li** and John Grundy, Characterizing Sensor Leaks in Android Apps, The 32nd International Symposium on Software Reliability Engineering (ISSRE 2021), 2021

Mobile Software Engineering

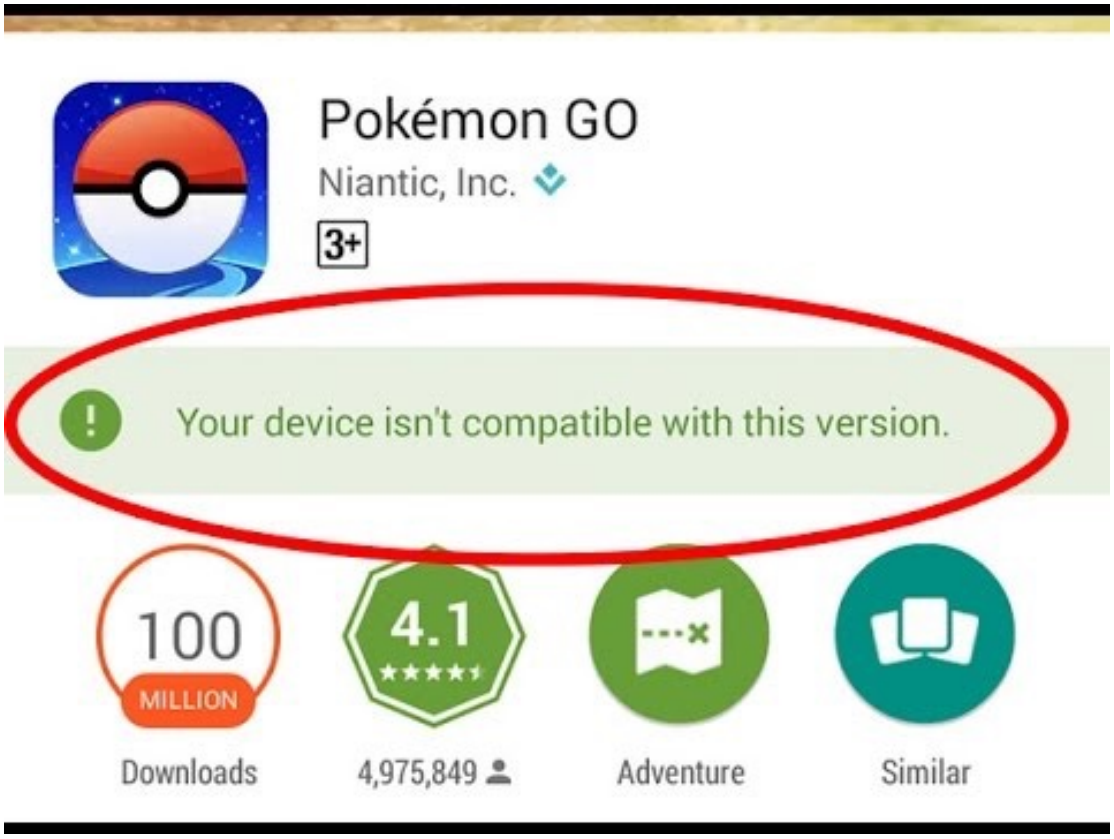
MONASH University



Compatibility Issues

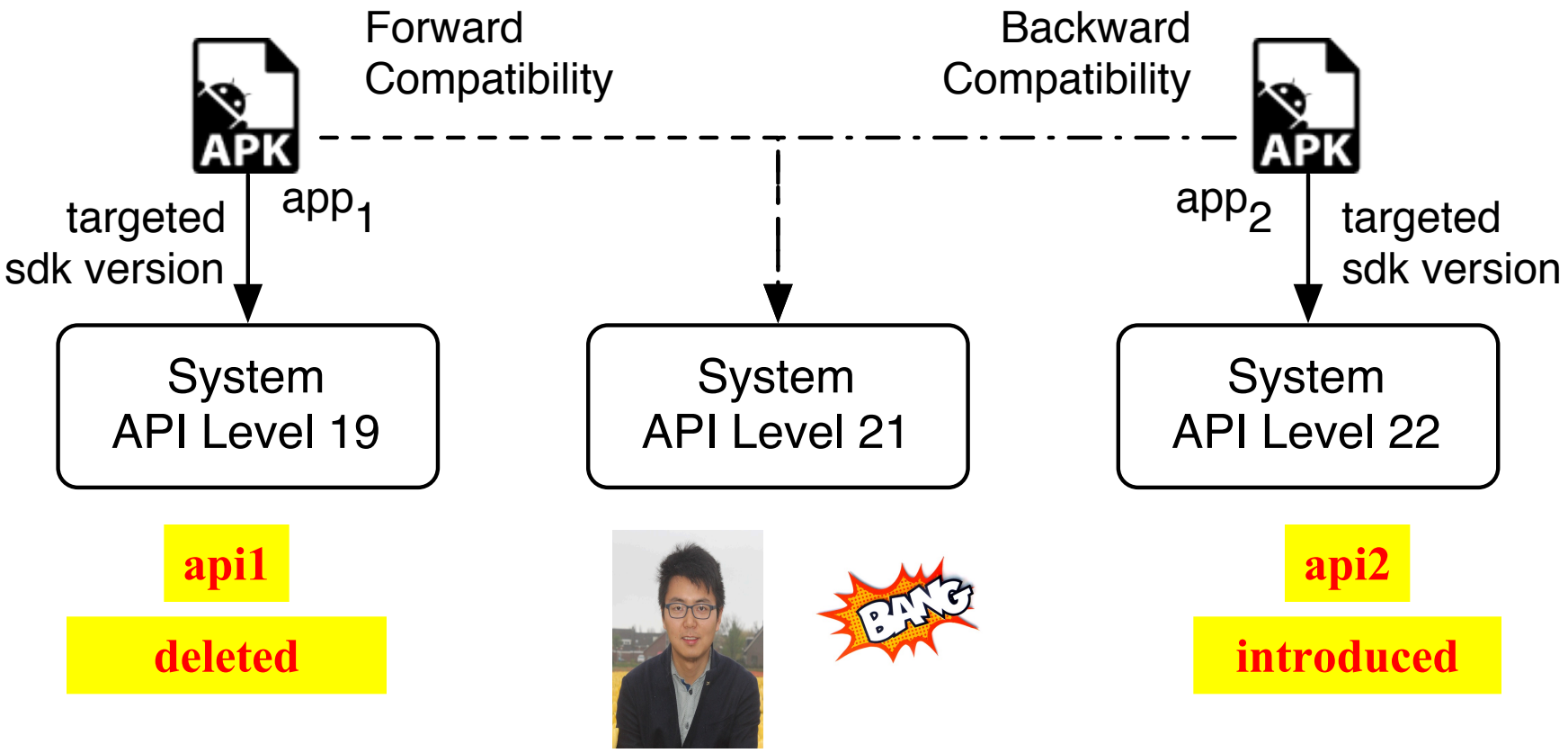


Compatibility is a state in which two things are able to exist or occur together without problems or conflict

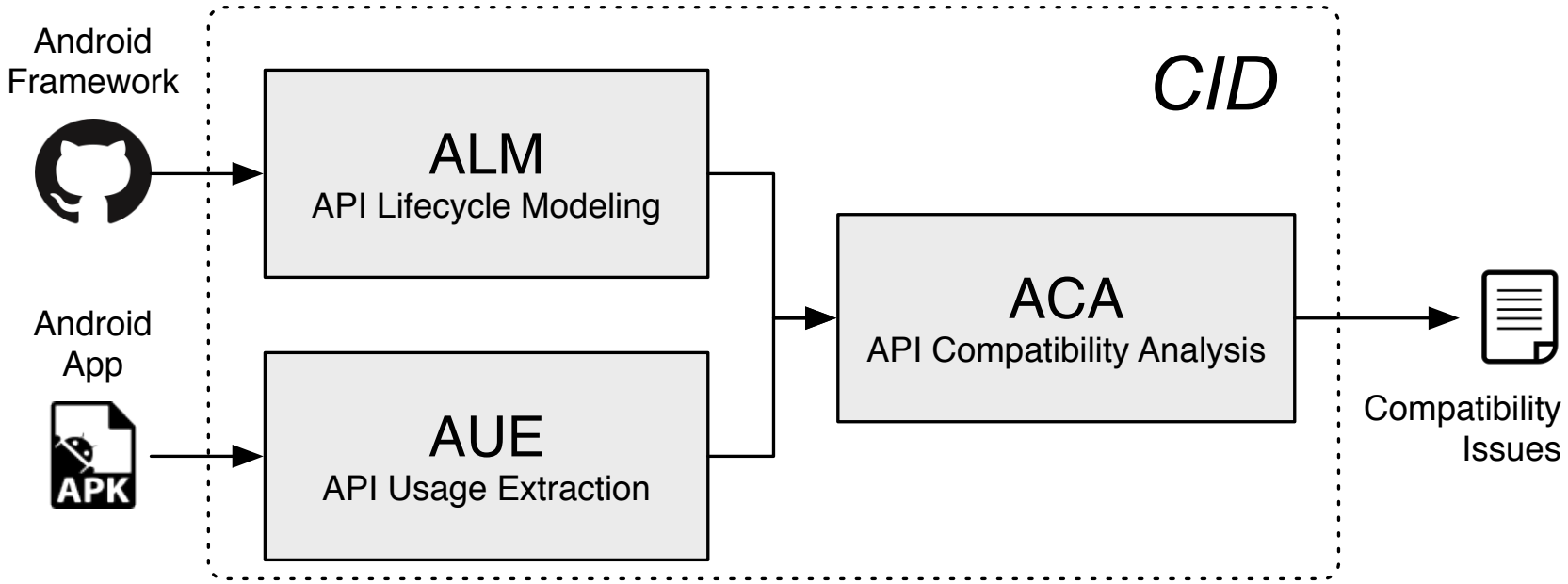


Compatibility Issues

MONASH University



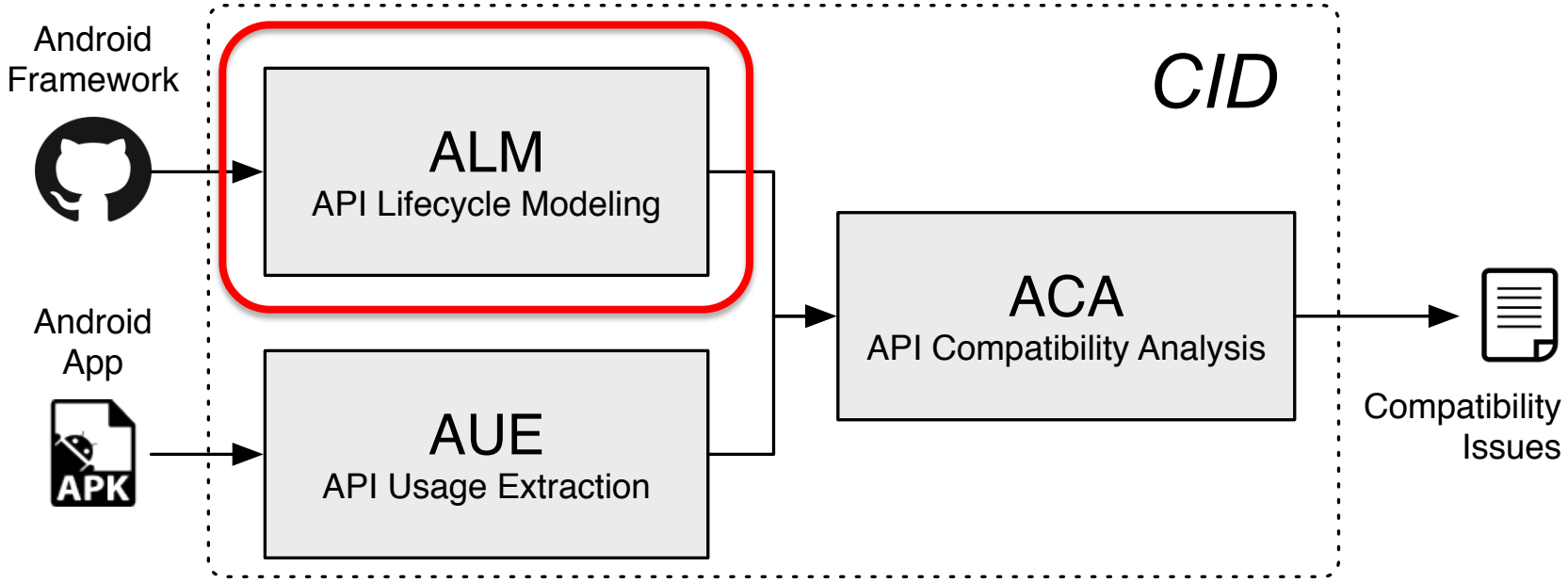
Compatibility Issues



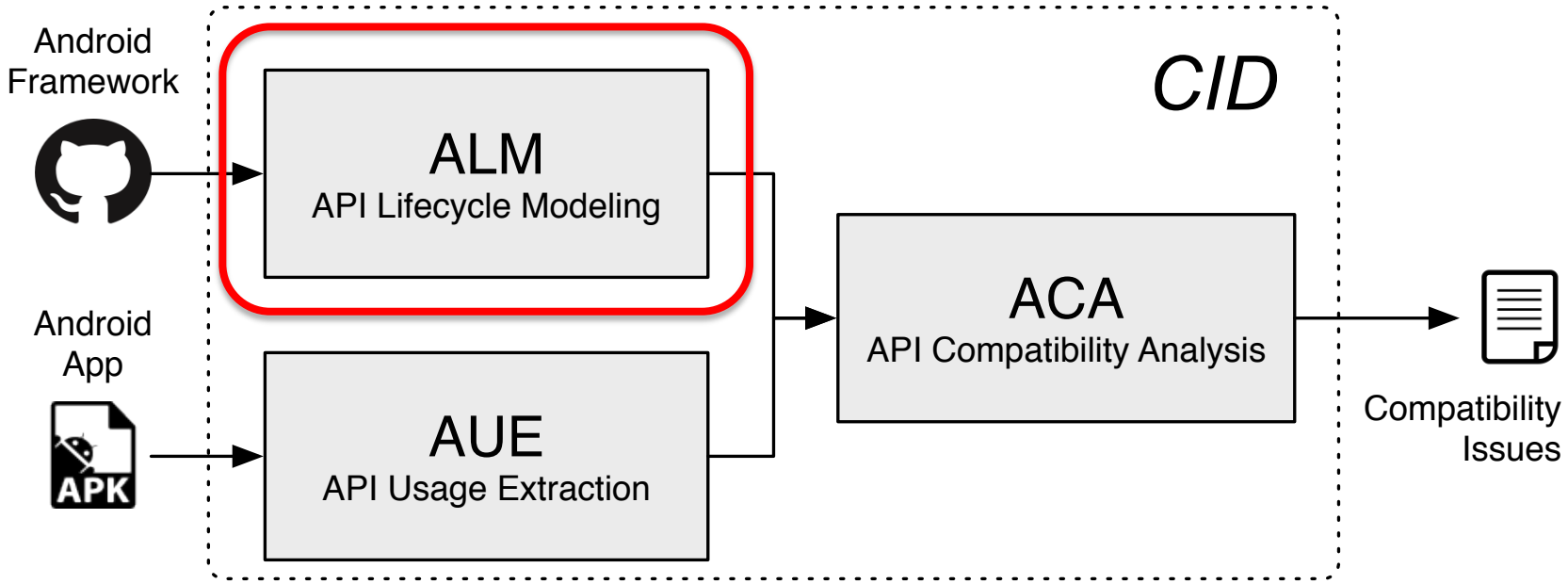
CID aims at flagging potential API compatibility issues and helping developers understand better the identified compatibility issues.

Compatibility Issues

MONASH University

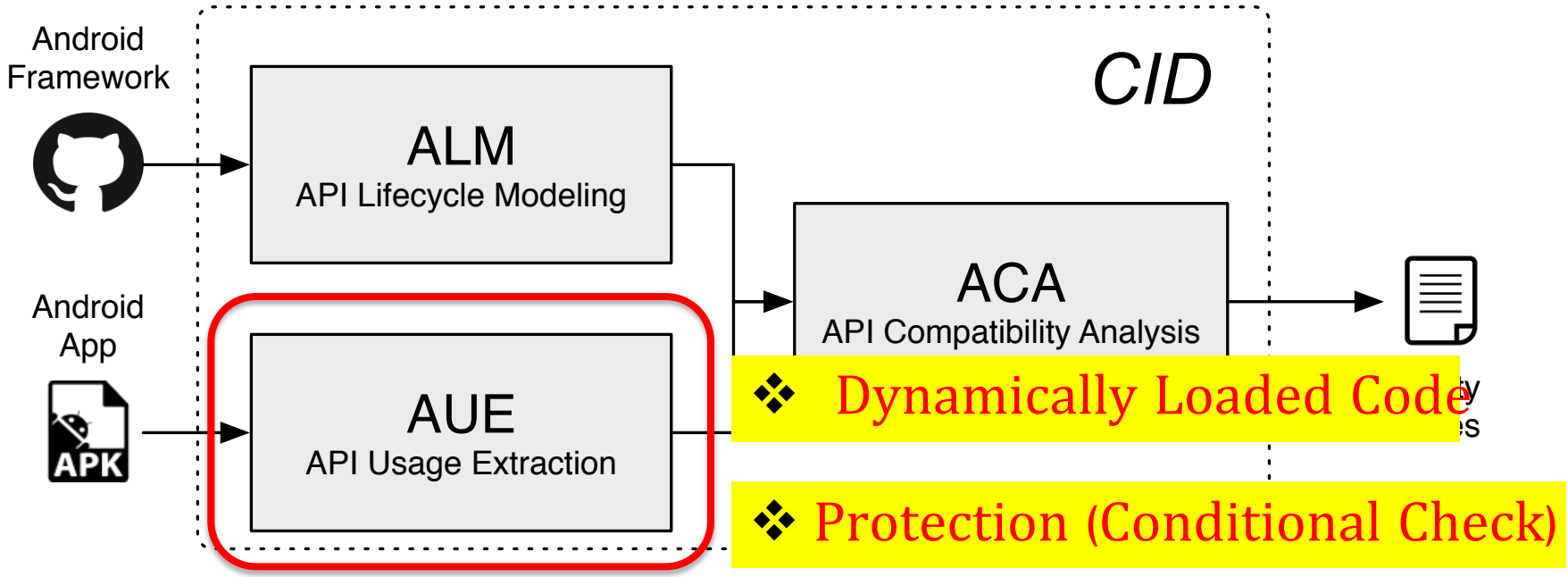


Compatibility Issues



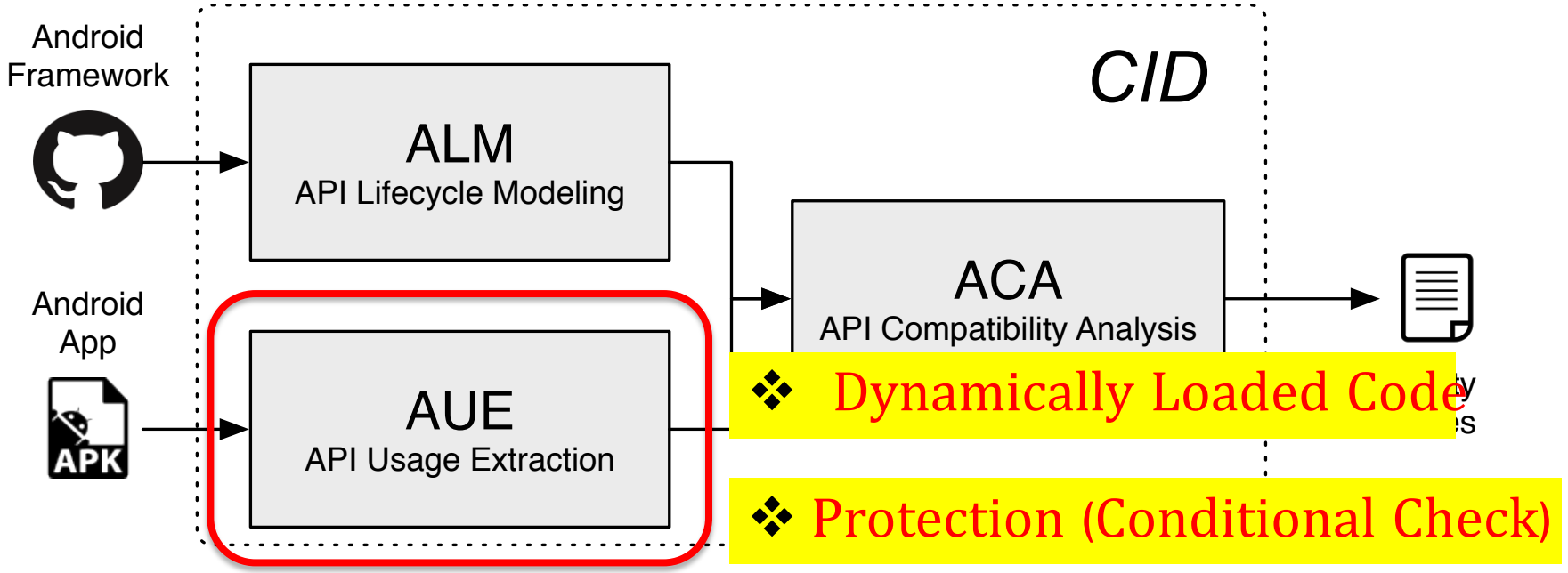

```
<android.nfc.NdefRecord: void (short,byte[],byte[],byte[])>:[9,25]
<android.app.backup.BackupAgentHelper: void
addHelper(java.lang.String,android.app.backup.BackupHelper)>:[8,25]
<android.content.SharedPreferences.Editor: void apply()>:[9,25]
<android.os.StrictMode.ThreadPolicy.Builder:
android.os.StrictMode.ThreadPolicy.Builder detectAll()>:[9,25]
```

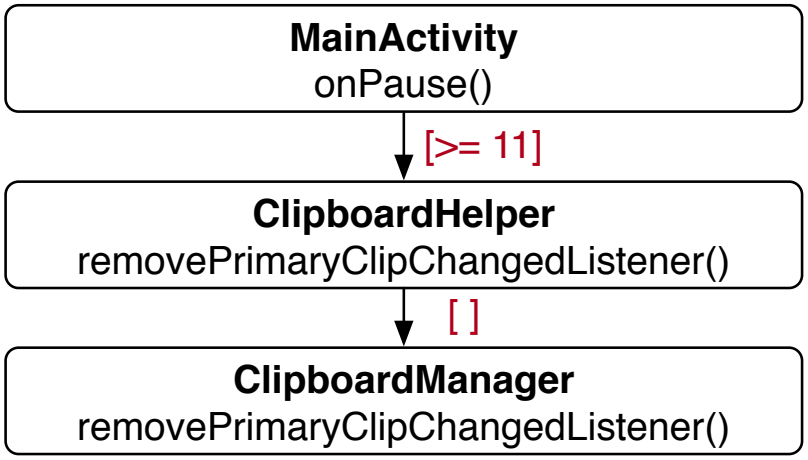
Compatibility Issues




```
if (Build.VERSION.SDK_INT >= 11 && clipboardListener != null) {
  clipboardHelper.removeClipboardListener(clipboardListener);
}
public void removeClipboardListener(Runnable runnable) {
  if (runnable != null && listeners.containsKey(runnable)) {
    //removePrimaryClipChangedListener() is introduced at level 11
    clipboard.removePrimaryClipChangedListener( listeners.get(runnable));
  }
}
```

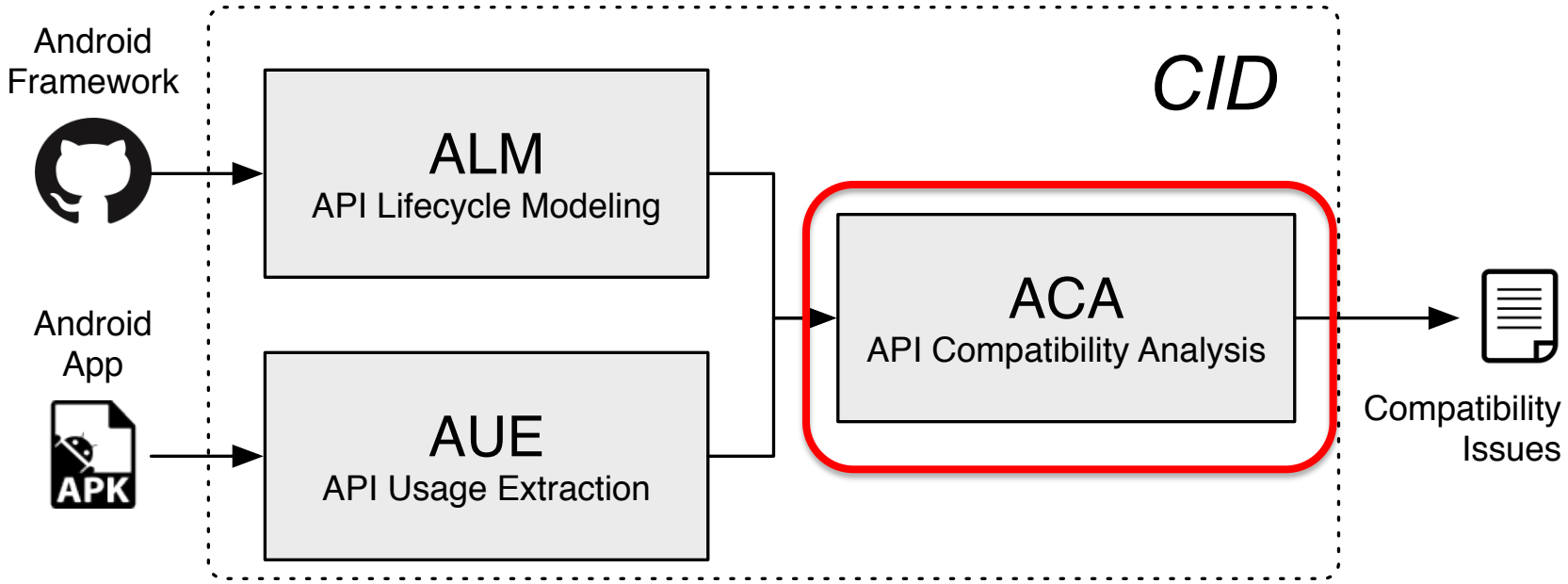
Compatibility Issues





CCG: Conditional Call Graph

Compatibility Issues




```
//Generic Programming
<java.util.LinkedList: E set(int,E)>
<java.util.LinkedList: String set(int,String)>
<java.util.LinkedList: String set(int,Double)>

//Varargs
<android.app.DownloadManager: int remove(long...)>
<android.app.DownloadManager: int remove(long)>
<android.app.DownloadManager: int remove(long,long)>
```

Compatibility Issues

MONASH University

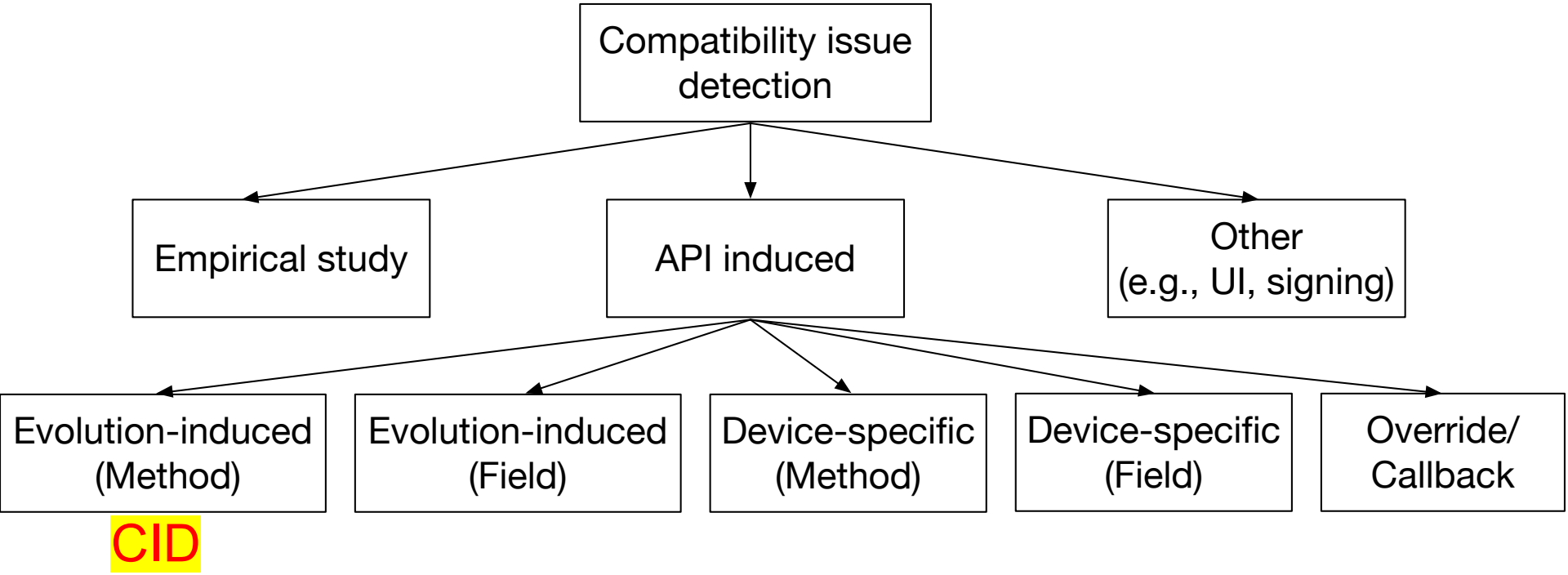
App Name	# Commits	Dex Size	Issue ID(s)	Fix Commit
Rabbit Escape	1,785	1,872 K	478	-
ShareViaHttp	167	1,545 K	24	9ed54f
FRCAndroidWidget	285	320 K	32,33	fc0364
Nextcloud Notes	240	4,137 K	177	-
DragonGoApp	180	1,185 K	13	-
EnigmAndroid	54	1,792 K	9	-
Red Screen	6	4 K	2	31a560

CID can provide helpful information for developers to quickly understand, evaluate and eventually fix API compatibility issues that their apps may encounter.

Compatibility Issues

MONASH University

Literature Review

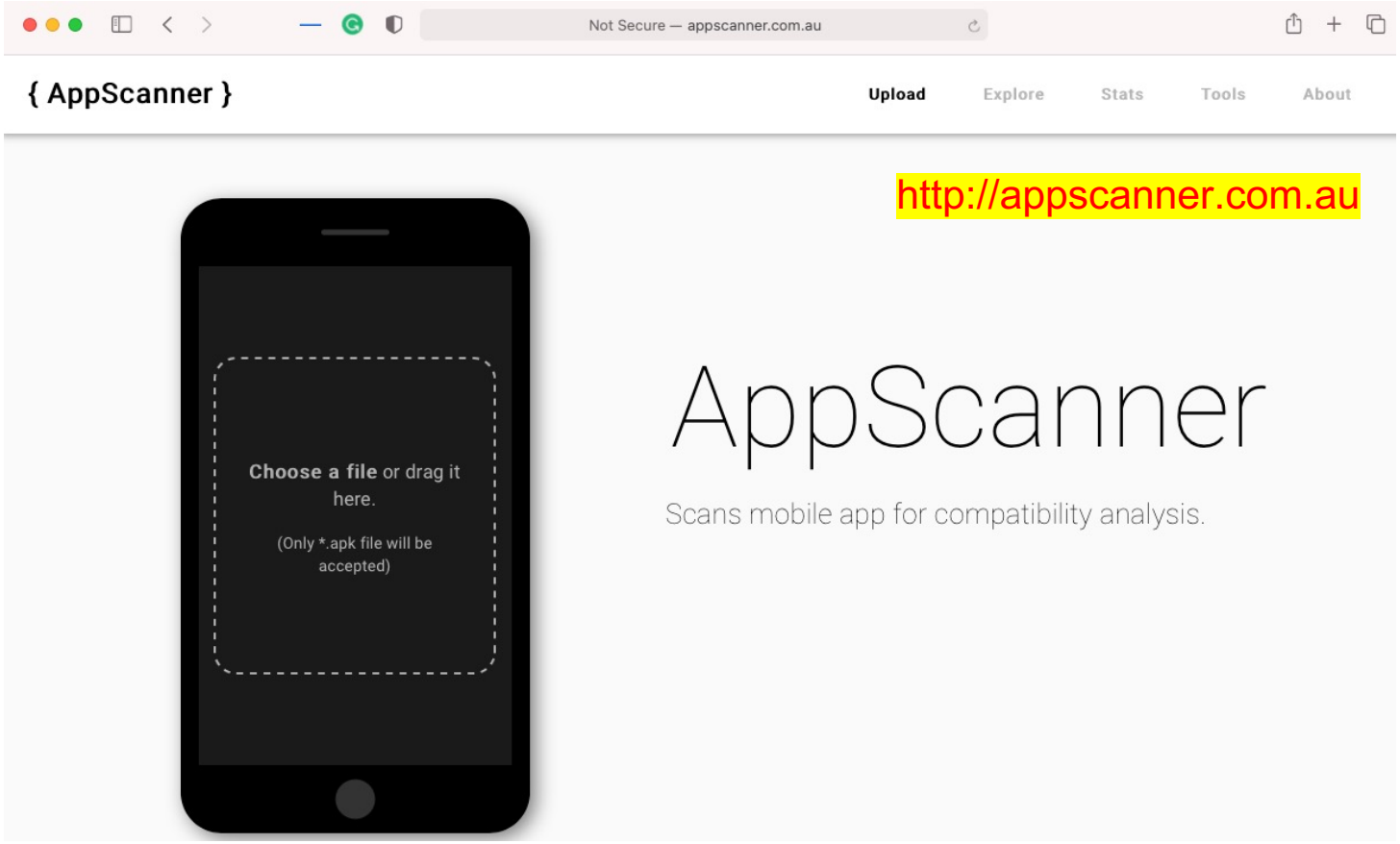


Compatibility Issues

MONASH University

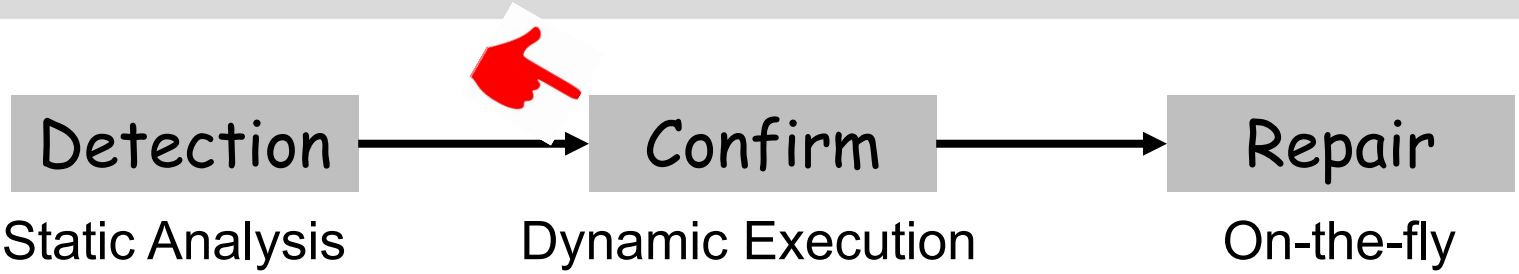
Detection → Confirm → Repair

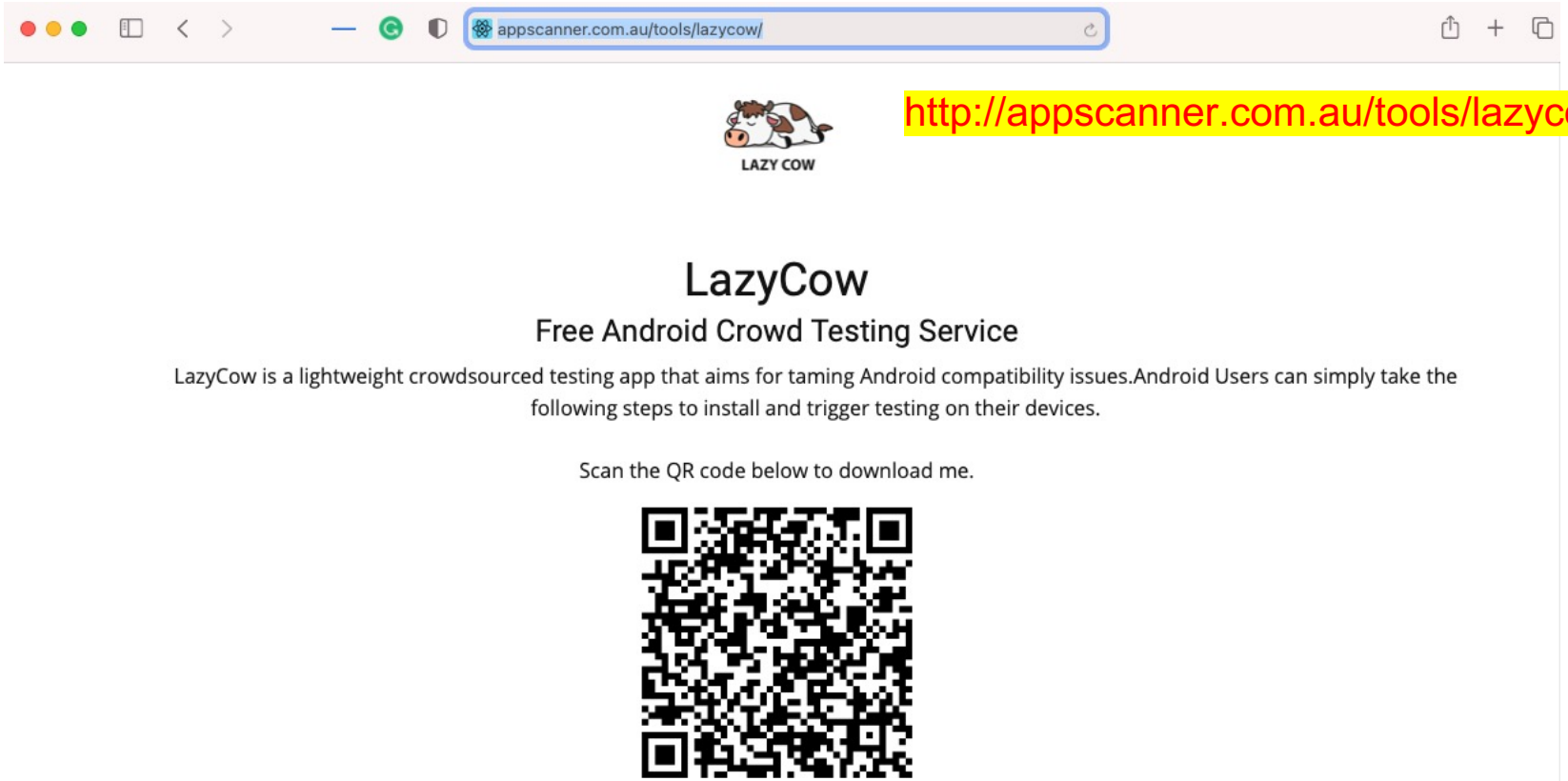
Static Analysis — Dynamic Execution — On-the-fly



Compatibility Issues

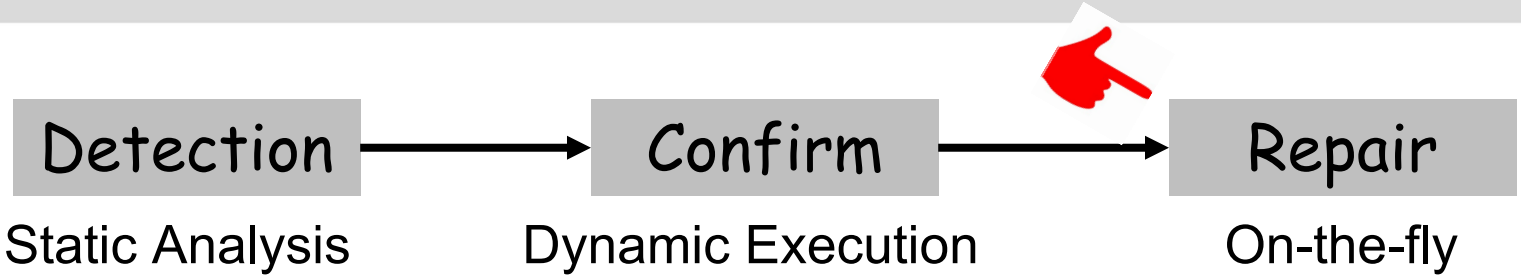
MONASH University



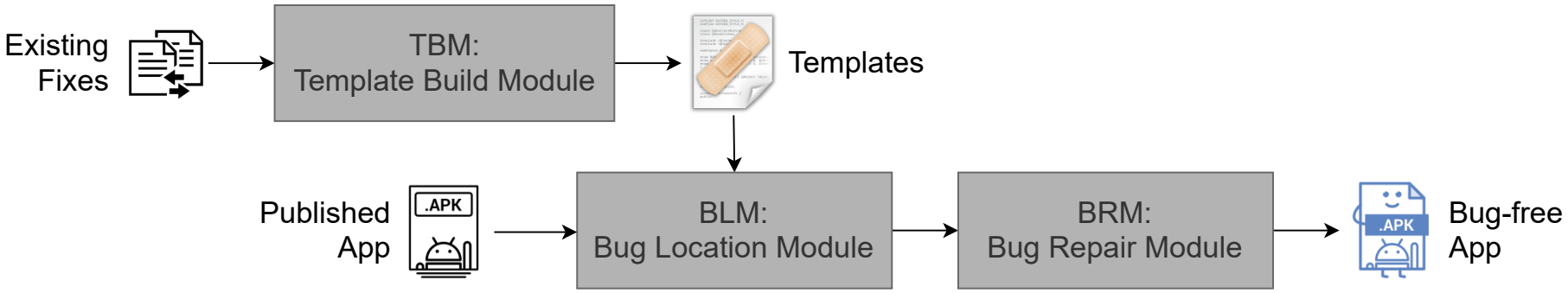


Compatibility Issues

MONASH University



RepairDroid: Automatically repair compatibility issues for published Android apps



Compatibility Issues

MONASH University


```
@@ Variable Declaration
$v0 := boolean
$v1 := ANY
[SEARCH] $v2 := <TYPE>
```

```
@@ Issue Location
[<ISSUE_TYPE>] <CONDITION>
```

```
@@ Patch Denotation
+ //Replacement Statements
- //Original Statements

OR

+ $v0 = <CONDITION EXPRESSION>
+ if $v0 == true
+   //Replacement Statements
+ else
    //Original Statements
```

Supporting Direct Jimple Statements to describe the patch

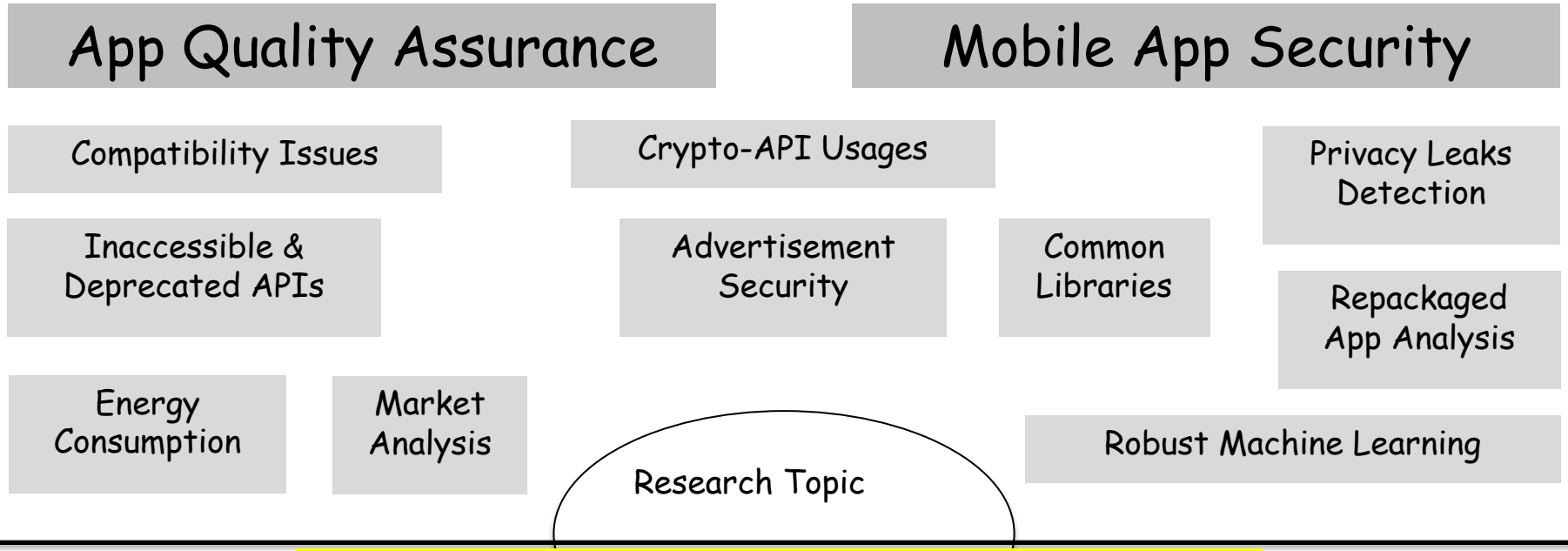
Jimple is the default Intermediate Representation of Soot, a popular Javastatic analyzer framework

→ Flexibility

Conclusion

Towards Engineering High-quality and Secure Mobile Apps for Social Good

MONASH University



Li Li,
ARC DECRA Fellow
FIT, Monash University

Li.Li@Monash.edu
http://lilicoding.github.io