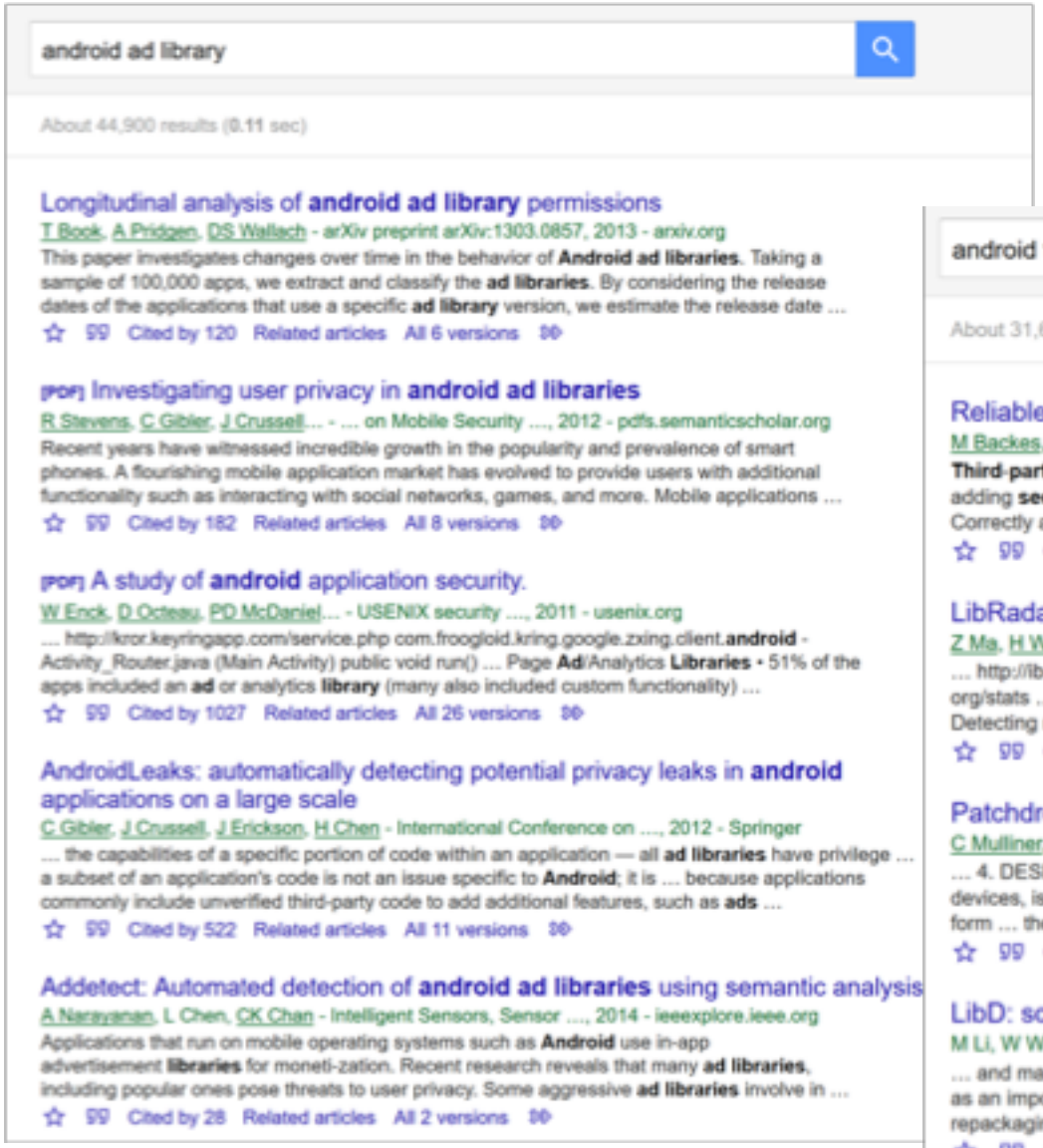
FLEXDROID: Enforcing In-App Privilege Separation in Android

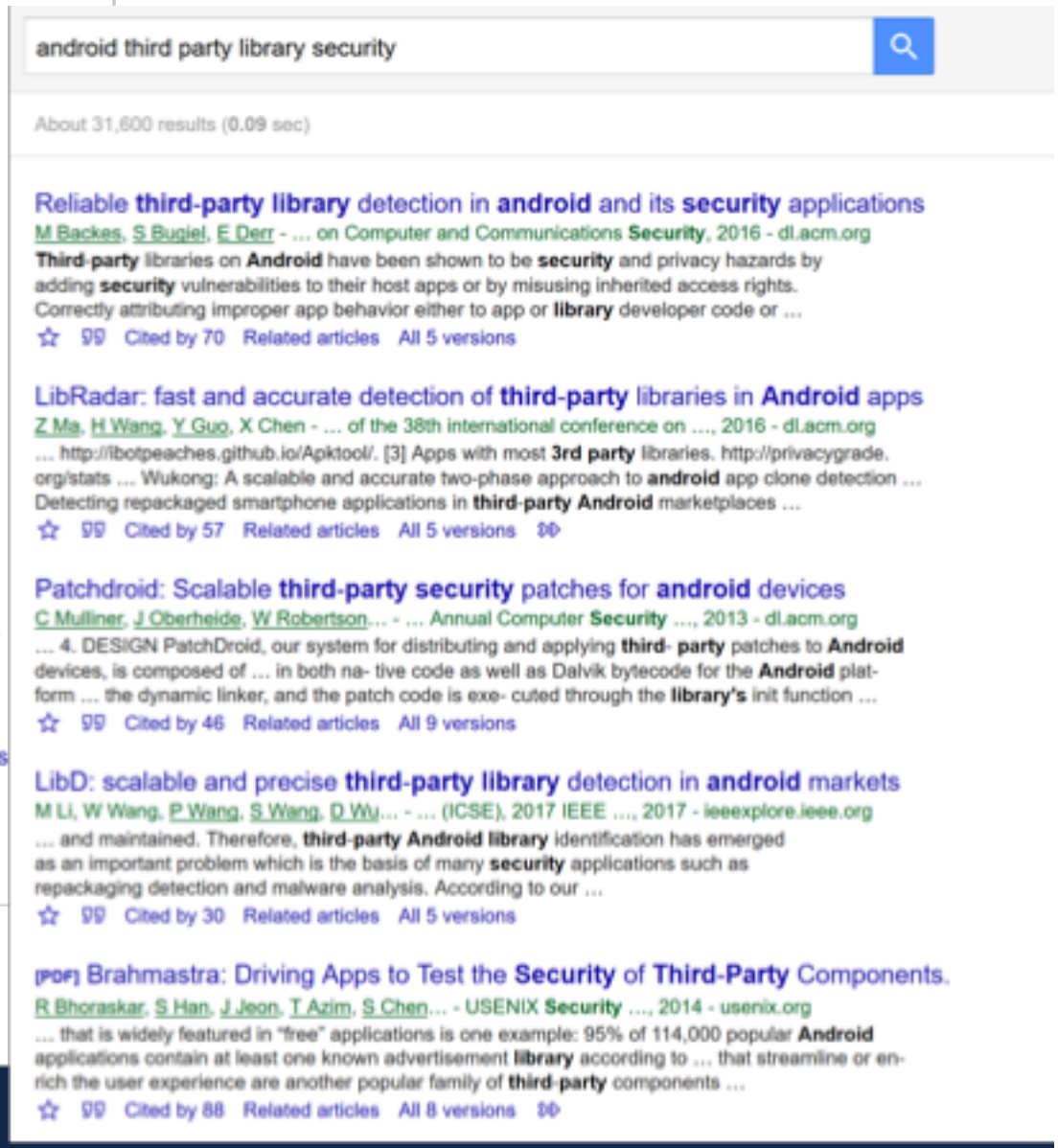
Jaebaek Seo, Daehyeok Kim, Donghyun Cho, Taesoo Kim, Insik Shin
(NDSS ‘16)

Presented by Shivansh Chandnani
CS 563 (Fall 2018)

3rd party libraries are very popular in Android

Can we trust these third party libraries?


android ad library
About 44,900 results (0.11 sec)
Longitudinal analysis of android ad library permissions
T Book, A Pridgen, DS Wallach - arXiv preprint arXiv:1303.0857, 2013 - arxiv.org
This paper investigates changes over time in the behavior of Android ad libraries. Taking a sample of 100,000 apps, we extract and classify the ad libraries. By considering the release dates of the applications that use a specific ad library version, we estimate the release date ...
Cited by 120 Related articles All 6 versions
[PDF] Investigating user privacy in android ad libraries
R Stevens, C Gibler, J Crussell... - ... on Mobile Security ..., 2012 - pdfs.semanticscholar.org
Recent years have witnessed incredible growth in the popularity and prevalence of smart phones. A flourishing mobile application market has evolved to provide users with additional functionality such as interacting with social networks, games, and more. Mobile applications ...
Cited by 182 Related articles All 8 versions
[PDF] A study of android application security.
W Enck, D Octeau, PD McDaniel... - USENIX security ..., 2011 - usenix.org
... http://kror.keyringapp.com/service.php com.froogloid.kring.google.zxing.client.android - Activity_Router.java (Main Activity) public void run() ... Page Ad/Analytics Libraries • 51% of the apps included an ad or analytics library (many also included custom functionality) ...
Cited by 1027 Related articles All 26 versions
AndroidLeaks: automatically detecting potential privacy leaks in android applications on a large scale
C Gibler, J Crussell, J Erickson, H Chen - International Conference on ..., 2012 - Springer
... the capabilities of a specific portion of code within an application — all ad libraries have privilege ... a subset of an application's code is not an issue specific to Android; it is ... because applications commonly include unverified third-party code to add additional features, such as ads ...
Cited by 522 Related articles All 11 versions
Addetect: Automated detection of android ad libraries using semantic analysis
A Narayanan, L Chen, CK Chan - Intelligent Sensors, Sensor ..., 2014 - ieeexplore.ieee.org
Applications that run on mobile operating systems such as Android use in-app advertisement libraries for moneti-zation. Recent research reveals that many ad libraries, including popular ones pose threats to user privacy. Some aggressive ad libraries involve in ...
Cited by 28 Related articles All 2 versions


android third party library security
About 31,600 results (0.09 sec)
Reliable third-party library detection in android and its security applications
M Backes, S Bugiel, E Derr - ... on Computer and Communications Security, 2016 - dl.acm.org
Third-party libraries on Android have been shown to be security and privacy hazards by adding security vulnerabilities to their host apps or by misusing inherited access rights. Correctly attributing improper app behavior either to app or library developer code or ...
Cited by 70 Related articles All 5 versions
LibRadar: fast and accurate detection of third-party libraries in Android apps
Z Ma, H Wang, Y Guo, X Chen - ... of the 38th international conference on ..., 2016 - dl.acm.org
... http://ibotpeaches.github.io/Apktool/. [3] Apps with most 3rd party libraries. http://privacygrade.org/stats ... Wukong: A scalable and accurate two-phase approach to android app clone detection ... Detecting repackaged smartphone applications in third-party Android marketplaces ...
Cited by 57 Related articles All 5 versions
Patchdroid: Scalable third-party security patches for android devices
C Mulliner, J Oberheide, W Robertson... - ... Annual Computer Security ..., 2013 - dl.acm.org
... 4. DESIGN PatchDroid, our system for distributing and applying third- party patches to Android devices, is composed of ... in both na- tive code as well as Dalvik bytecode for the Android plat- form ... the dynamic linker, and the patch code is exe- cuted through the library's init function ...
Cited by 46 Related articles All 9 versions
LibD: scalable and precise third-party library detection in android markets
M Li, W Wang, P Wang, S Wang, D Wu... - ... (ICSE), 2017 IEEE ..., 2017 - ieeexplore.ieee.org
... and maintained. Therefore, third-party Android library identification has emerged as an important problem which is the basis of many security applications such as repackaging detection and malware analysis. According to our ...
Cited by 30 Related articles All 5 versions
[PDF] Brahmastra: Driving Apps to Test the Security of Third-Party Components.
R Bhoraskar, S Han, J Jeon, T Azim, S Chen... - USENIX Security ..., 2014 - usenix.org
... that is widely featured in "free" applications is one example: 95% of 114,000 popular Android applications contain at least one known advertisement library according to ... that streamline or en- rich the user experience are another popular family of third-party components ...
Cited by 88 Related articles All 8 versions

Fundamental Problem

Third party libraries in Android have the same access to permissions as the host app

How can this lead to problems?

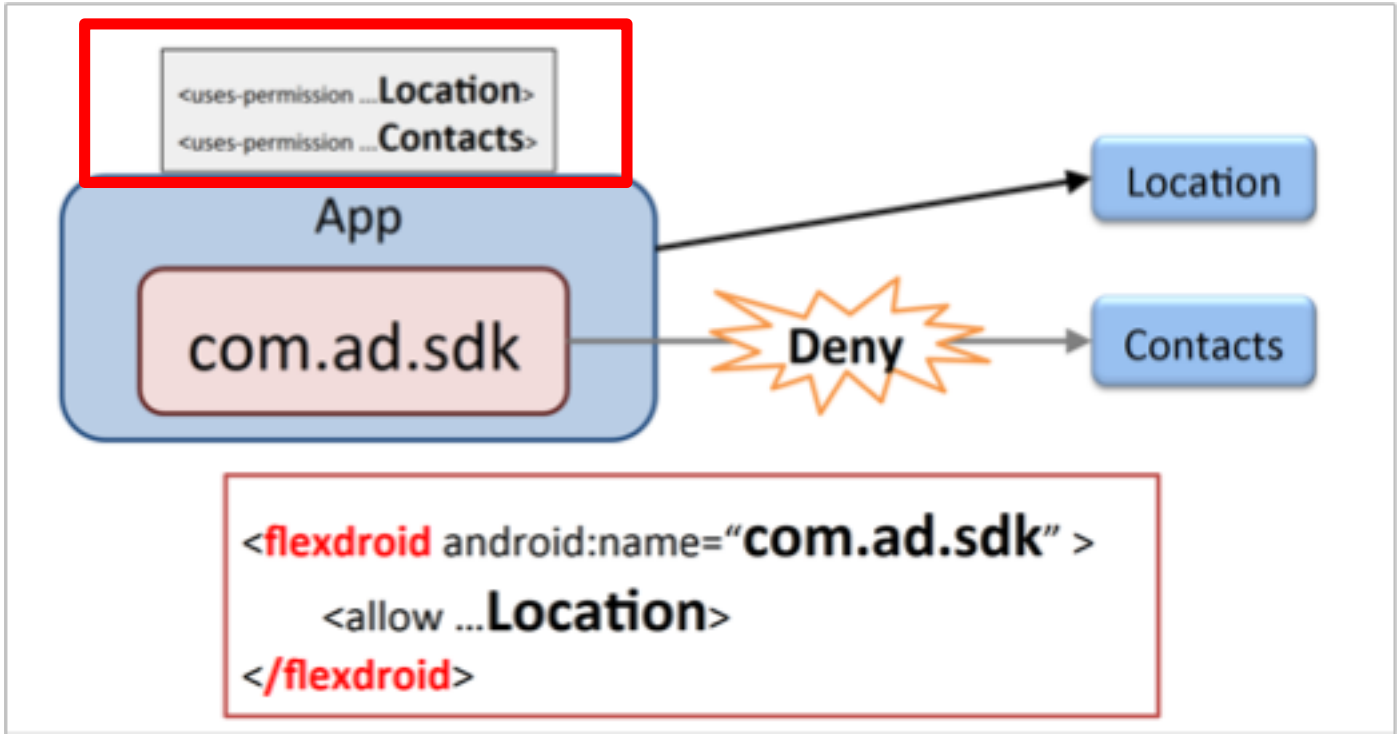
Results from analysis of 100,000 apps:

Name	Category	Accounts	Phone information	Internet	Read SMS	Write SMS	Read Calendar	Write Calendar	Write Settings	Get Tasks	Read Bookmark	Record Audio	Location
Facebook	Social	·	×	O	·	·	·	·	×	·	·	·	×
Flurry	Analytics	·	×	O	·	·	·	·	·	·	·	·	O
RevMob	Advertising	×	△	O	·	·	·	·	·	·	·	·	·
Chartboost	Advertising	·	×	O	·	·	·	·	·	·	·	·	·
InMobi	Advertising	·	·	O	×	×	△	△	·	·	·	·	△
Millennialmedia	Advertising	·	·	O	·	·	·	·	·	·	·	O	×
Paypal	Billing	·	×	O	·	·	·	·	·	·	·	·	×
Umeng	Analytics	·	O	O	·	·	·	·	×	×	·	·	×
AppLovin	Advertising	△	O	O	·	·	·	·	·	·	·	·	·
Pushwoosh	Notification	·	O	O	·	·	·	·	·	·	·	·	×
Tapjoy	Advertising	·	O	O	·	·	×	×	·	·	·	·	×
AppFlood	Advertising	·	△	O	·	·	·	·	·	·	·	·	△
OpenFeint	Social	O	O	O	·	·	·	·	·	·	·	·	×
Airpush	Advertising	×	△	O	·	·	·	·	·	·	×	·	×
Youmi	Advertising	·	O	O	·	·	·	·	·	×	·	·	×
Cauly	Advertising	·	·	O	·	·	·	·	·	×	·	·	△
Socialize	Social	·	△	O	·	·	·	·	·	·	·	·	△
Domob	Advertising	·	O	O	·	·	·	·	·	·	·	·	×
Leadbolt	Advertising	×	△	O	·	·	×	×	·	·	·	·	△
MobFox	Advertising	·	×	O	·	·	·	·	·	·	·	·	△

Thoughts about threat model?

Solution

In-app privilege separation between a host application and it's third party libraries



Main challenges

- From the analysis of 295 libraries amongst the 100,000 apps
  - Class inheritance => 71.5%
  - Java Native Interface => 17.1%
  - Runtime class loading => 27.9%
  - Reflection => 49.6%

Name	Class Loading	Reflection	Callback	Class Inheritance	JNI
Facebook	✓	✓	✓	✓	✓
Flurry	✓	✓	·	✓	·
RevMob	·	·	✓	✓	✓
Chartboost	✓	✓	·	✓	✓
InMobi	✓	✓	✓	✓	·
Millennialmedia	✓	✓	·	✓	✓
Paypal	✓	·	·	✓	·
Umeng	✓	✓	·	✓	✓
AppLovin	✓	✓	✓	✓	·
Pushwoosh	·	✓	·	✓	·
Tapjoy	✓	✓	✓	✓	·
AppFlood	·	✓	·	·	·
OpenFeint	·	✓	✓	✓	·
Airpush	·	✓	·	✓	·
Youmi	·	✓	·	✓	·
Cauly	·	·	✓	✓	·
Socialize	✓	✓	·	✓	·
Domob	✓	✓	·	✓	·
Leadbolt	✓	✓	✓	✓	·
MobFox	✓	✓	·	✓	·

- From the 20 most popular third party libraries:
  - 19 use class inheritance
  - All use atleast one form of dynamic code exection

JNI

- Java Native Interface
- Allows developers to use libraries in native language
- Could improve an app’s performance
- Renders memory safety features of Java obsolete

Runtime class loading

```
public class MainClass {

  public static void main(String[] args){

    ClassLoader classLoader = MainClass.class.getClassLoader();

    try {
        Class aClass = classLoader.loadClass("com.jenkov.MyClass");
        System.out.println("aClass.getName() = " + aClass.getName());
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }

}
```

Source: http://tutorials.jenkov.com/java-reflection/dynamic-class-loading-reloading.html

Reflection

```
// Without reflection
Foo foo = new Foo();
foo.hello();

// With reflection
Object foo = Class.forName("complete.classpath.and.Foo").newInstance();

// Alternatively: Object foo = Foo.class.newInstance();
Method m = foo.getClass().getDeclaredMethod("hello", new Class<?>[0]);
m.invoke(foo);
```

Source: https://en.wikipedia.org/wiki/Reflection_(computer_programming)

Key Idea

Adjusting permissions dynamically whenever an app requests a resource

FLEXDROID Design

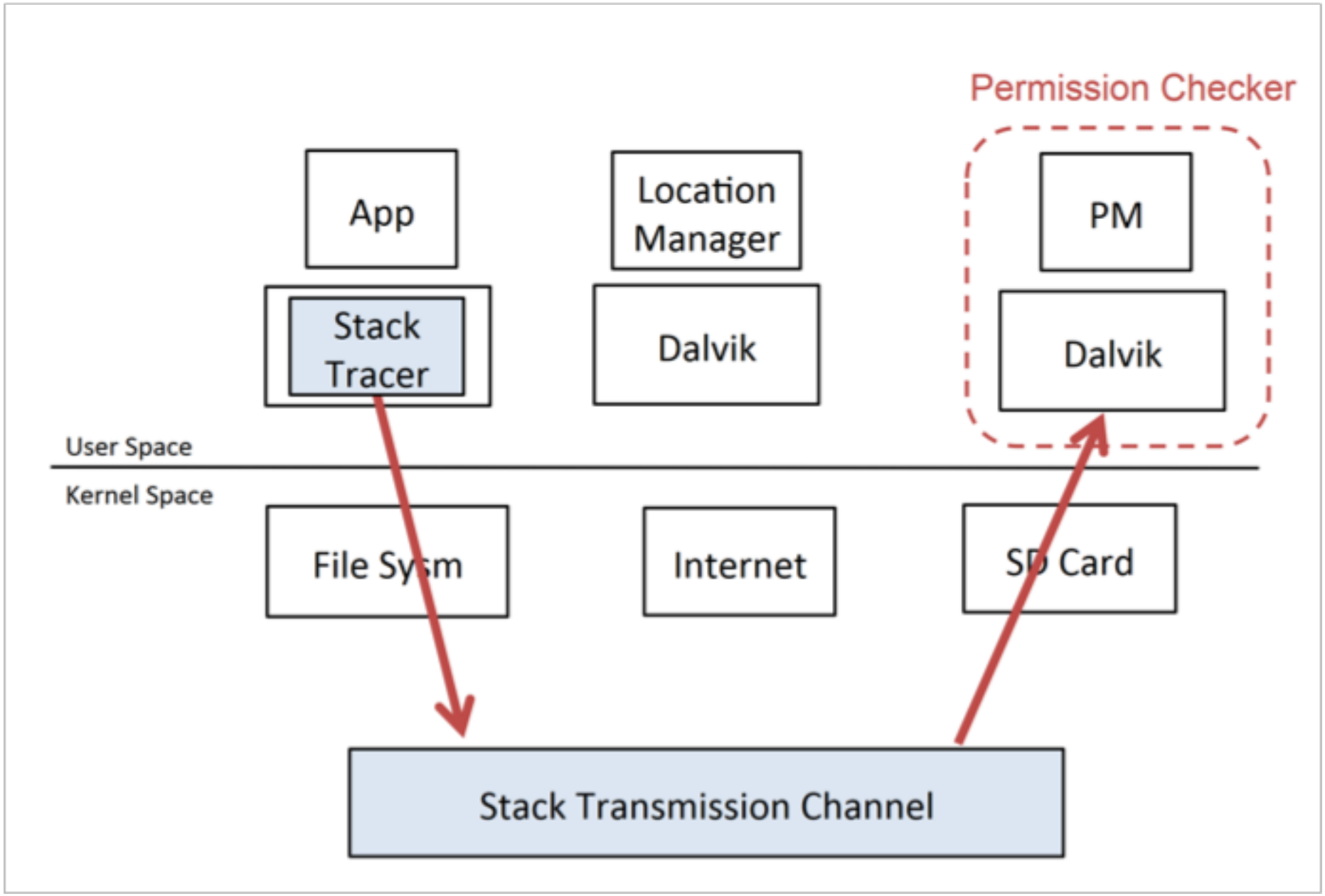
- Identify the principle using stack tracer
- Protect the integrity of the stack trace using tamper resistant memory protection mechanism
- Handle dynamic code execution
- Are there any alternate designs you think would be more reliable or easier to implement?

Stack tracer

- New special purpose thread for each process
- Uses secure transmission for data
- Amidst the initialization process of an app

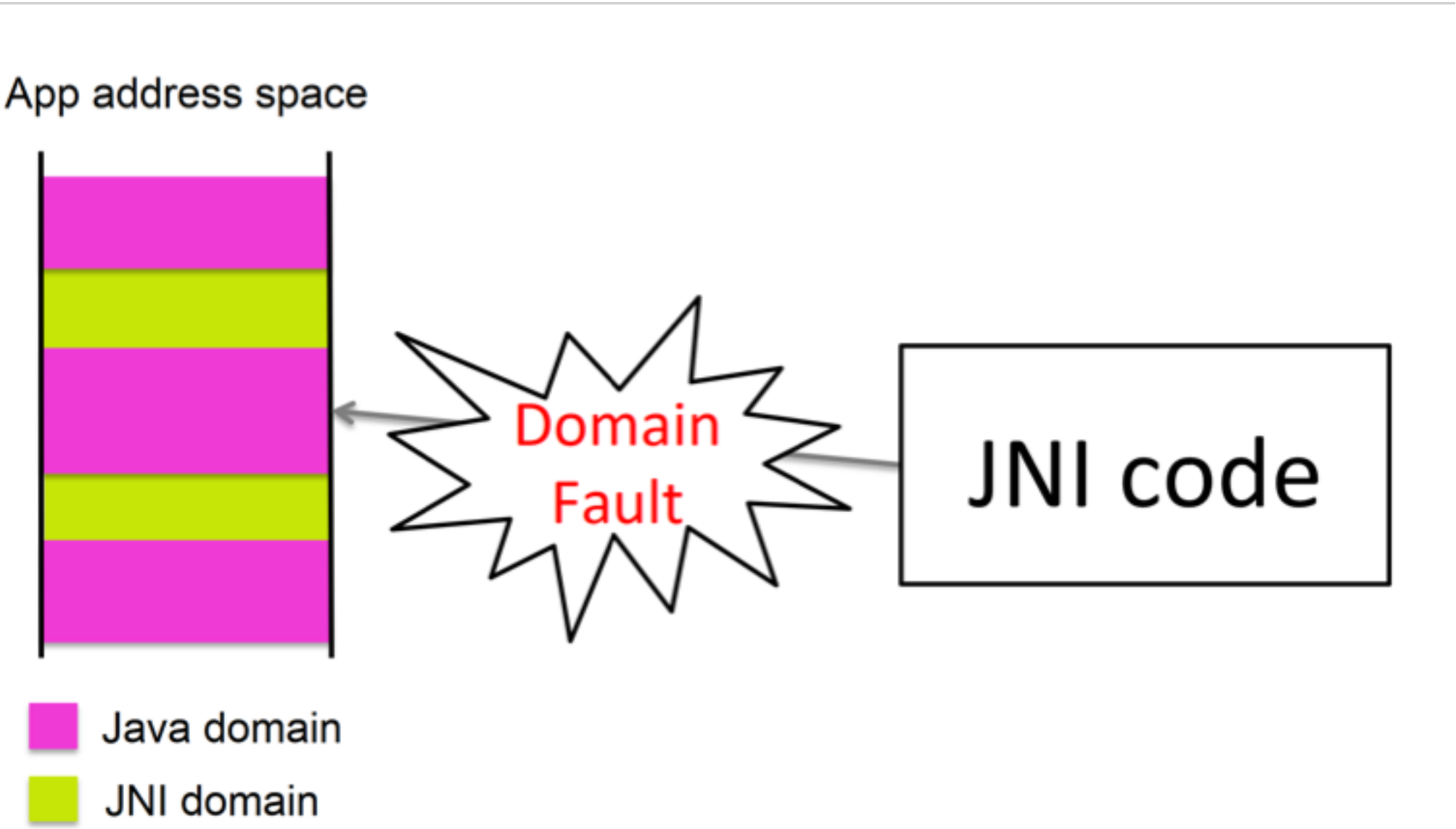
	P	Call stack
↓	A	com.A.functionA
	B	com.B.functionB
	C	com.C.functionC

Perm = Perm(A)
∩ Perm(B)
∩ Perm(C)


Permission Checker
App
Location Manager
PM
Stack Tracer
Dalvik
Dalvik
User Space
Kernel Space
File Sysm
Internet
SD Card
Stack Transmission Channel

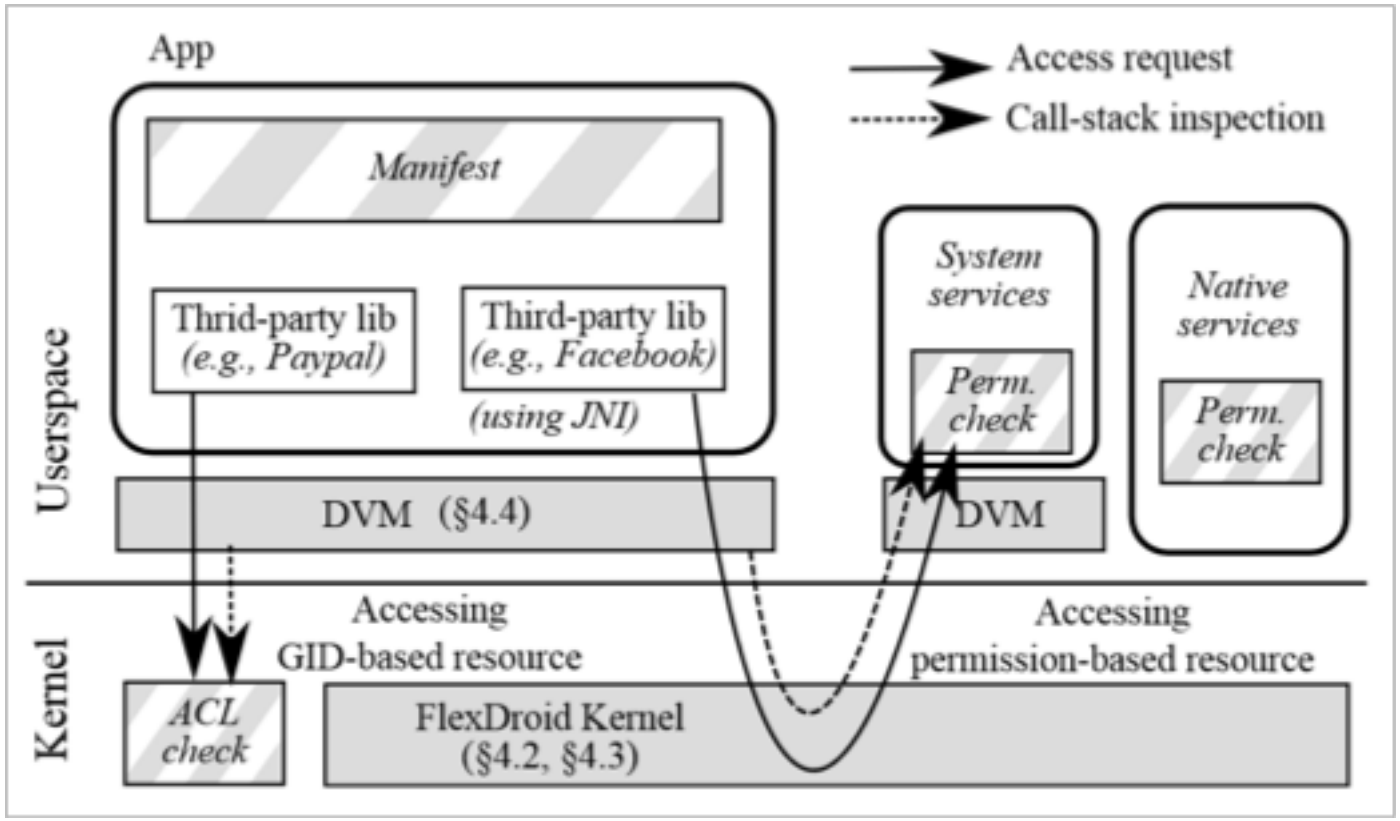
Memory isolation

- Inspired by ARMLock (CCS ‘14)
- Regard JNI code as potentially malicious code
  - Run it in a separate and restricted memory domain


App address space
Domain Fault
JNI code
Java domain
JNI domain

Protection against dynamic techniques

- Store the context of class loader
- Store the parent thread's permissions in case of threads
- Basic idea: Use dynamic permissions with context at runtime and creation


App
Access request
Call-stack inspection
Manifest
Thrid-party lib
(e.g., Paypal)
Third-party lib
(e.g., Facebook)
(using JNI)
System
services
Perm.
check
Native
services
Perm.
check
Userspace
DVM (§4.4)
DVM
Accessing
GID-based resource
Accessing
permission-based resource
Kernel
ACL
check
FlexDroid Kernel
(§4.2, §4.3)

Code modified

- Experiments performed on Android 4.4.4. Had 40% market share in 2015.
  - Dalvik replaced with Android Runtime (ART) in Android 5.0. Should the authors have used a different android version to test?

	# of Files	Insertion (LoC)	Deletion (LoC)
Kernel	28	1831	25
Android Framework	46	1466	77
Dalvik VM	24	6081	22
Bionic	23	2827	70
Others	12	95	24
Total	**133**	**12300**	**218**

Evaluation

- Evaluated 32 top apps across categories
- Ran for 10 minutes in both stock android and FLEXDROID.
- 5 apps crashed: Waze, Uber, Acrobat Reader, Facebook and UC Browser
- Is it necessary for a security modification to be backwards compatible?

Usability

Target Third-party Library	Role	App Name	Blocked Resource
com.google.ads.*†	Ad	ZingBox Manga	Internet
jp.naver.line.*‡	Photo	LINE Messenger	Camera
com.ebay.redlasersdk.*	Barcode scanner	eBay	Camera
com.facebook.*†	Login	Airbnb	Internet
com.tapjoy.*	Ad	Subway Surf	Internet
com.twitter.*†	Login	Drugs.com	Internet
com.android.volley.*	HTTP	Yahoo News	Internet
com.flurry.*†	Analytics	Yahoo Mail	Internet

† Used in two or more apps
‡ A component of the app, not a third-party library

- Recompiled apps with flexdroid tag to black third party library's access
- Is this convincing enough that FLEXDROID works as expected?
  - How about dynamic code execution?

Performance Overhead

- Seems to add very little overhead
- Any better way to measure performance overhead?

Use scenario	Android	FlexDroid	Over.
Launch an application*	39.13 ms	39.73 ms	1.55%
Launch a service	3.76 ms	3.95 ms	5.22%
Download 1.3MB file	136.54 ms	139.59 ms	2.24%
Take a photo	443.01 ms	448.99 ms	1.35%
Send an email*	100.56 ms	101.70 ms	1.13%
Read 8.4MB file via JNI	88.71 ms	89.16 ms	0.51%

∗ Functionalities of open-source K-9 email app

Micro-benchmarks

Benchmark	Android	FlexDroid	Over.
`startActivity()`	3,935 μs	4,529 μs	594 μs
`startService()`	1,221 μs	1,734 μs	513 μs
file open*	782 μs	1,657 μs	875 μs
file open (create)*	1,390 μs	2,338 μs	948 μs
file delete	745 μs	1,330 μs	585 μs
file read†	138 μs	142 μs	4 μs
file write†	1,076 μs	1,134 μs	58 μs
call JNI method	97 μs	186 μs	89 μs
call JNI method after loading libs‡	963 μs	8,436 μs	7,473 μs

∗ Two stack inspections are required during a file open
† No stack inspection is required during file read and write
‡ This includes the process of loading (and dynamic linking) the JNI code and shared libraries needed by the JNI code

- File open and delete have performance overheads as high as 100%.
- JNI methods have very high overhead.
- Does this mean the benchmarks with K-9 email app were biased?

Key Takeaways

- Android permission system has a fundamental problem with 3rd party libraries
- Third party libraries are using more data than they inform the developer about
- FLEXDroid allows to separate the app's trust from its libraries

Discussion

- How does this change with runtime permissions?
- Do the sweeping changes required in popular apps disincentivize google to adopt these changes?
- Better to provide fake data or no data?
- Thoughts on how their performance evaluation could be more convincing?