

MALintent

Coverage Guided Intent Fuzzing
Framework for Android

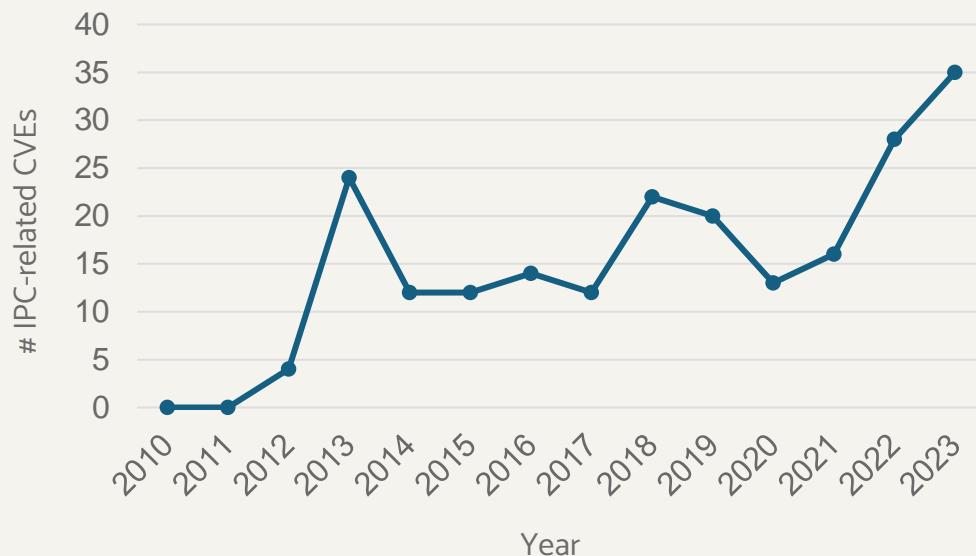
Ammar Askar, Fabian Fleischer,
Christopher Kruegel, Giovanni Vigna,
Taesoo Kim



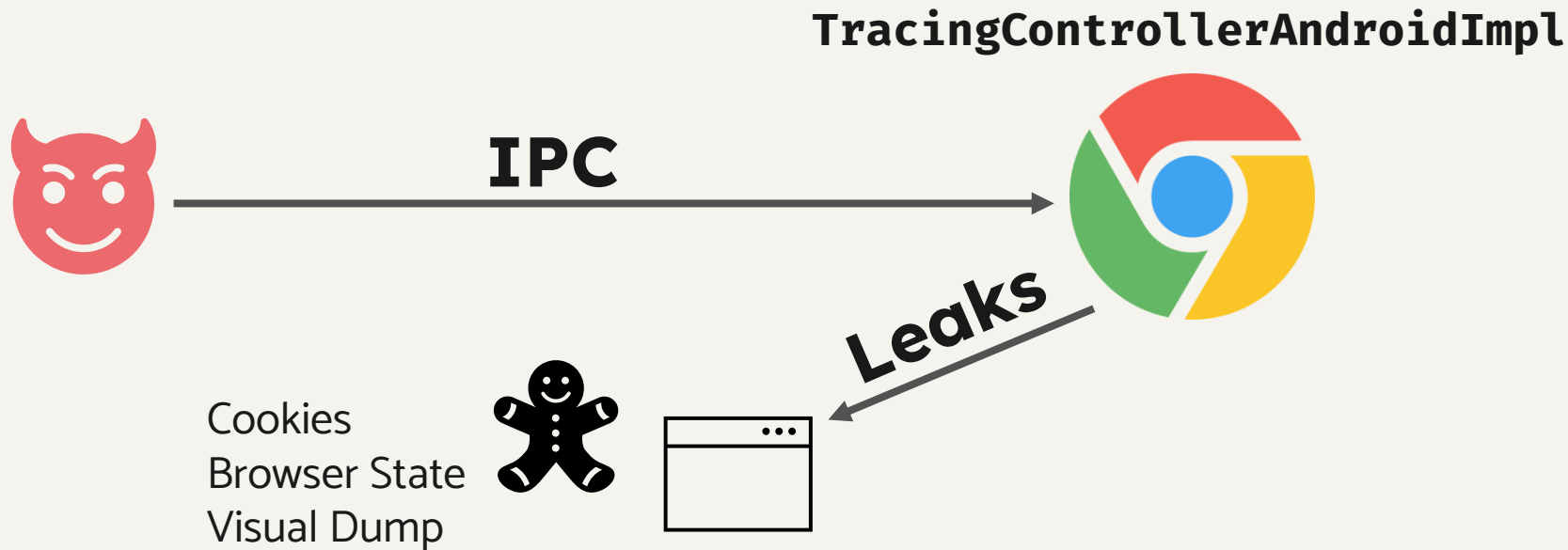
Android IPC-related CVEs

App isolation turns IPC into a key attack vector on Android

Increasing number of IPC-related CVEs in Android apps



Motivation



Android Intents

Primary Inter Process Communication method in Android.



Android Intents

An object sent across app boundaries. Android apps are isolated and cannot directly access each other's data or special permissions.

Intent

Actions: ACTION_VIEW (view an image)
ACTION_DIAL (dial a number)
ACTION_SEND (send an email)

Metadata

EXTRA_EMAIL (email address to send to)

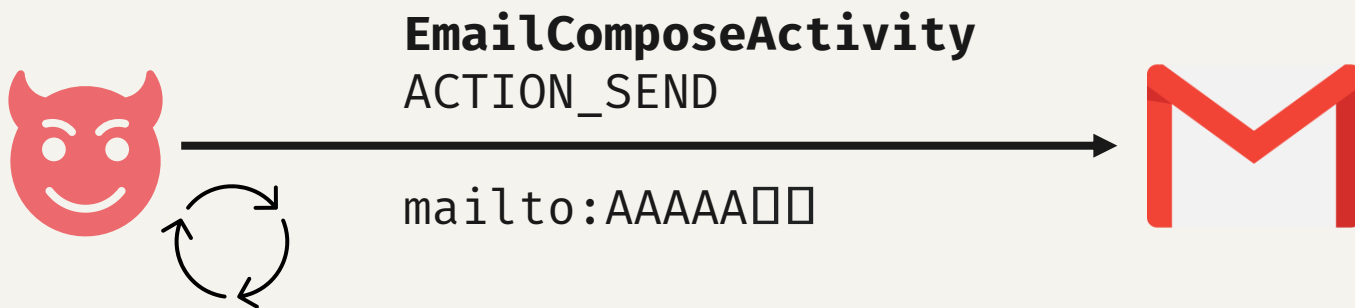
Android Intents

Apps can trigger intents to launch other applications.

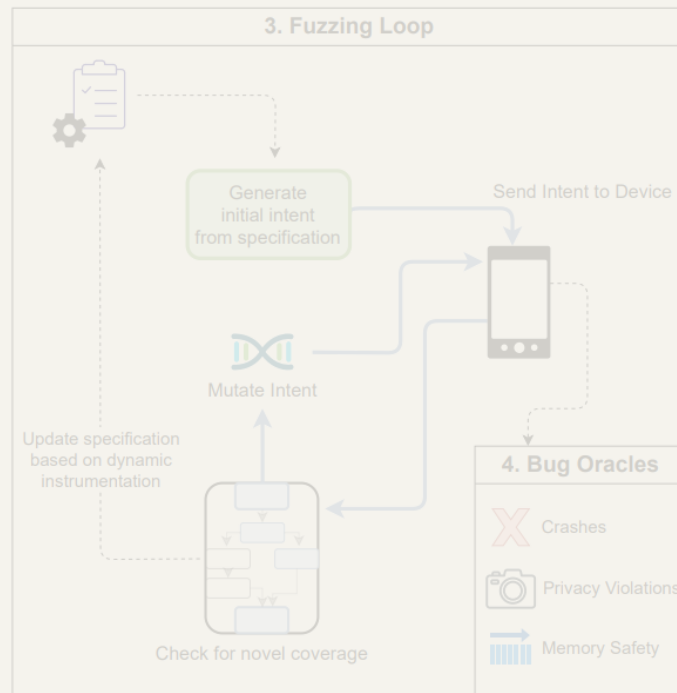
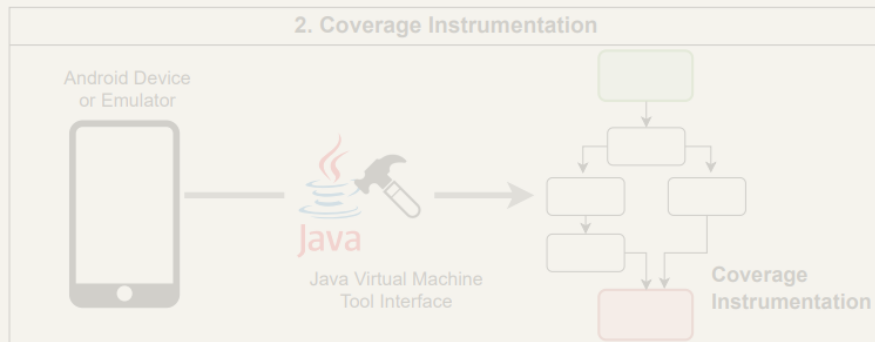
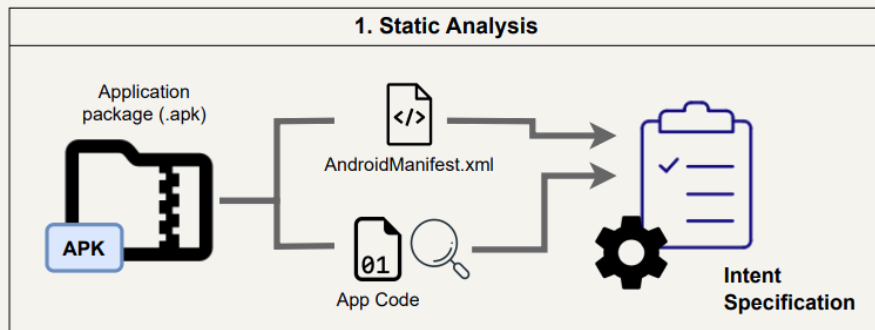
```
private void contactSupportEmail() {  
    Intent i = new Intent(Intent.ACTION_SEND);  
    i.setData(Uri.parse("mailto:"));  
    i.putExtra(Intent.EXTRA_EMAIL, "support@game.com");  
    i.putExtra(Intent.EXTRA_SUBJECT, "Support Ticket XYZ");  
  
    File bugReport = generateBugReportLog();  
    i.putExtra(Intent.EXTRA_STREAM, Uri.fromFile(bugReport));  
  
    // Start the user's preferred email app to send email.  
    startActivity(i);  
}
```

Android Intents

Intents are ideal for fuzzing. They contained well-structured data and can be sent by apps without any privilege to attack other apps.

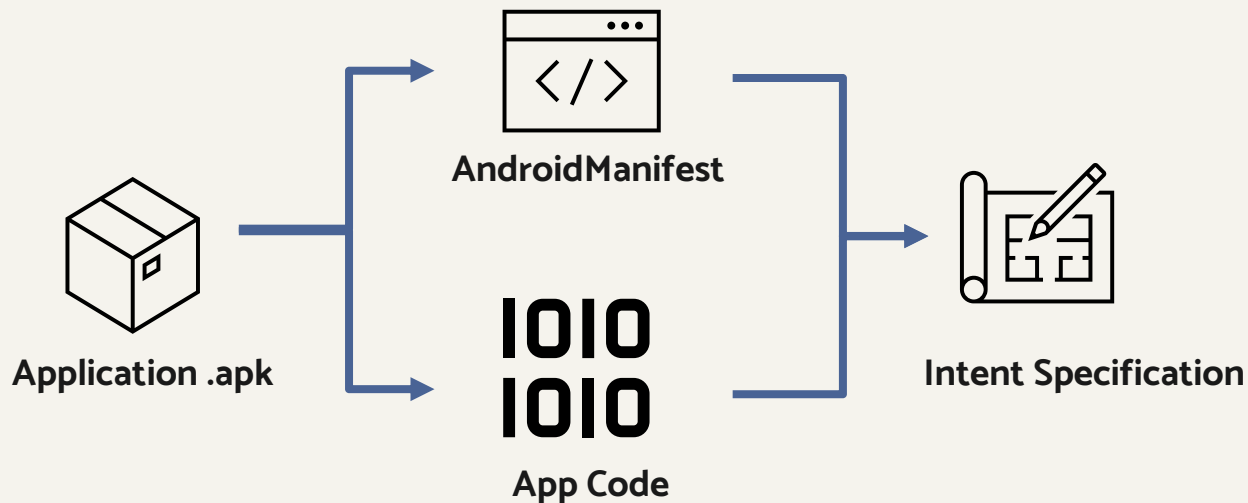


Overall Design

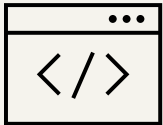


Design: Static Analysis

Start with the application to determine what intents can be sent to it.



Design: Static Analysis



AndroidManifest

```
<activity android:exported="true"
          android:name=".EmailComposeActivity">
  <intent-filter>
    <action android:name="android.intent.action.SEND" />
    <data android:scheme="mailto" />
  </intent-filter>
</activity>
```

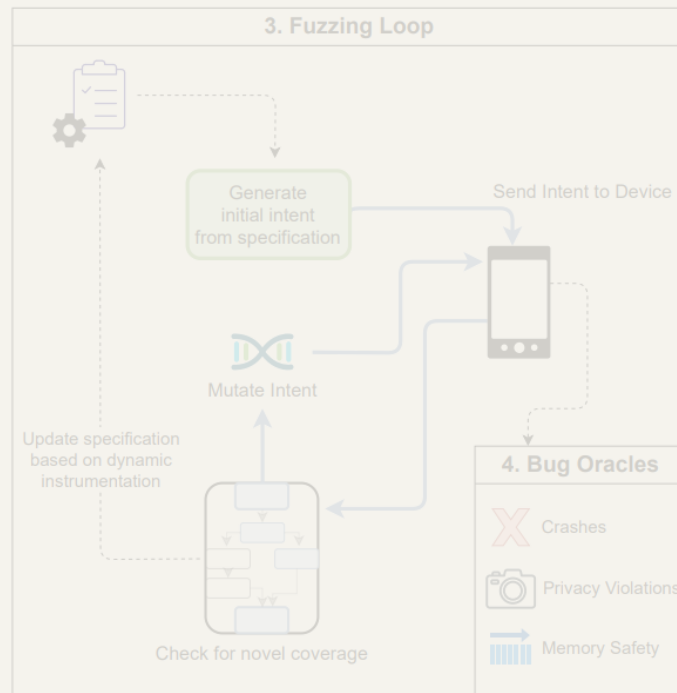
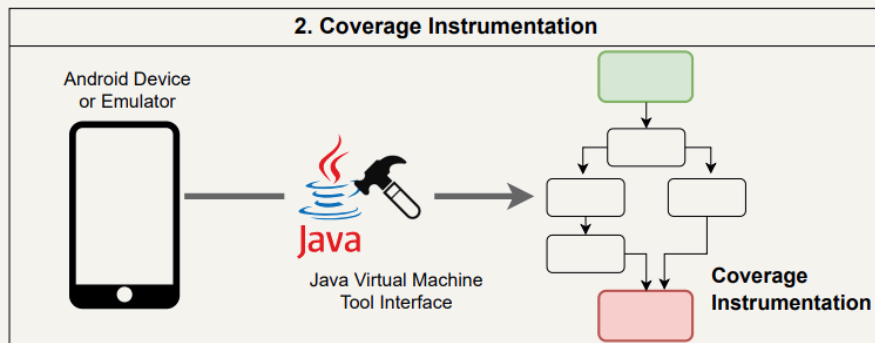
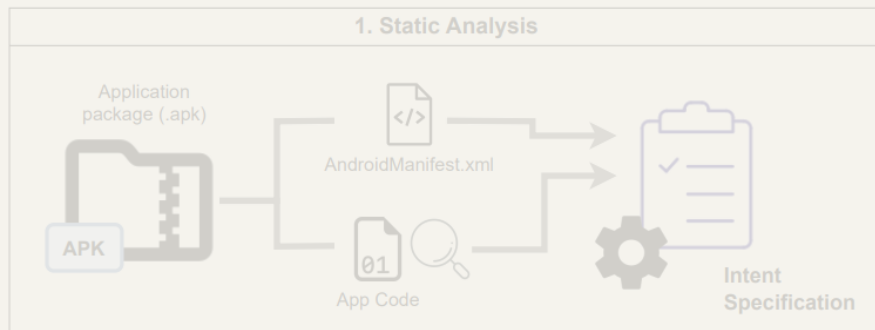
Design: Static Analysis



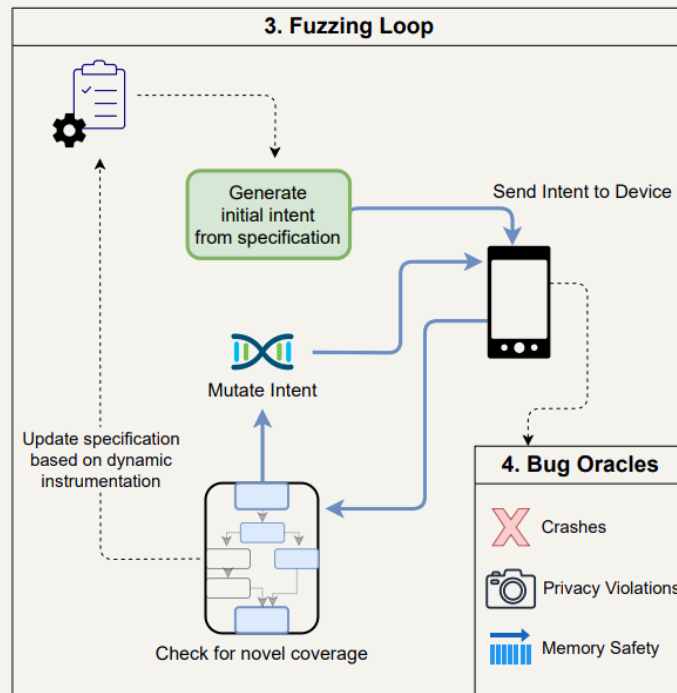
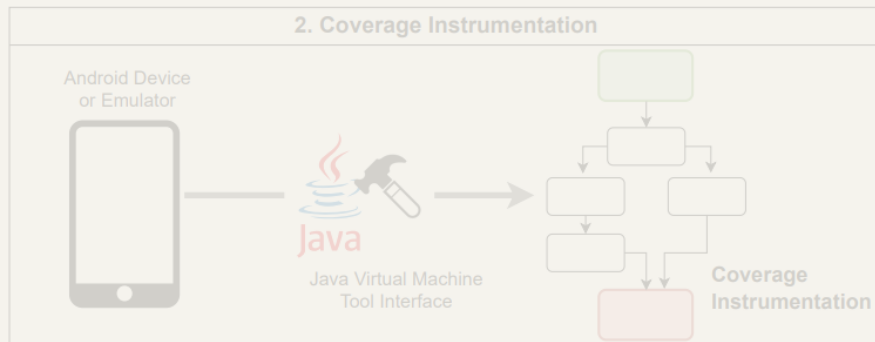
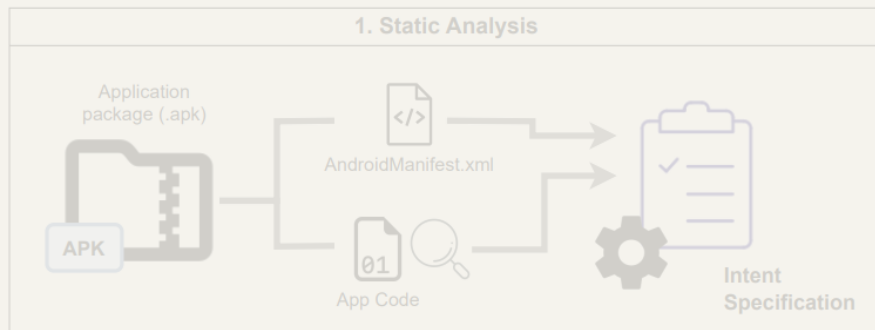
Intent Specification

```
{
  "package": "com.sec.android.app.sbrowser",
  "name":
    "com.sec.android.app.sbrowser.SBrowserLauncherActivity",
  "component": "<activity>",
  "action": "android.intent.action.VIEW",
  "categories": [
    "android.intent.category.DEFAULT",
    "android.intent.category.BROWSABLE",
  ],
  "data": {
    "scheme": ["http", "https", "about", "javascript"],
  },
  "extras": {
    "android.intent.extra.REFERRER_NAME": "string",
    "create_new_tab": "boolean",
    "trusted_application_code_extra": "string",
    "com.android.browser.headers": "bundle",
    "// ..."
    "// More extras omitted for space."
  },
}
```

Overall Design



Overall Design



Android Native Code

There are a wealth of libraries in C/C++ that app developers use



Android Native Code

Interfacing with these libraries done with JNI (Java Native Interface)

```
public class HelloWorldJNI {  
  
    private native void sayHello();  
  
}
```

Java

```
JNIEXPORT void JNICALL Java_HelloWorldJNI_sayHello  
    (JNIEnv* env, jobject thisObject) {  
  
    std::cout << "Hello from C++ !!" << std::endl;  
  
}
```

C

JNI Bug Finding

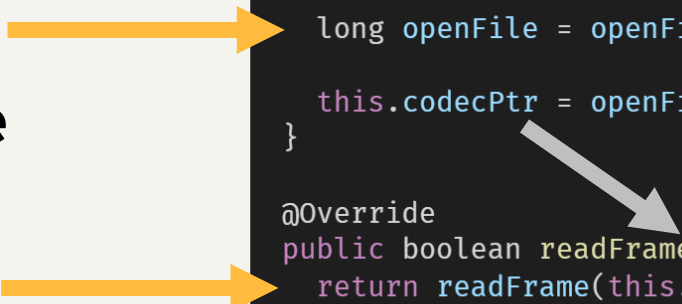


Basic fuzzing loop works but it is slow for JNI bugs.



JNI Bug Finding

Native Calls



```
@Override
public boolean openFile() {
    String file = this.filename;
    Rect rect = this.cropRect;

    long openFile = openFile(file, rect.left, rect.right,
                             rect.top, rect.bottom, this.metaData);
    this.codecPtr = openFile;
}

@Override
public boolean readFrame() {
    return readFrame(this.codecPtr);
}
```


JNI Bug Finding

Dynamic traces from real intents invoking native code
give us data-flow information



JNI Bug Finding

With the data and control flow information, we can generate a libFuzzer harness



```
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *Data, size_t Size) {  
    // File contents fuzzed.  
    // Other variables like cropRect uses constants from dynamic traces.  
    void* codecPtr = Java_com_mobigames_openFile(Data, 0, 0, 100, 200, nullptr);  
    Java_com_mobigames_readFrame(codecPtr);  
}
```

JNI Bug Finding

Example bug found in the Facebook Fresco image library.

Native code: Fixed rounded corners off by one calculations

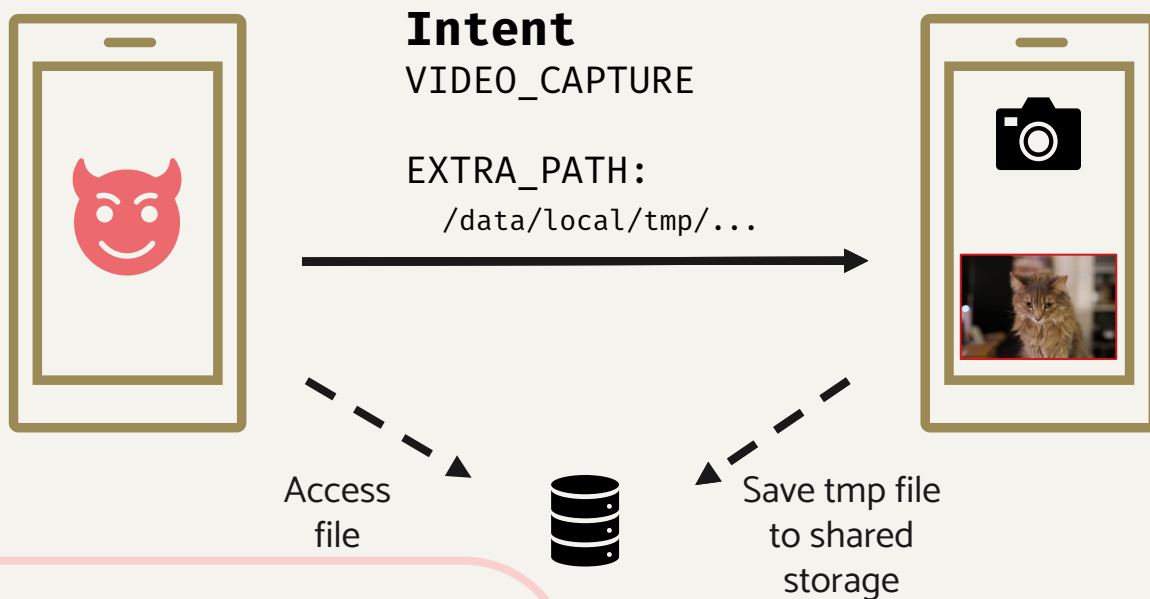
1 file changed +4 -4 lines changed

native-filters/src/main/jni/filters/rounding_filter.c

@@ -402,16 +402,16 @@ static void addRoundCorner(
402 centerY = radius;	402 centerY = radius;
403 break;	403 break;
404 case TOP_RIGHT:	404 case TOP_RIGHT:
405 - centerX = w - radius;	405 + centerX = w - radius - 1;
406 centerY = radius;	406 centerY = radius;
407 break;	407 break;
408 case BOTTOM_RIGHT:	408 case BOTTOM_RIGHT:
409 - centerX = w - radius;	409 + centerX = w - radius - 1;
410 - centerY = h - radius;	410 + centerY = h - radius - 1;
411 break;	411 break;
412 case BOTTOM_LEFT:	412 case BOTTOM_LEFT:
413 centerX = radius;	413 centerX = radius;
414 - centerY = h - radius;	414 + centerY = h - radius - 1;
415 }	415 }
416	416
417 if (radius < 1) {	417 if (radius < 1) {

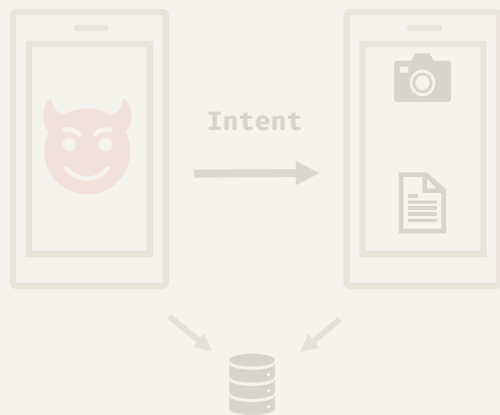
Privacy Violations

Improper intent handling may introduce vulnerabilities

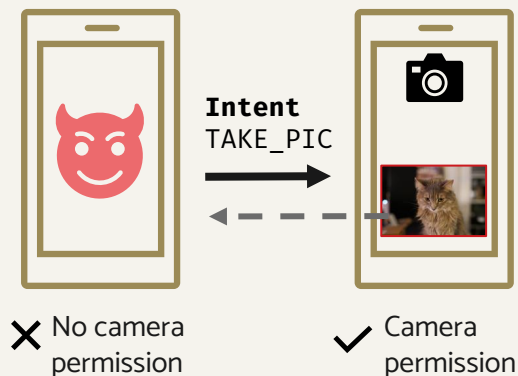


Privacy Violations: Attack Scenarios

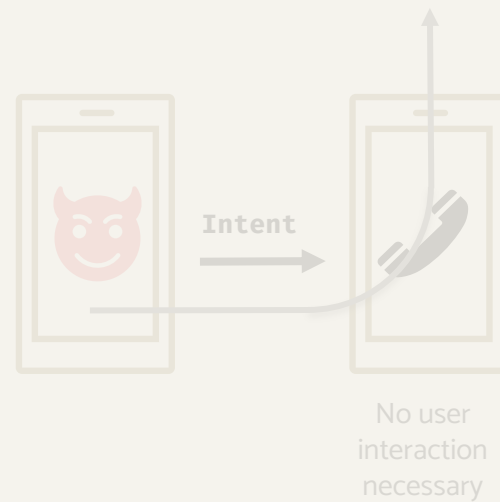
Data leak through
file system



Permission
escalation



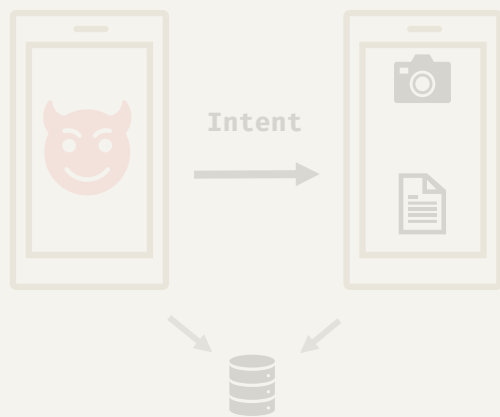
Call without user
interaction



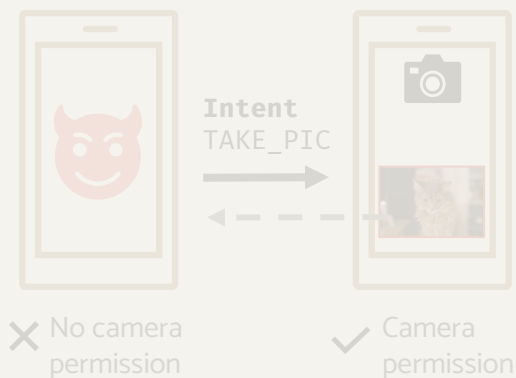
➔ Data flow analysis to detect privacy violations

Privacy Violations: Attack Scenarios

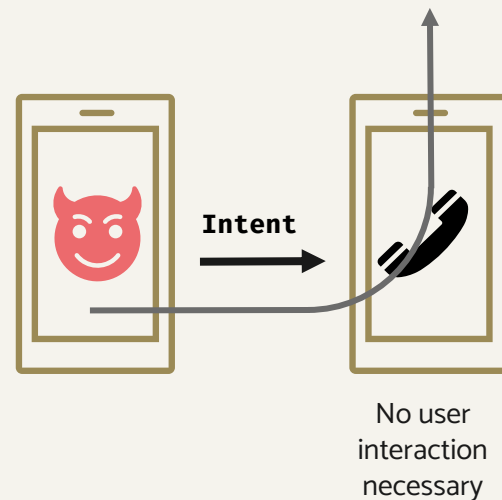
Data leak through
file system



Permission
escalation

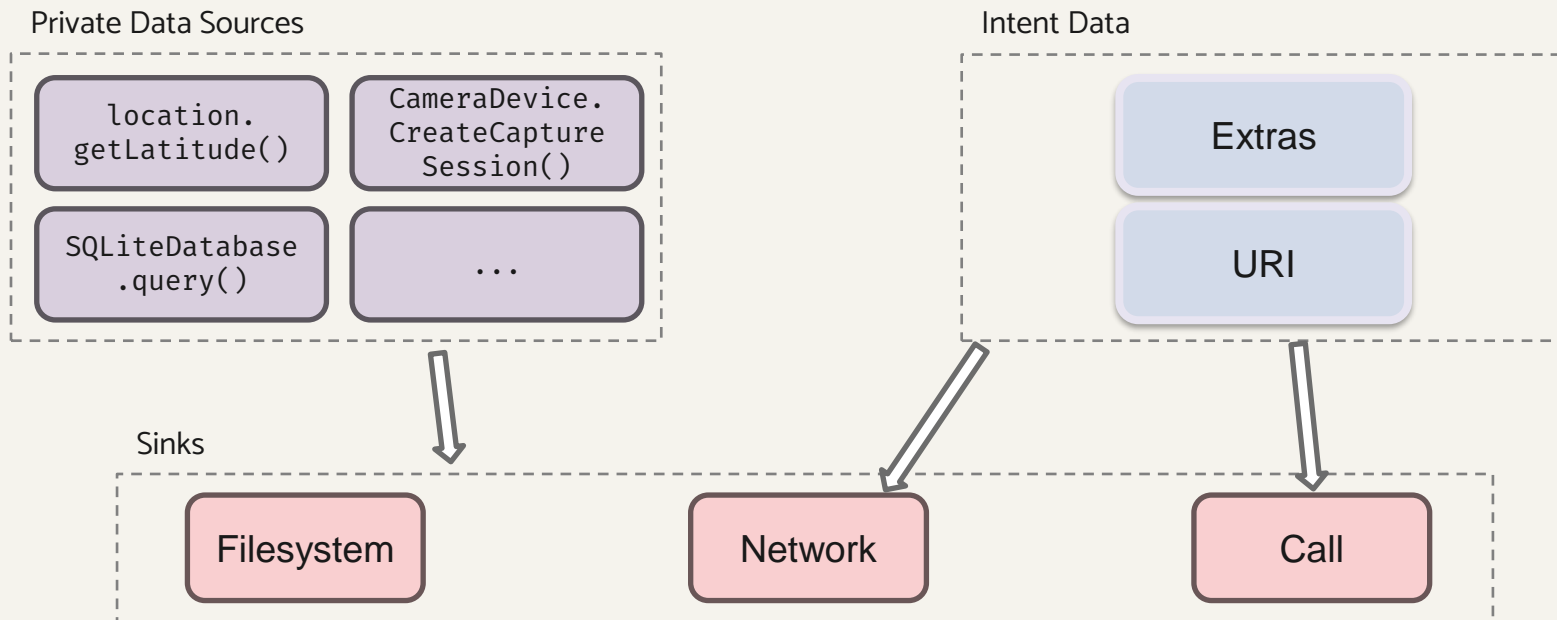


Call without user
interaction



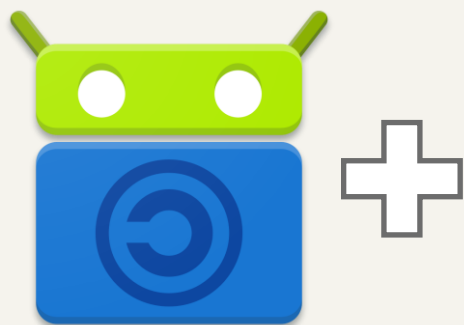
Privacy Violations

Dynamic taint analysis to identify leaking resources



Evaluation

Ran against 500 F-Droid and Google Play Store's top-50 and top-50 productivity apps.



F-Droid



**Google Play
Store**



**16 hours of
Fuzzing / App**

Evaluation Results

9



Privacy
Violations



49



Crashes

1



Memory
Safety

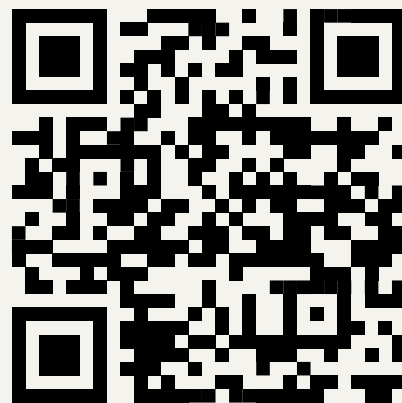


MALintent: Coverage Guided Intent Fuzzing Framework for Android

- Framework for fuzzing Intent handlers in Android apps
- Includes oracles for bug detection
 - Privacy violations, memory safety, crashes
- Found 49 crashes, 9 privacy violations, and 1 memory safety bug
- Open-source implementation available

Ammar Askar, Fabian Fleischer, Christopher Kruegel, Giovanni Vigna, Taesoo Kim
Network and Distributed System Security (NDSS) Symposium 2025. San Diego, CA.

aaskar@gatech.edu, fleischer@gatech.edu



Paper

Design: Static Analysis

IOIO
IOIO

App Code

```
private void handleIntent(Intent i) {  
    // ...  
    if (i.hasExtra(Intent.EXTRA_EMAIL)) {  
        setRecipient(i.getStringExtra(Intent.EXTRA_EMAIL));  
    }  
    if (i.hasExtra(Intent.EXTRA_SUBJECT)) {  
        setSubject(i.getStringExtra(Intent.EXTRA_SUBJECT));  
    }  
    if (i.hasExtra(Intent.EXTRA_STREAM)) {  
        // ...  
    }  
}
```

Android Native Code

Since it's native code, it's subject to memory-safety issues.

```
JNIEXPORT void JNICALL Java_HelloWorldJNI_sayHello(
    JNIEnv* env, jobject thisObject) {

    jclass jCls = (*env)→GetObjectClass(env, object);
    jfieldId fieldId = (*env)→GetFieldId(env, jCls, "doubleArr", "[D");
    jdoubleArray* array = (*env)→GetObjectField(env, object, fieldId);

    char[512] buf;
    memcpy(env→GetDoubleArrayElements(buf,
                                        env→GetDoubleArrayElements(array),
                                        env→GetArrayLength(array) * sizeof(double));

}
```

Android Native Code

```
// ...
if (i.hasExtra(Intent.EXTRA_STREAM)) {
    Uri uri = i.getParcelableExtra(Intent.EXTRA_STREAM);
    File attachment = new File(uri.getPath());

    renderPreviewIfImage(attachment);
}
// ...

private void renderPreviewIfImage(File f) {
    // ...
    // Use GIFLib through the JNI.
    GifInfoHandle handle = new GifInfoHandle(f);
    byte[] pixels = new byte[header_height * header_width];
    handle.renderFrame(pixels);
}
```

and... Intent handlers can
and do invoke native code

Android Native Code

JNI allows access to Java data from C/C++ land

```
JNIEXPORT void JNICALL Java_HelloWorldJNI_sayHello(
    JNIEnv* env, jobject thisObject) {

    jclass jCls = (*env)→GetObjectClass(env, object);
    jfieldID fieldId = (*env)→GetFieldID(env, jCls, "foo", "D");
    jdouble dblVar = (*env)→GetDoubleField(env, object, fieldId);

    std::cout << "Value of this.foo is " << dblVar;
}
```

```
public class HelloWorldJNI {
    private native void sayHello();
    private double foo;
}
```

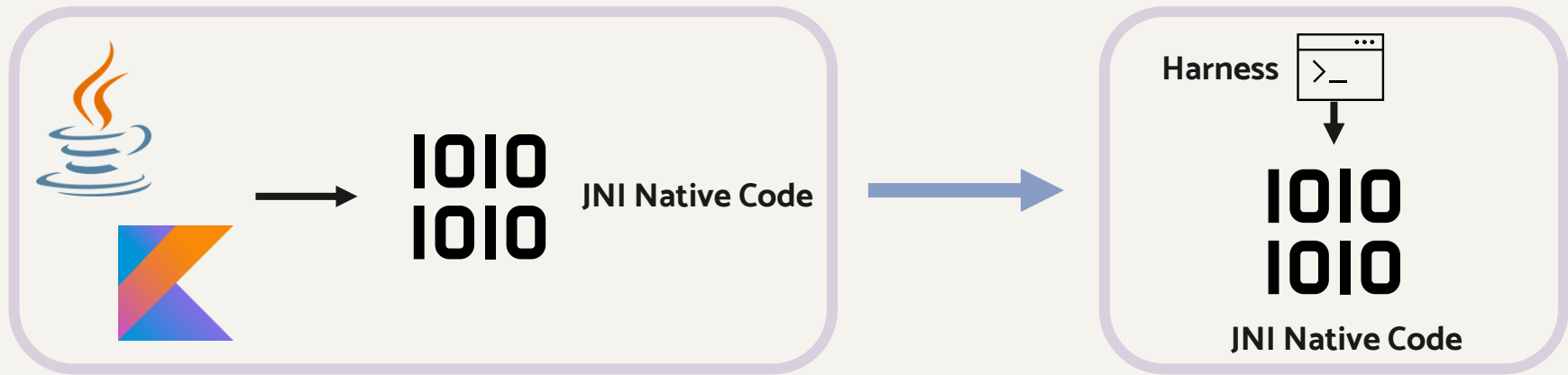
JNI Bug Finding

Key is to isolate this portion and run it as fast as possible



JNI Bug Finding

Generate fuzzing harnesses from how the Java code uses native libraries, and then directly fuzz the library.



JNI Bug Finding

Challenge: data
flow and method
invocations to
native code

```
long this.ptr;  
// ...
```

```
this.ptr = jni_funcA(...)
```

```
jni_funcB(this.ptr, ...)
```

jni_funcA

```
return struct_ptr
```

jni_funcB

```
struct_t ptr = ...;  
ptr->field_a;
```



**IOIO
IOIO**

JNI Native Code

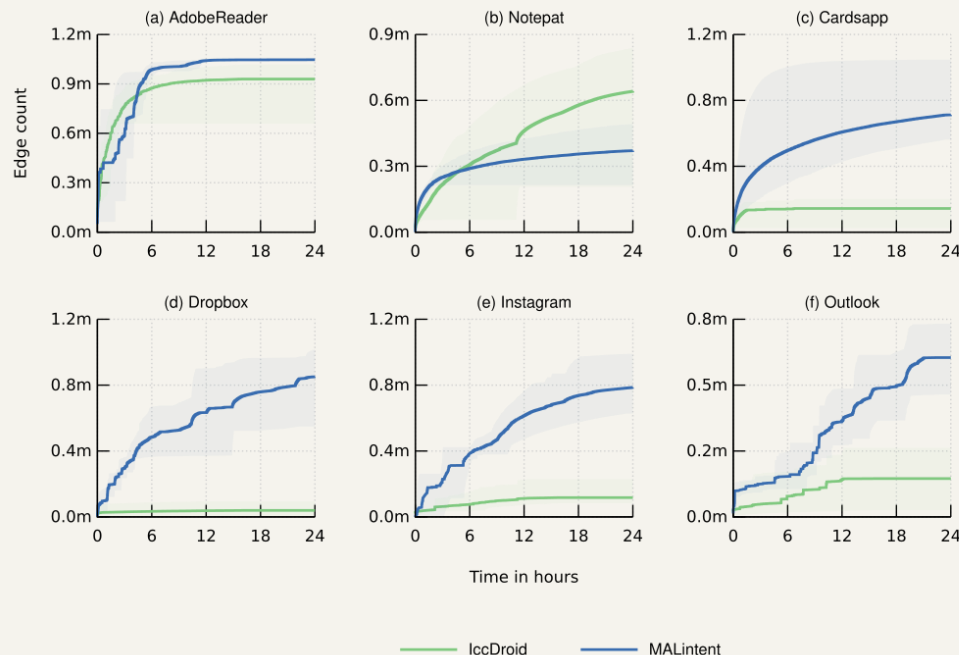
Evaluation

System	Coverage Instrumentation	Bug Types
MALintent	Yes	Crashes, Privacy, Memory
IccDroid	Only with Source Code	Crashes
Intents of Death	Only with Source Code	Crashes
DroidFuzzer	No	Crashes
Demissie <i>et al.</i>	No	Privacy
AndroidIntentFuzzer	No	Crashes
MindMacIntentFuzzer	No	Crashes

Evaluation

Coverage comparison
between MALintent and
IccDroid (previous state-of-the-art)

IccDroid does not support
coverage instrumentation
without source code.



Bugs Found

App	Component	Oracle	Description
Instagram	libnative-filters.so	Memory Safety	Out-of-bounds write in Fresco GUI library.
Chrome	TracingController	Privacy	Exposed profiler leaks private browser data.
WhatsApp	CameraActivity	Privacy	Sending an intent with <code>add_more_images</code> causes image to be taken without interaction.
TextNow	DialerActivity	Privacy	Intent with <code>answer_call</code> and <code>phone_number</code> causes app to dial number and pick up automatically.
AndroODB GPS	GpsProvider	Privacy	GPSPProvider leaks location data in the form of an intent result.
Rethink DNS	HomeScreenActivity	Privacy	Allows restoring configs from backup, can set a malicious DNS server and intercept all traffic.
OpenGPX	CacheListActivity	Privacy	App accepts arbitrary URI when copying gpx map file. Allows leak of all app data.
...			