

Analysis in Practice: Compositional Analysis of Android inter-app Security Vulnerabilities



FQ 2016

IN4MATX 221

Alireza Sadeghi

Outline

Motivation

- Mobile Security Threats
- Android Overview
- Inter-app Vulnerability

COVERT

- Static Analysis
- Formal Verification
- Challenges
- Evaluation
- Demo

Outline

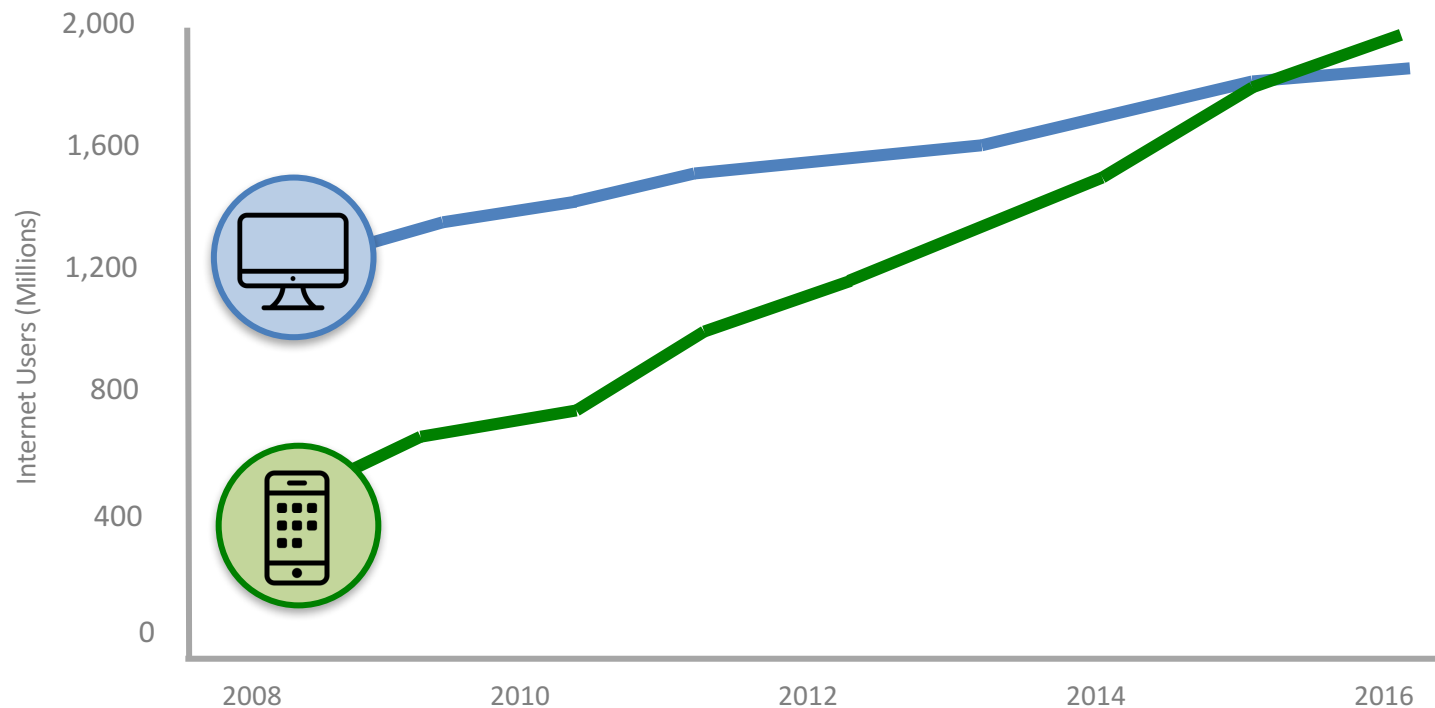
Motivation

- **Mobile Security Threats**
- Android Overview
- Inter-app Vulnerability

COVERT

- Static Analysis
- Formal Verification
- Challenges
- Evaluation
- Demo

The Rise of Mobile



Source:  comSCORE.

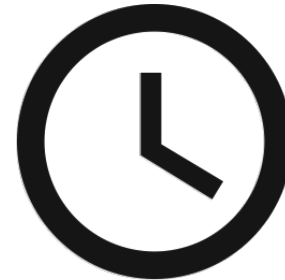
The Rise of Mobile

3.65 Billion



unique global mobile users

2:54 hours/day



Time spent on mobile device by Americans

\$400 Billion



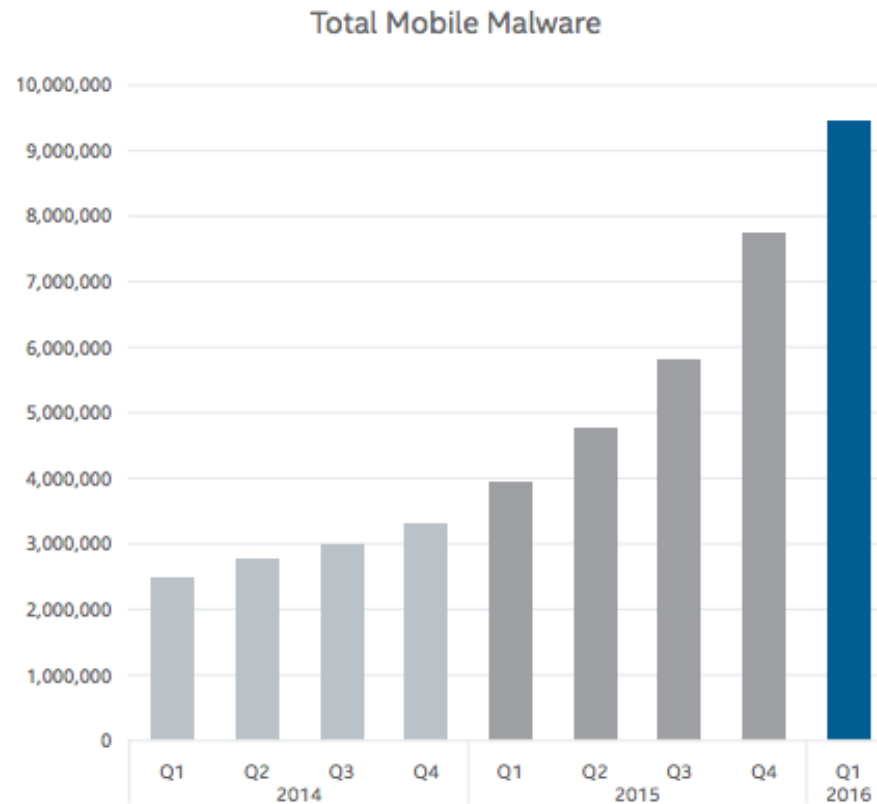
U.S. Mobile Revenue

55%



Share of Mobile devices for U.S Internet usage

The Rise of ~~Mobile~~ Security Threats



Source:  McAfee

The Rise of ~~Mobile~~ Security Threats

Forbes / Security

JUL 27, 2015 @ 06:00 AM 134,744 VIEWS

Stagefright: It Only Takes One Text To Hack 950 Million Android Phones



Thomas Fox-Brewster

Six critical vulnerabilities have left 95 per cent of Google GOOGL -1.07% Android phones open to an attack, a security expert warned today. In some cases, where phones parse the attack code prior to the message, a user would have little chance of defending their data. The vulnerabilities are said to be the worst Android

Joshua Drake, from Zimperium zLabs, who reported the bugs in April this year, said whilst Google and most manufacturers have not made fixes available to protect their customers. "All devices should be protected by platform research and exploitation at Zimperium, told FORBES. He believes as many as 950 million



The Washington Post Your Android smartphone could be attacked with a text message

Andrea Peterson July 28 Follow @karsennet



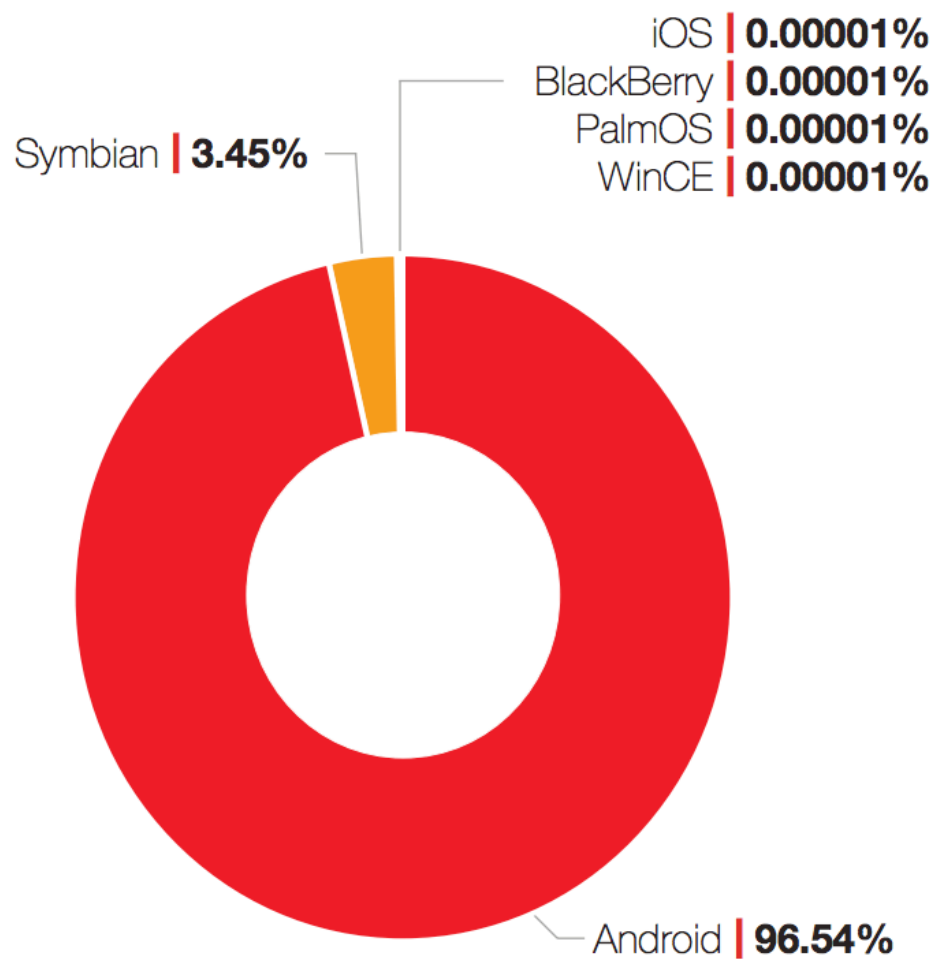
A hacker could take over Android smartphones with nothing more than a text message, according to mobile security experts at cybersecurity firm

Most Read

- 1 "Pathological" definitely doesn't mean what Donald Trump thinks it means
- 2 What a creepy Bloomberg's ad tells us about America's understanding of rape
- 3 This is how glibble General Mills thinks Americans are
- 4 Scientists say Greenland just opened up a major new 'biogate' of ice into the ocean
- 5 Clinton and Sanders are divided over a big Obama promise: Not raising taxes on the middle class

Unlimited Access to The Post. Just 99¢

Android is the Primary Target

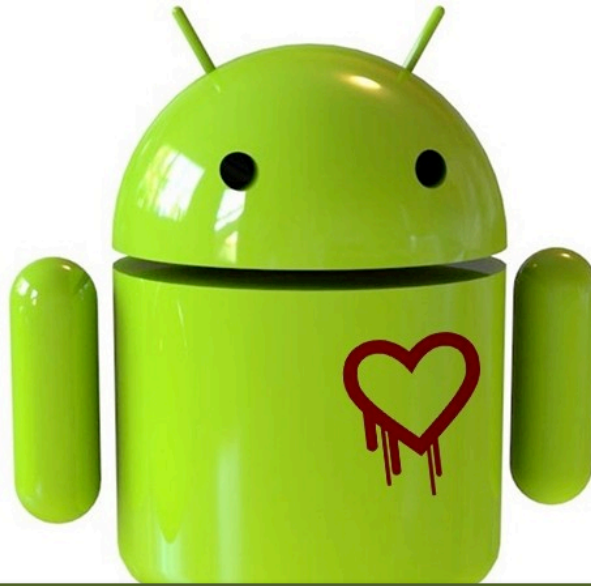


Source: **FORTINET**



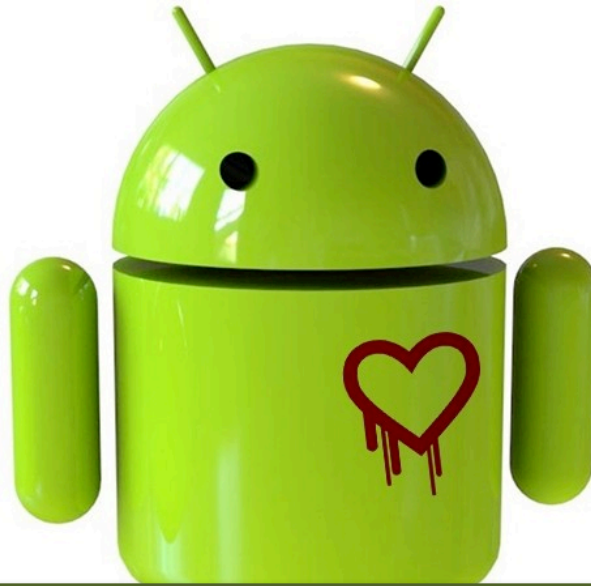
What makes Android so vulnerable?

- Most Popular (Global over 80%, U.S. 60%)



What makes Android so vulnerable?

- Open Platform



What makes Android so vulnerable?

- Security flaws in Android

Outline

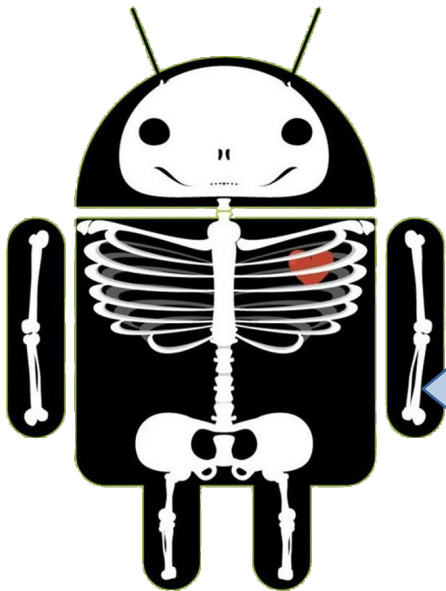
Motivation

- Mobile Security Threats
- **Android Overview**
- Inter-app Vulnerability

COVERT

- Static Analysis
- Formal Verification
- Challenges
- Evaluation
- Demo

Android Apps: Components



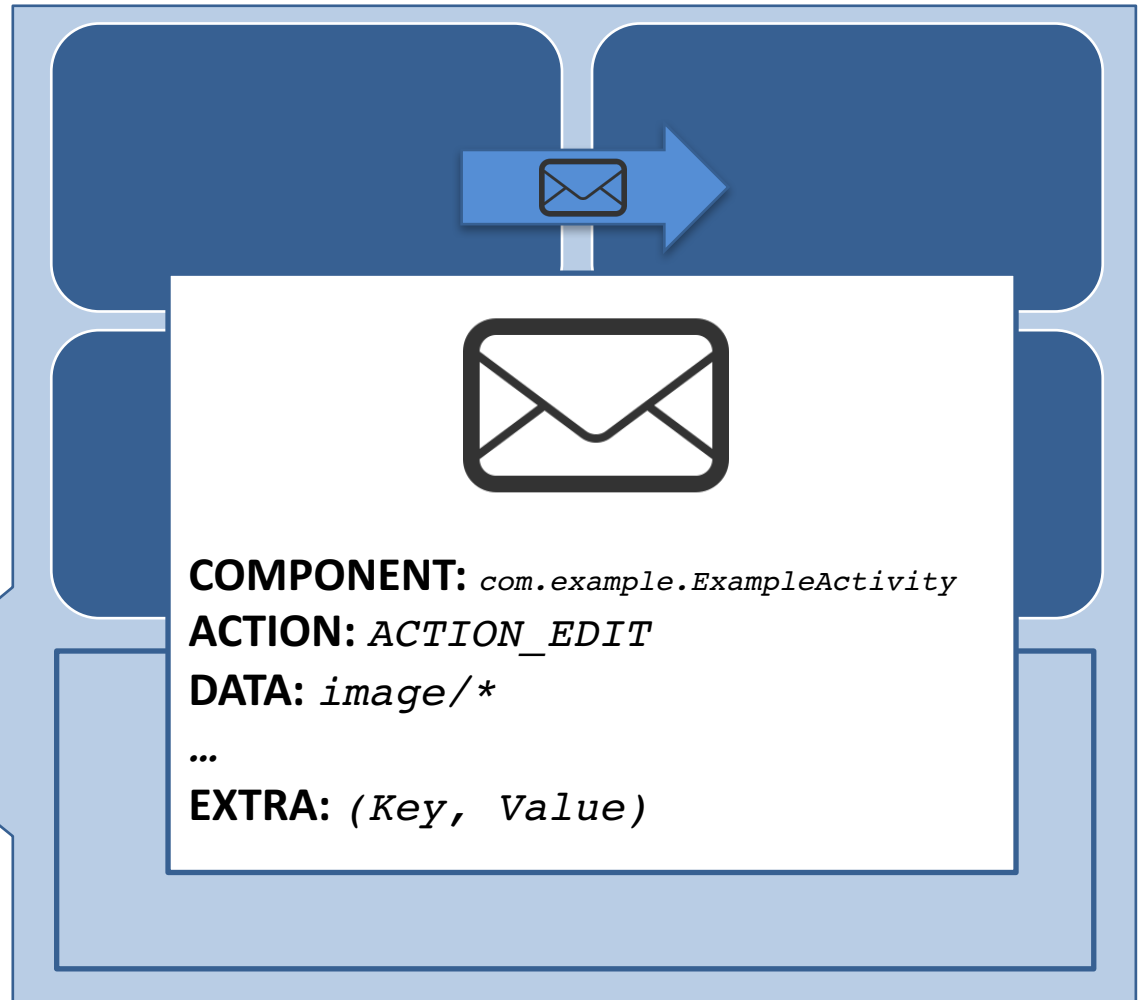
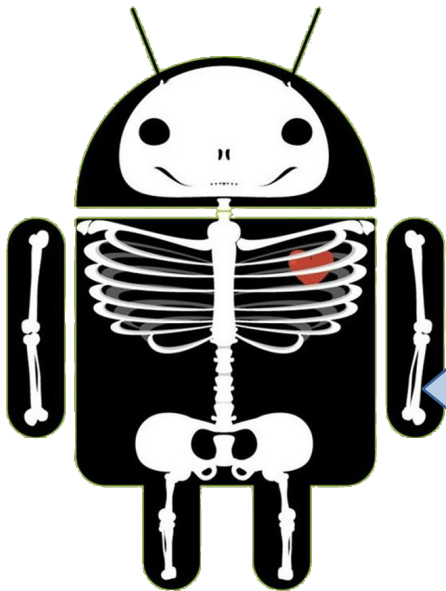
Activity
Provides User
Interface

Service
Executes Background
Processes

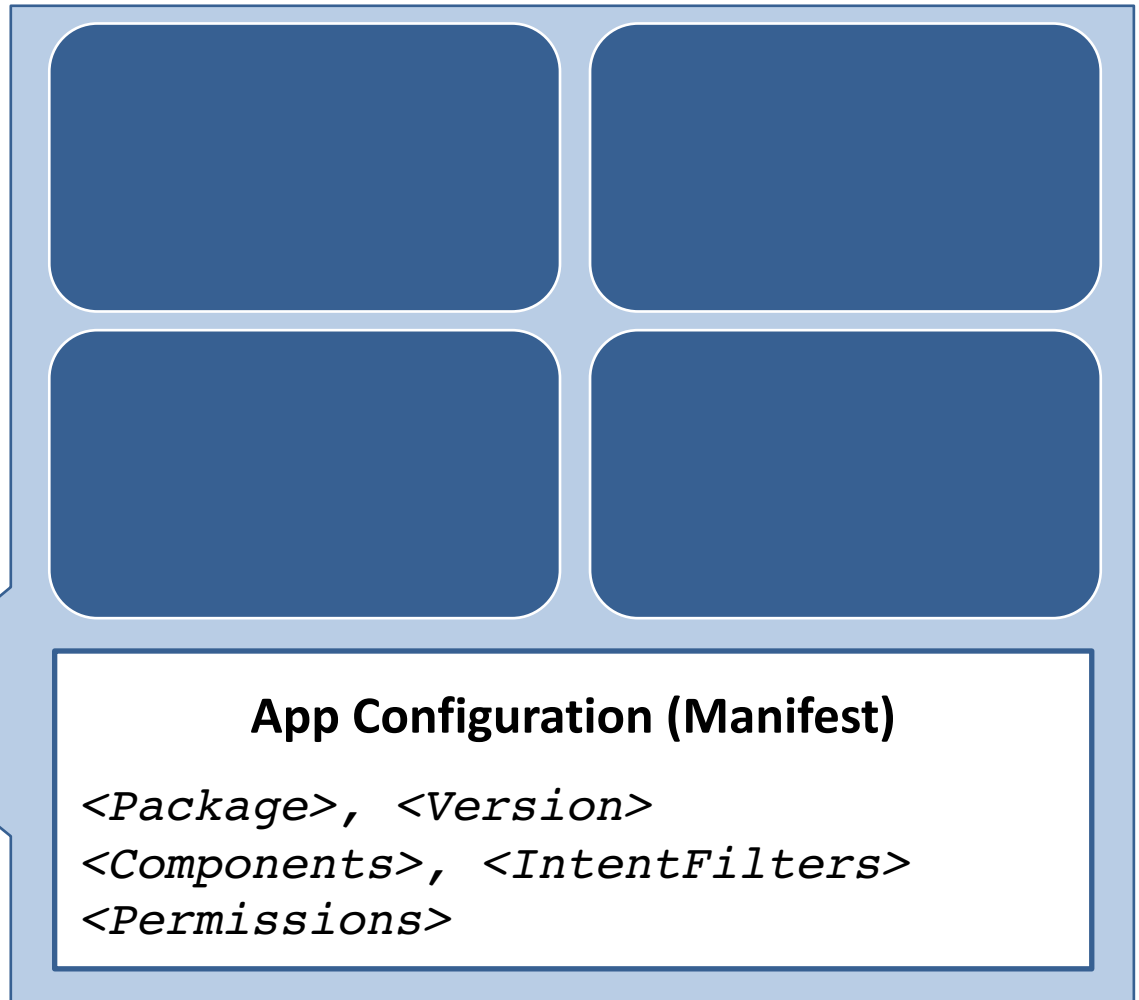
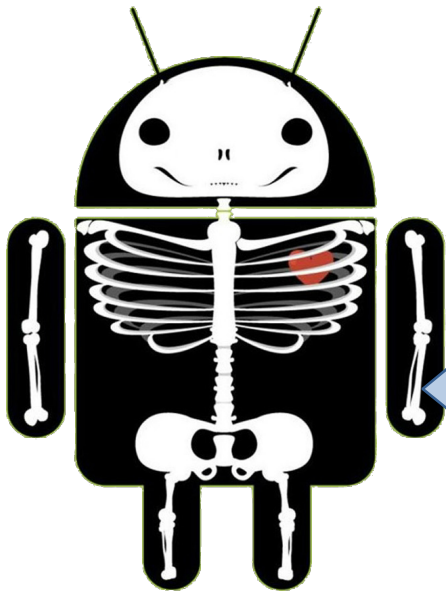
Content Provider
Data Sharing

Broadcast Receiver
Responds to system-
wide announcement
messages

Android Apps: **Intents**



Android Apps: Manifest



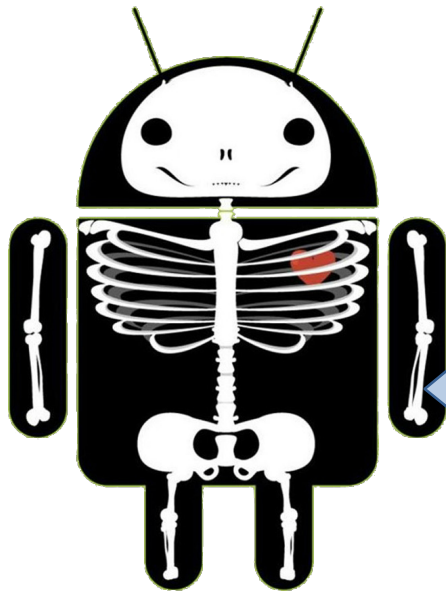
App Configuration (Manifest)

<Package>, <Version>

<Components>, <IntentFilters>

<Permissions>

Android Apps: Permissions



< v. 6

App info

PERMISSIONS

This app can access the following on your phone:

- record audio
- precise location (GPS and network-based)
- read your contacts
- modify or delete the contents of your USB storage
read the contents of your USB storage
- find accounts on the device
read Google service configuration
use accounts on the device
- full network access
view network connections
view Wi-Fi connections
- run at startup
- control vibration
prevent phone from sleeping
- read sync settings

Static, install-time

≥ v. 6

App permissions

Keep

- Contacts ☒
- Location ☒
- Microphone ☐
- Storage ☒

Dynamic, Run-time

Outline

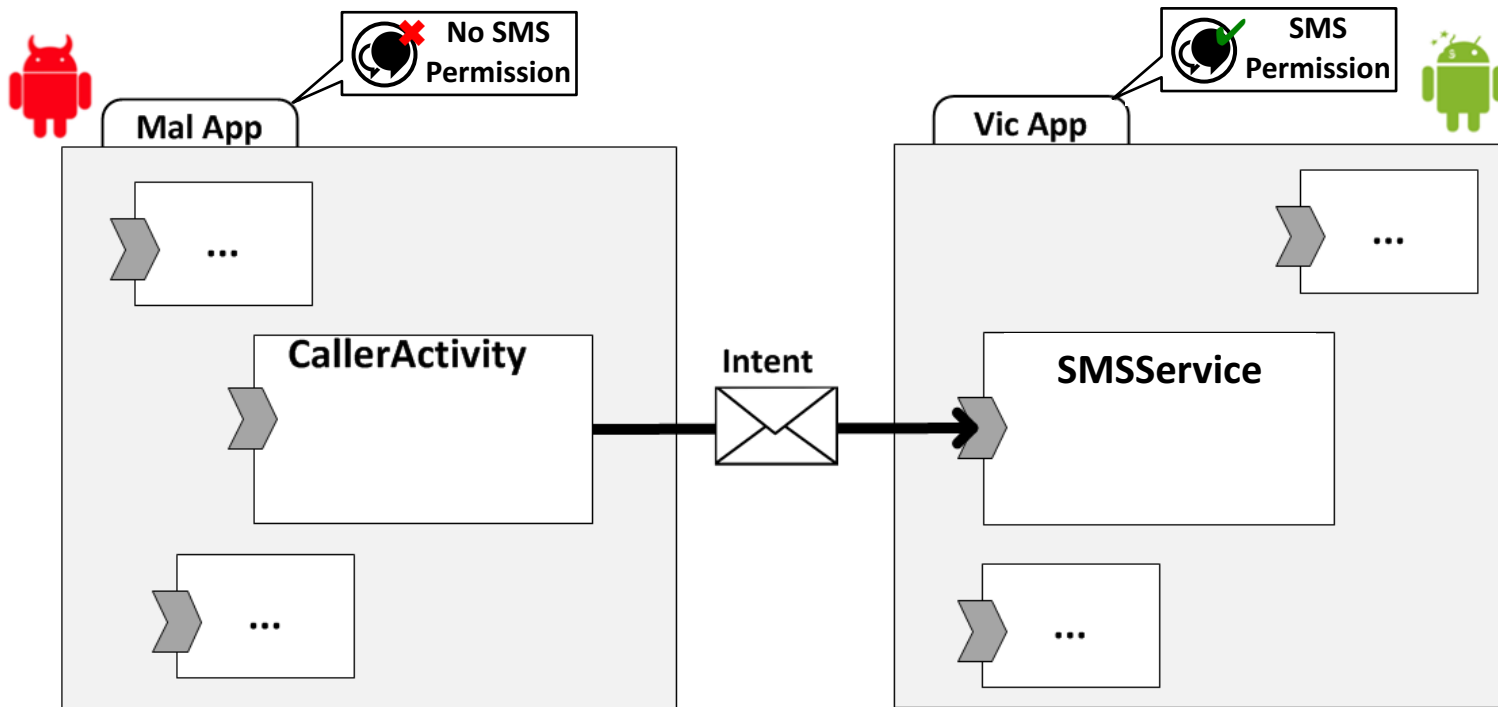
Motivation

- Mobile Security Threats
- Android Overview
- **Inter-app Vulnerability**

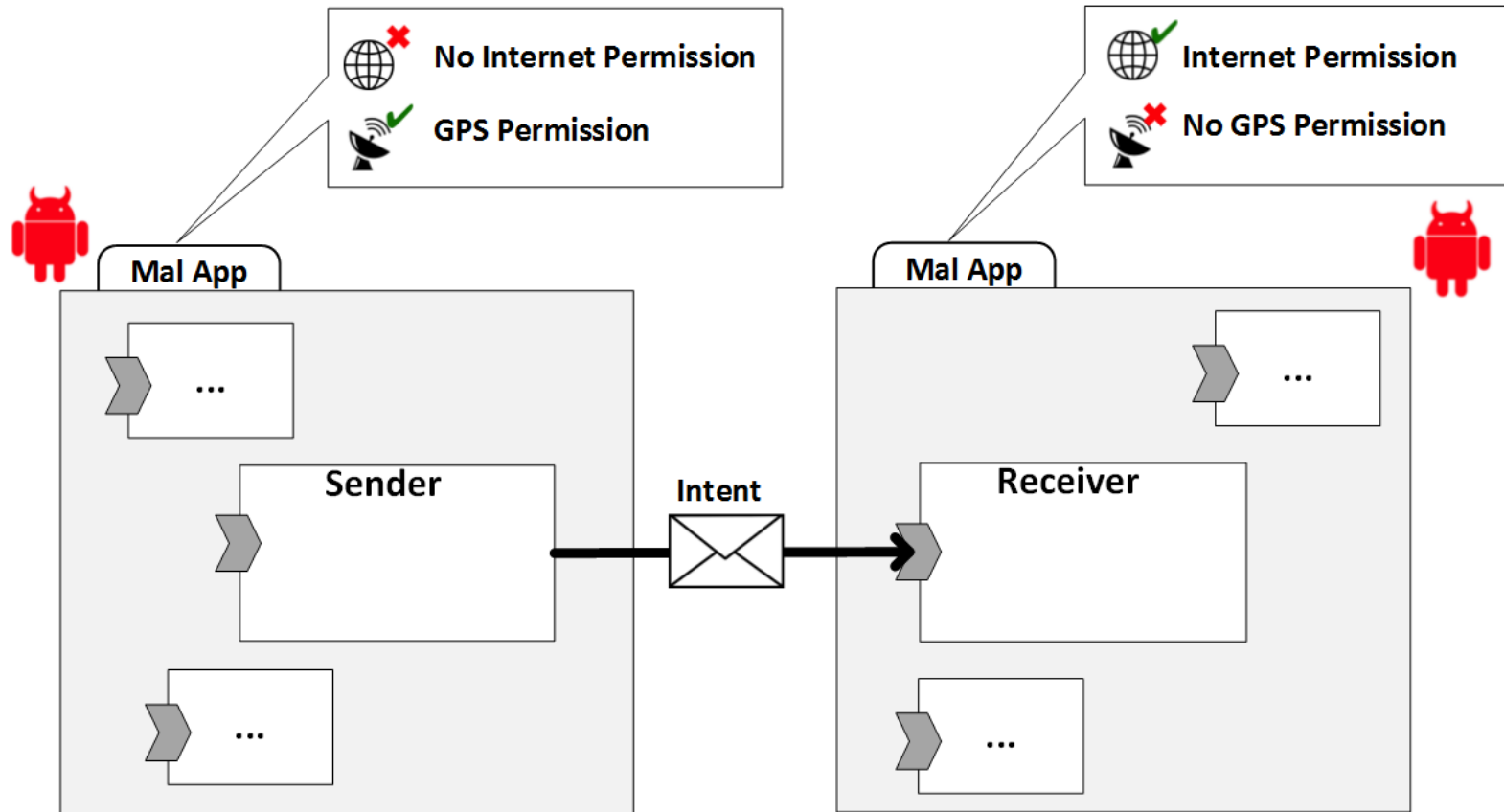
COVERT

- Static Analysis
- Formal Verification
- Challenges
- Evaluation
- Demo

Inter-app vulnerability example: privilege escalation



Inter-app vulnerability example: app collusion



Outline

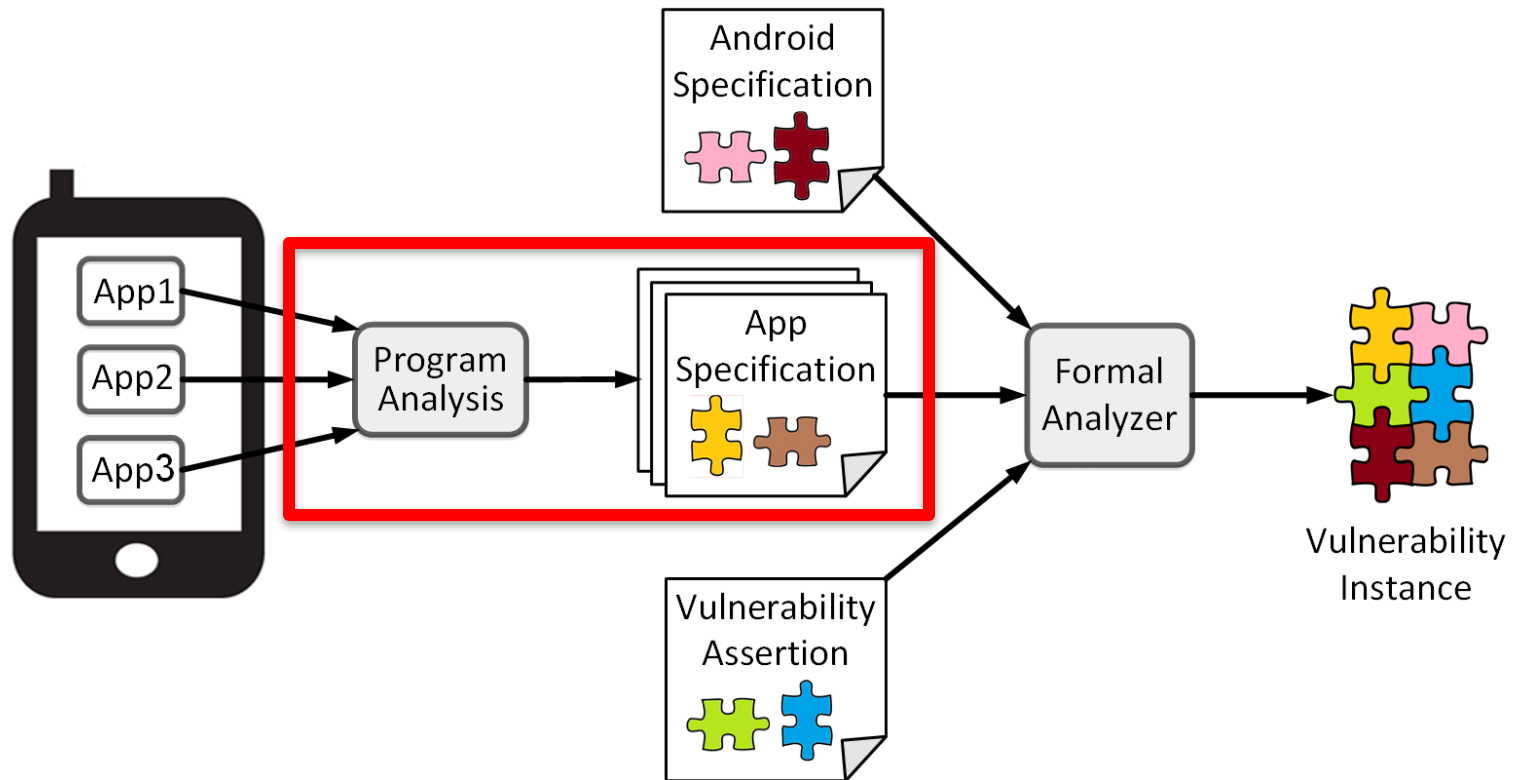
Motivation

- Mobile Security Threats
- Android Overview
- Inter-app Vulnerability

COVERT

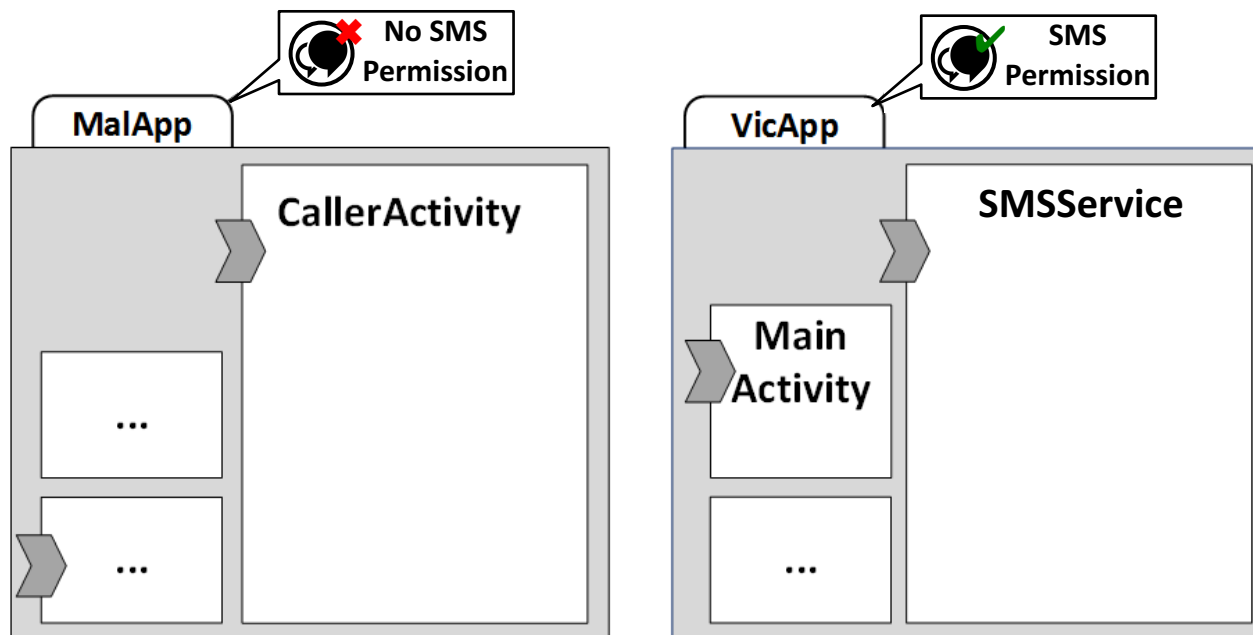
- **Static Analysis**
- Formal Verification
- Challenges
- Evaluation
- Demo

COVERT: Compositional Analysis of Inter-app Vulnerabilities



Static extraction of relevant elements

1. Principal entities and properties defined in the manifest file



Static extraction of relevant elements

1. Principal entities and properties defined in the manifest file

```

<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.VIBRATE"/>
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

<permission...>
<uses-permission android:name="${applicationId}.permission.READ_ATTACHMENT"/>

<permission...>
<uses-permission android:name="${applicationId}.permission.REMOTE_CONTROL"/>

<permission...>
<uses-permission android:name="${applicationId}.permission.READ_MESSAGES"/>

<permission...>
<uses-permission android:name="${applicationId}.permission.DELETE_MESSAGES"/>

<application
    android:name="K9"
    android:allowTaskReparenting="false"
    android:icon="@drawable/icon"
    android:label="K-9 Mail"
    android:theme="@style/Theme.K9.Startup"
    android:allowBackup="false">

<receiver
    android:name=".service.BootReceiver"
    android:enabled="true">
    <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.DEVICE_STORAGE_LOW"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.DEVICE_STORAGE_OK"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.net.conn.CONNECTIVITY_CHANGE"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.net.conn.BACKGROUND_DATA_SETTING_CHANGED"/>
    </intent-filter>
    <intent-filter>
        <action android:name="com.android.sync.SYNC_CONN_STATUS_CHANGED"/>
    </intent-filter>
</receiver>

```

```

<activity
    android:name=".activity.MessageCompose"
    android:configChanges="locale"
    android:enabled="false"
    android:label="K-9 Mail">
    <intent-filter>
        <action android:name="android.intent.action.SENDTO"/>
        <data android:scheme="mailto"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <data android:mimeType="*//*"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <data android:mimeType="*//*"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.VIEW"/>
        <data android:scheme="mailto"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <category android:name="android.intent.category.BROWSABLE"/>
    </intent-filter>
</activity>

<service
    android:name=".service.SleepService"
    android:enabled="true"/>

<service
    android:name=".service.DatabaseUpgradeService"
    android:exported="false"/>

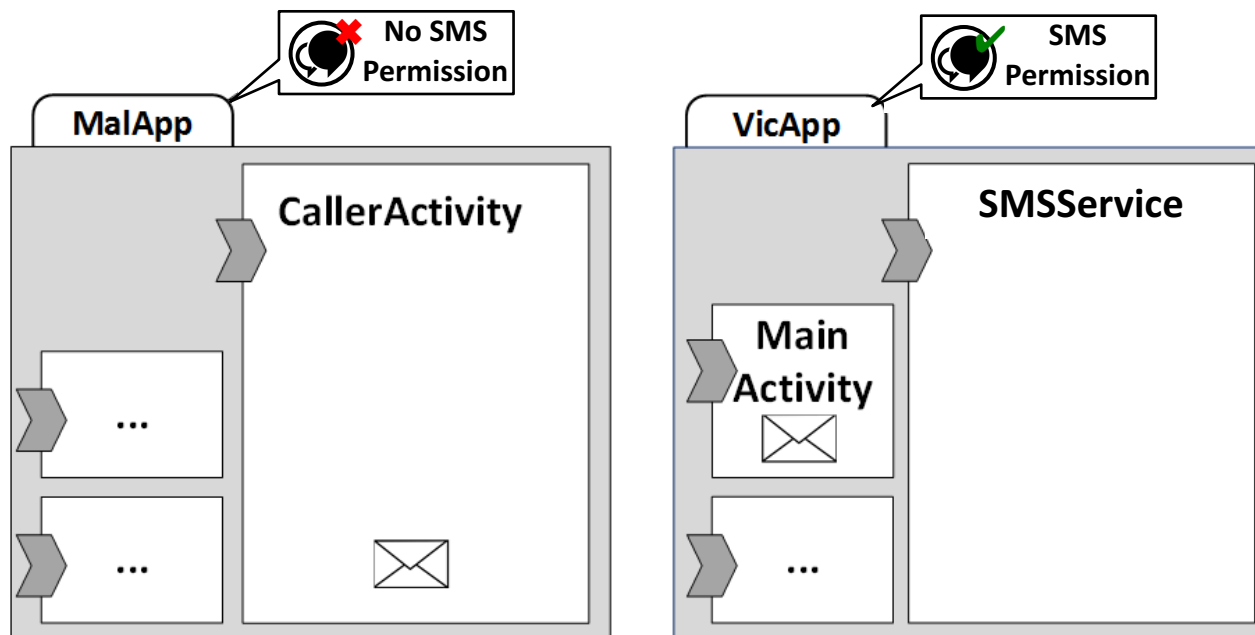
<provider
    android:name=".provider.AttachmentProvider"
    android:authorities="${applicationId}.attachmentprovider"
    android:exported="true"
    android:grantUriPermissions="true"
    android:multiprocess="true"
    android:readPermission="${applicationId}.permission.READ_ATTACHMENT">

    <meta-data
        android:name="de.cketti.safecontentresolver.ALLOW_INTERNAL_ACCESS"
        android:value="true" />
</provider>

```

Static extraction of relevant elements

1. Principal entities and properties defined in the manifest file
2. Principal entities (e.g., Intent and Filters) that are latent in code



Static extraction of relevant elements

1. Principal entities and properties defined in the manifest file
2. Principal entities (e.g., Intent and Filters) that are latent in code

```
public static void listAccounts(Context context) {
    Intent intent = new Intent(context, Accounts.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TOP |
        Intent.FLAG_ACTIVITY_SINGLE_TOP | Intent.FLAG_ACTIVITY_CLEAR_TASK);
    intent.putExtra(EXTRA_STARTUP, false);
    context.startActivity(intent);
}
```

```
protected void registerReceivers() {
    final StorageGoneReceiver receiver = new StorageGoneReceiver();
    final IntentFilter filter = new IntentFilter();
    filter.addAction(Intent.ACTION_MEDIA_EJECT);
    filter.addAction(Intent.ACTION_MEDIA_UNMOUNTED);
    filter.addDataScheme("file");

    final BlockingQueue<Handler> queue = new SynchronousQueue<>();

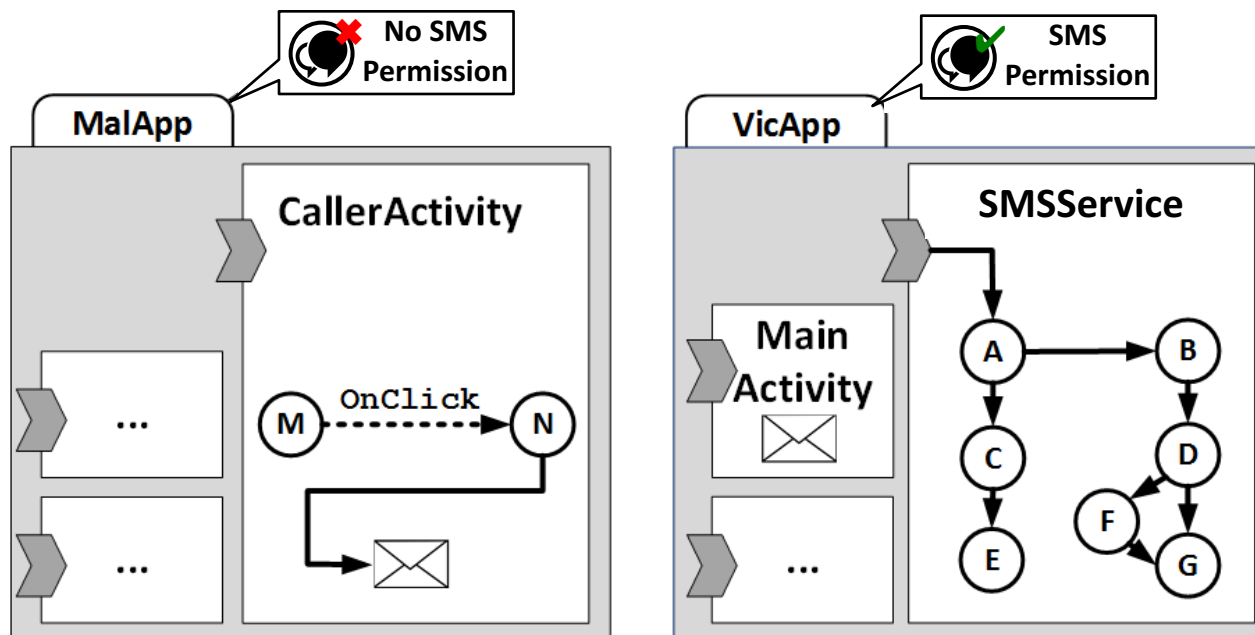
    // starting a new thread to handle unmount events
    new Thread(new Runnable() {
        @Override
        public void run() {
            Looper.prepare();
            try {
                queue.put(new Handler());
            } catch (InterruptedException e) {
                Log.e(K9.LOG_TAG, "", e);
            }
            Looper.loop();
        }
    }, "Unmount-thread").start();

    try {
        final Handler storageGoneHandler = queue.take();
        registerReceiver(receiver, filter, null, storageGoneHandler);
        Log.i(K9.LOG_TAG, "Registered: unmount receiver");
    } catch (InterruptedException e) {
        Log.e(K9.LOG_TAG, "Unable to register unmount receiver", e);
    }

    registerReceiver(new ShutdownReceiver(), new IntentFilter(Intent.ACTION_SHUTDOWN));
    Log.i(K9.LOG_TAG, "Registered: shutdown receiver");
}
```

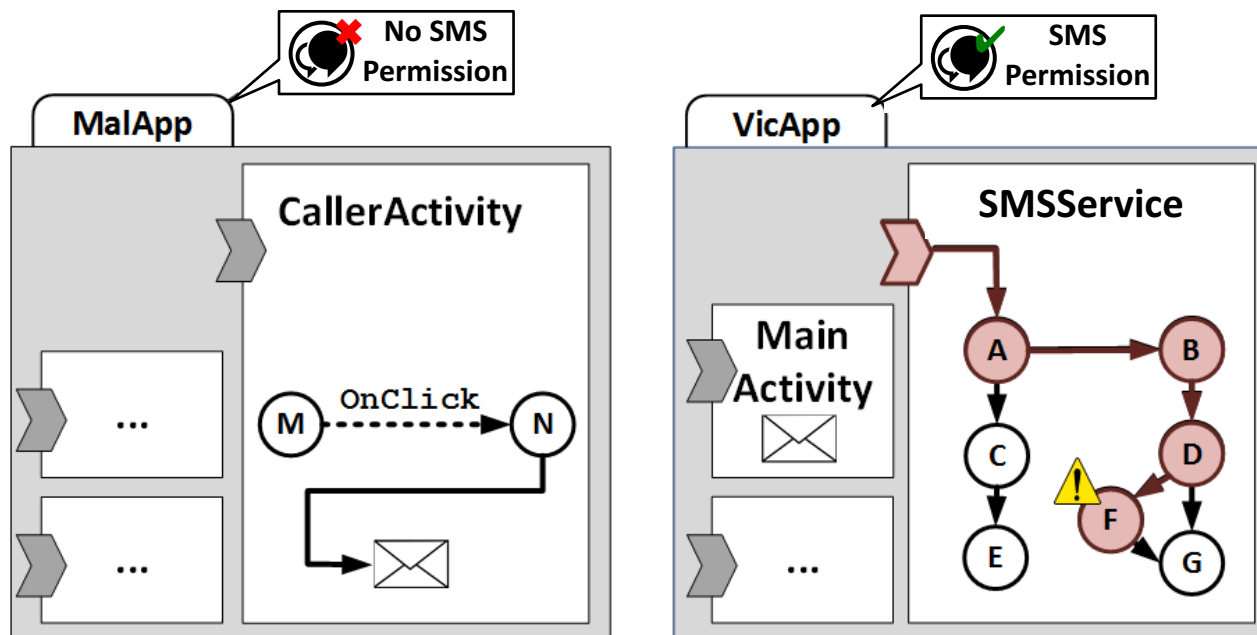
Static extraction of relevant elements

1. Principal entities and properties defined in the manifest file
2. Principal entities (e.g., Intent and Filters) that are latent in code
3. Event-driven behavior of each app

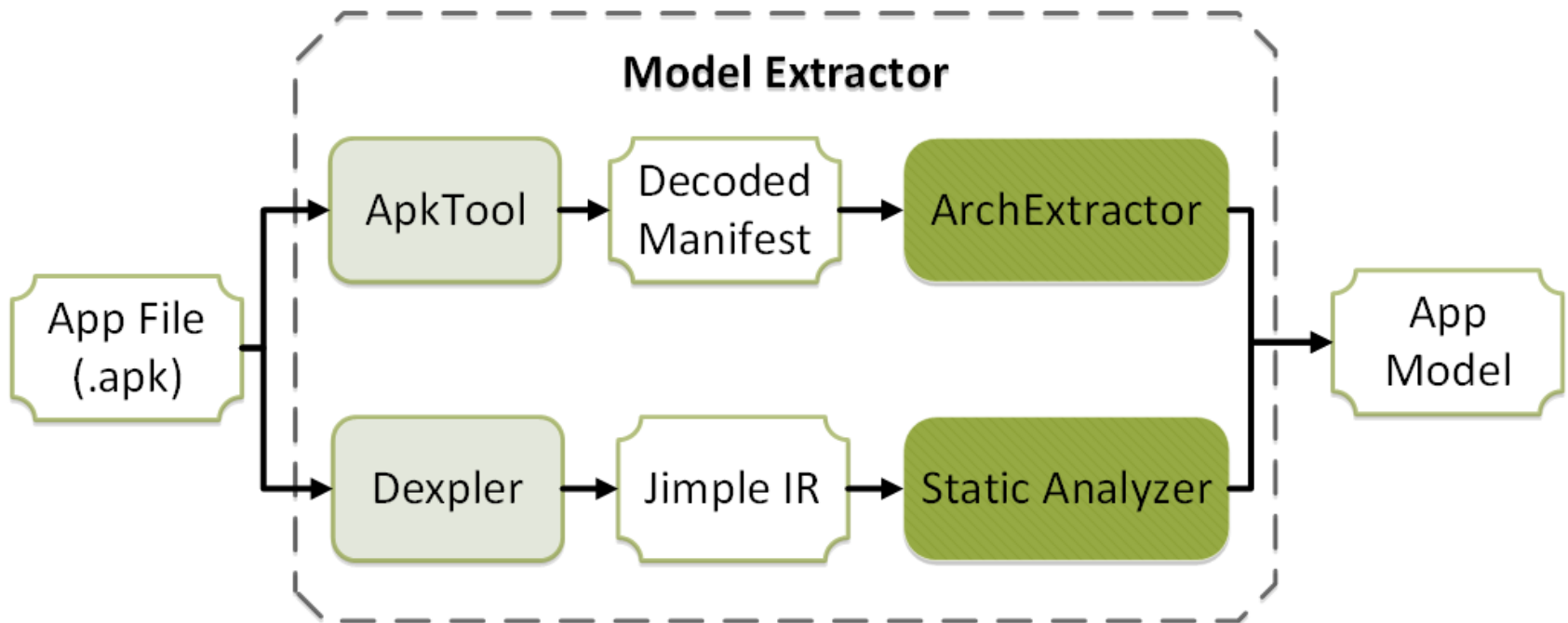


Static extraction of relevant elements

1. Principal entities and properties defined in the manifest file
2. Principal entities (e.g., Intent and Filters) that are latent in code
3. Event-driven behavior of each app
4. Sensitive Paths



Static extraction of relevant elements



Static Analysis

- Manual Security Assessment
 - labor intensive
 - error-prone

```
1 public int[] handleSqlCommands(String data, String action) {
2     sanitizeData(data);
3     Connection dbConnection = IO.getDBConnection();
4     Statement sqlStmt = dbConnection.createStatement();
5     /* if (action == null) return null; */
6     if (action.equals("getBatch")) {
7         String names[] = data.split("-");
8         for (int i = 0; i < names.length; i++) {
9             /* Potential SQL Injection */
10            sqlStmt.addBatch("SELECT * FROM users WHERE name='"+names[i]+'");
11            int resultsArray[] = sqlStmt.executeBatch();
12            return resultsArray;
13        }
14        if (action.equals("updateUserStatus")) {
15            /* Potential SQL Injection */
16            int rowCount = sqlStmt.executeUpdate("INSERT INTO users(status) VALUES
17                ('updated') WHERE name='"+data+"'");
18            int resultsArray[] = {rowCount};
19            return resultsArray;
20        }
21        return null;
22    }
```

Static Analysis

- Static Analysis
 - Automatically examines software for a specific property (e.g., security) without executing the program.
 - Extracts abstract representation of the code (e.g. Call Graph)

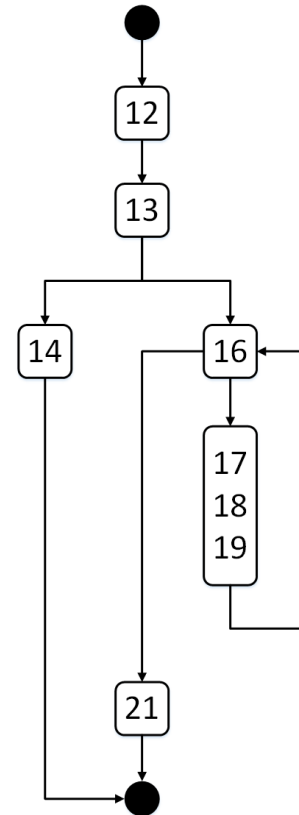
Static Analysis

```

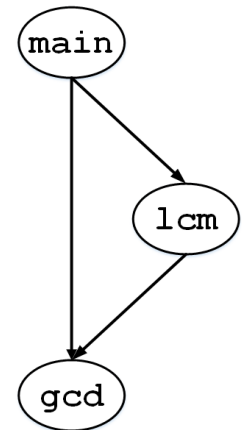
1  public static void main (String[] args) {
2      //num1 and num2 are defined here ...
3      int result1 = gcd(num1, num2);
4      int result2 = lcm(num1,num2);
5  }
6  static int lcm (int a, int b){
7      int gcd = gcd(a, b);
8      int lcm = (a*b)/gcd;
9      return lcm;
10 }
11 static int gcd (int a, int b){
12     int c;
13     if ( b == 0 ){
14         return a;
15     } else {
16         while ( b != 0 ){
17             c = b;
18             b = a % b;
19             a = c;
20         }
21         return a;
22     }
23 }

```

(a)



(b)



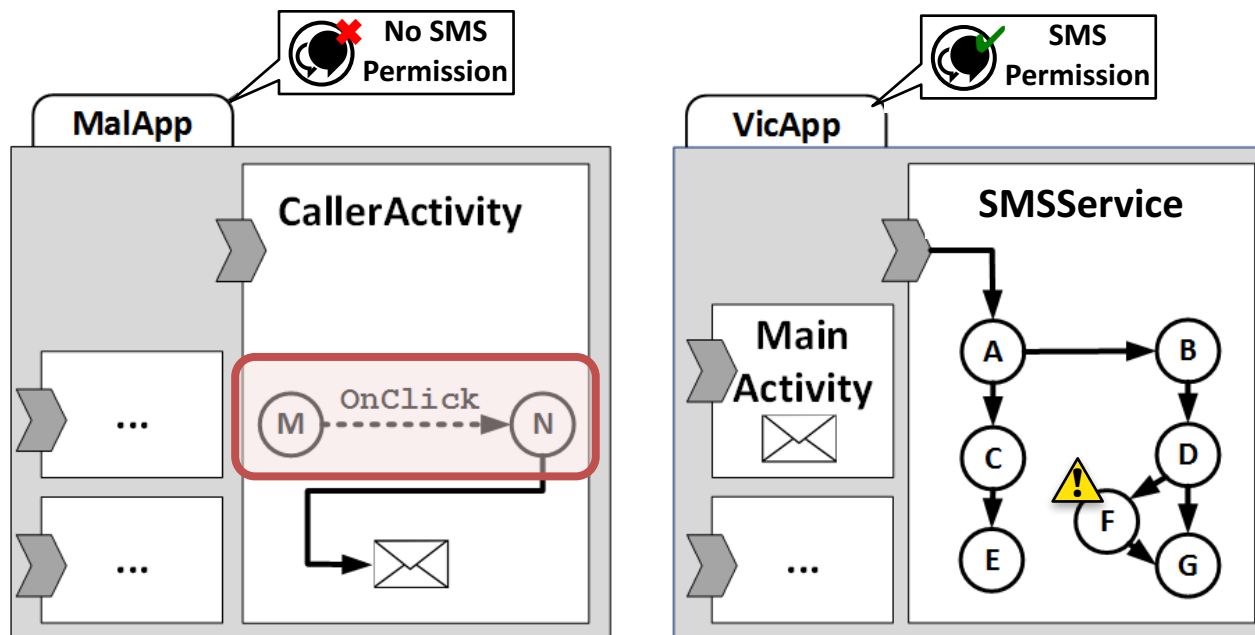
(c)

Static vs. Dynamic analysis

- **Static**
 - Sound but Conservative (Over-approximate)
 - More False Positives
- **Dynamic**
 - Unsound but Precise (Under-approximate)
 - More False Negative

Challenges of Static Analysis of Android

C1 Event-Driven Structure



Challenges of Static Analysis of Android

C1 Event-Driven Structure

```
@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    String item = (String)parent.getItemAtPosition(position);

    Toast.makeText(EmailAddressList.this, item, Toast.LENGTH_LONG).show();

    Intent intent = new Intent();
    intent.putExtra(EXTRA_EMAIL_ADDRESS, item);
    setResult(RESULT_OK, intent);
    finish();
}
```

User event (e.g. Click)

```
class MyLocationManager implements LocationListener {

    @Override
    public void onLocationChanged(Location location) {

    }

    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {

    }

    @Override
    public void onProviderEnabled(String provider) {

    }

    @Override
    public void onProviderDisabled(String provider) {

    }
}
```

System event
(e.g. Location Changed)

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}

@Override
protected void onStart() {
    super.onStart();
}

@Override
protected void onRestart() {
    super.onRestart();
}

@Override
protected void onResume() {
    super.onResume();
}

@Override
protected void onPause() {
    super.onPause();
}

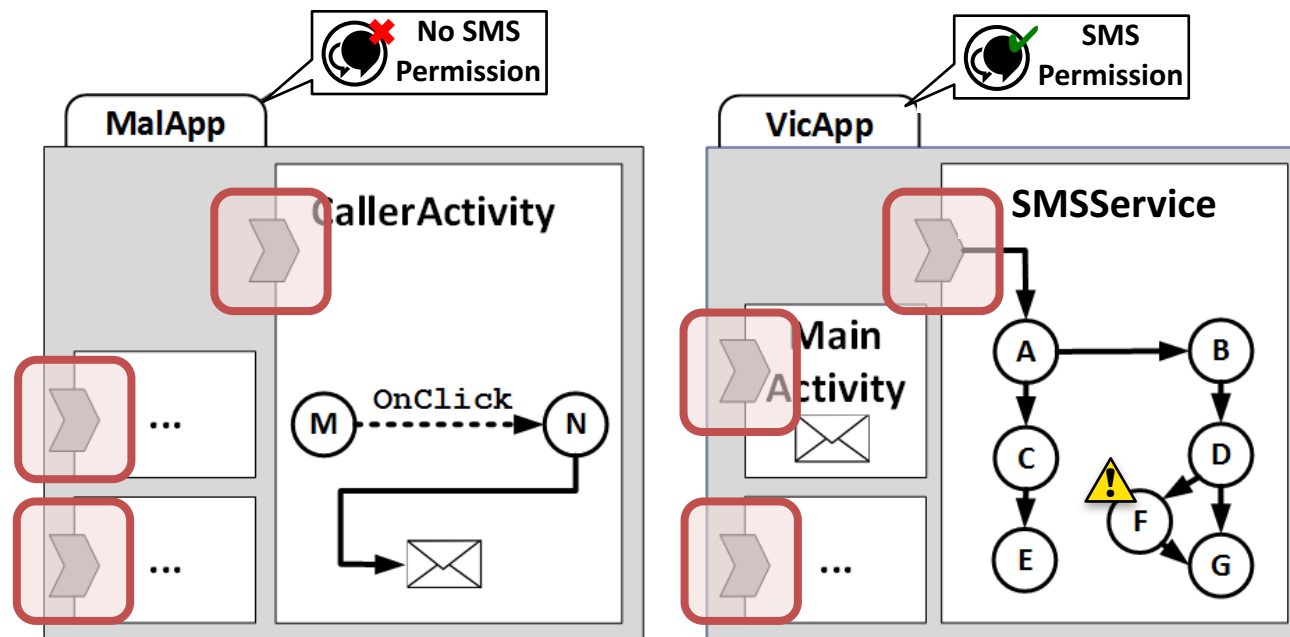
@Override
protected void onStop() {
    super.onStop();
}

@Override
protected void onDestroy() {
    super.onDestroy();
}
```

Component's life-cycle event
(e.g. Location Changed)

Challenges of Static Analysis of Android

C2 Multiple Entry Points



Challenges of Static Analysis of Android

c2 Multiple Entry Points

```
@Override
protected void onCreate(Bundle savedInstanceState) {...}

@Override
public boolean onCreateOptionsMenu(Menu menu) {...}

@Override
public boolean onOptionsItemSelected(MenuItem item) {...}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,

@Override
public void onClick(View v) { submitRating(); }
```

Android

```
public static void main (String[] args) {
    int[] nums = getInput(args);
    int num1 = nums[0];
    int num2 = nums[1];
    int result1 = gcd(num1, num2);
    int result2 = lcm(num1, num2);
}

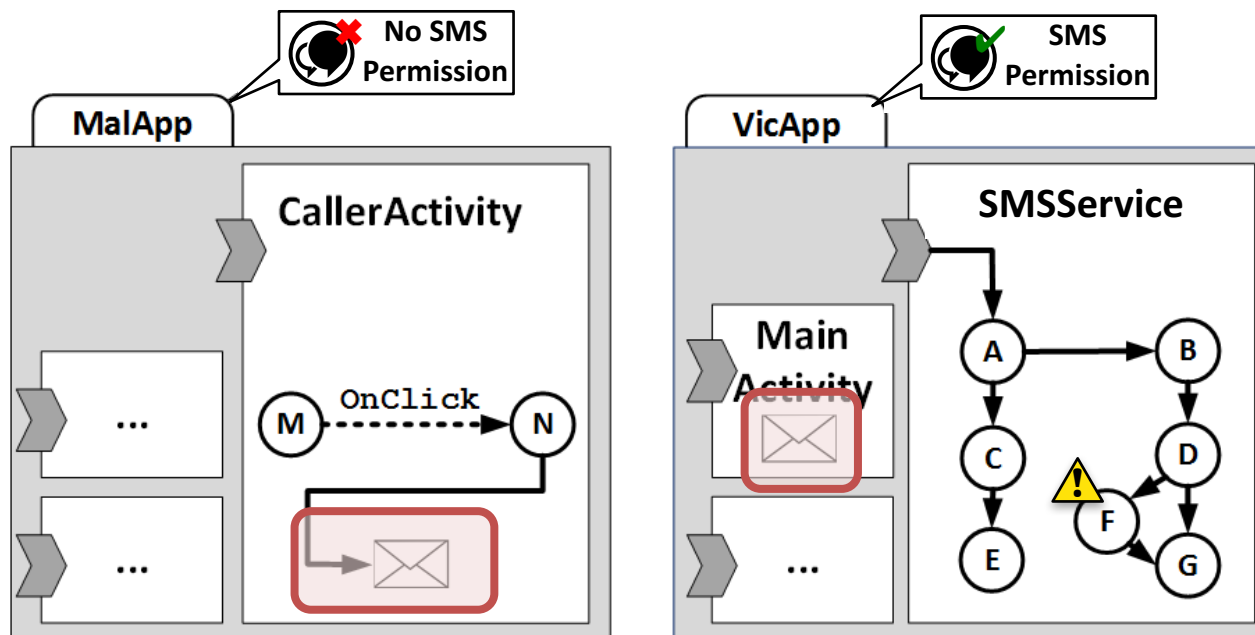
private static int lcm (int a, int b){...}
private static int gcd (int a, int b){...}

private static int[] getInput(String[] args) {
    return new int[]{1, 2};
}
```

POJO

Challenges of Static Analysis of Android

C3 Inter-component communication



Challenges of Static Analysis of Android

C3 Inter-component communication

```
public void sendAlternate(Context context, Account account, LocalMessage message) {
    if (K9.DEBUG)
        Log.d(K9.LOG_TAG, "Got message " + account.getDescription() + ":" + message.getFolder()
            + ":" + message.getUid() + " for sendAlternate");

    Intent msg = new Intent(Intent.ACTION_SEND);
    String quotedText = null;
    Part part = MimeUtility.findFirstPartByMimeType(message, "text/plain");
    if (part == null) {
        part = MimeUtility.findFirstPartByMimeType(message, "text/html");
    }
    if (part != null) {
        quotedText = MessageExtractor.getTextFromPart(part);
    }
    if (quotedText != null) {
        msg.putExtra(Intent.EXTRA_TEXT, quotedText);
    }
    msg.putExtra(Intent.EXTRA_SUBJECT, message.getSubject());

    Address[] from = message.getFrom();
    String[] senders = new String[from.length];
    for (int i = 0; i < from.length; i++) {
        senders[i] = from[i].toString();
    }
    msg.putExtra(Intent.EXTRA_FROM, senders);

    Address[] to = message.getRecipients(RecipientType.TO);
    String[] recipientsTo = new String[to.length];
    for (int i = 0; i < to.length; i++) {
        recipientsTo[i] = to[i].toString();
    }
    msg.putExtra(Intent.EXTRA_EMAIL, recipientsTo);

    Address[] cc = message.getRecipients(RecipientType.CC);
    String[] recipientsCc = new String[cc.length];
    for (int i = 0; i < cc.length; i++) {
        recipientsCc[i] = cc[i].toString();
    }
    msg.putExtra(Intent.EXTRA_CC, recipientsCc);

    msg.setType("text/plain");
    context.startActivity(Intent.createChooser(msg, "Choose sender"));
}
```

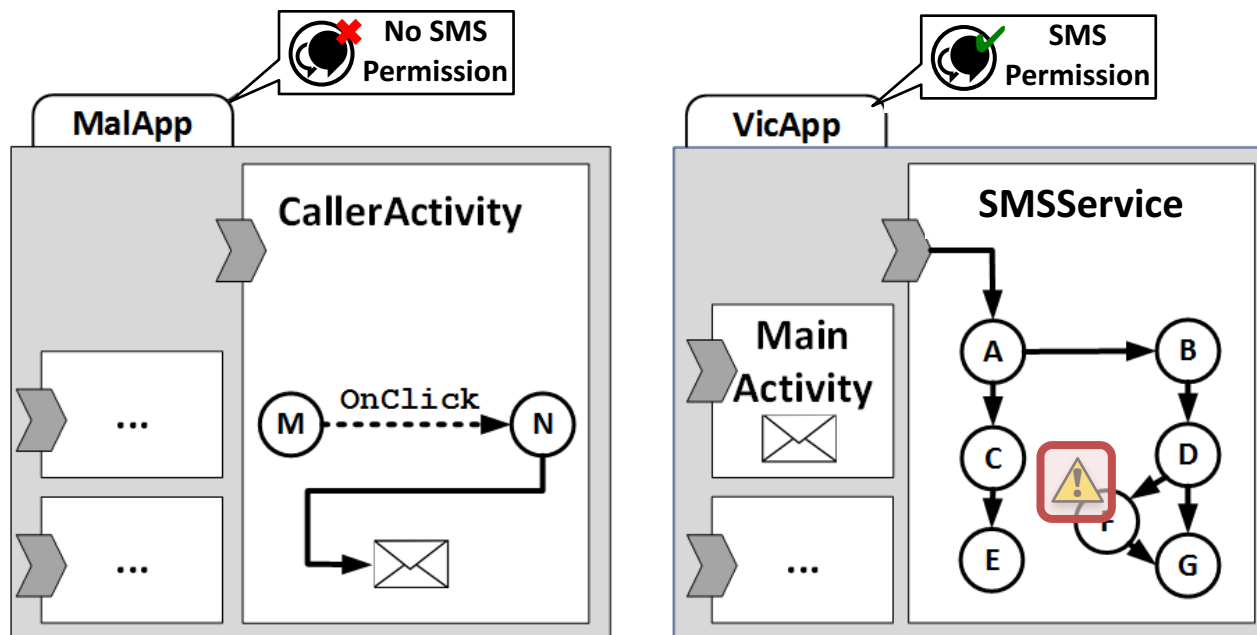
Implicit Intent

```
public static void listAccounts(Context context) {
    Intent intent = new Intent(context, Accounts.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TOP |
        Intent.FLAG_ACTIVITY_SINGLE_TOP | Intent.FLAG_ACTIVITY_CLEAR_TASK);
    intent.putExtra(EXTRA_STARTUP, false);
    context.startActivity(intent);
}
```

Explicit Intent

Challenges of Static Analysis of Android

C4 Modeling the underlying framework



Challenges of Static Analysis of Android

C4

Modeling the underlying framework

```
static Location getLastKnownLocation(Context context) {
    LocationManager locationManager = (LocationManager) context.getSystemService(LOCATION_SERVICE);
    locationManager.requestLocationUpdates(GPS_PROVIDER, 0, 0, new MyLocationManager());
    List<String> providers = locationManager.getProviders(true);
    Location bestLocation = null;
    for (String provider : providers) {
        Location l = locationManager.getLastKnownLocation(provider);
        if (l == null) {
            continue;
        }
        if (bestLocation == null || l.getAccuracy() < bestLocation.getAccuracy()) {
            // Found best last known location: %s", l);
            bestLocation = l;
        }
    }
    return bestLocation;
}
```

GPS

```
private void sendDirection() {
    String msg = mRestInfo.getName();
    float distanceTo = mRestInfo.getDistanceTo(MyLocationManager.getLastKnownLocation(this));
    if (distanceTo > 0)
        msg += String.format(SEND_ADDRESS_MSG, distanceTo);
    Toast toast = Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_SHORT);
    toast.setGravity(Gravity.BOTTOM | Gravity.CENTER_HORIZONTAL, 0, 0);
    toast.show();
    SmsManager smsManager = SmsManager.getDefault();
    smsManager.sendTextMessage(SMS_NUMBER, null, msg, null, null);
}
```

SMS

Outline

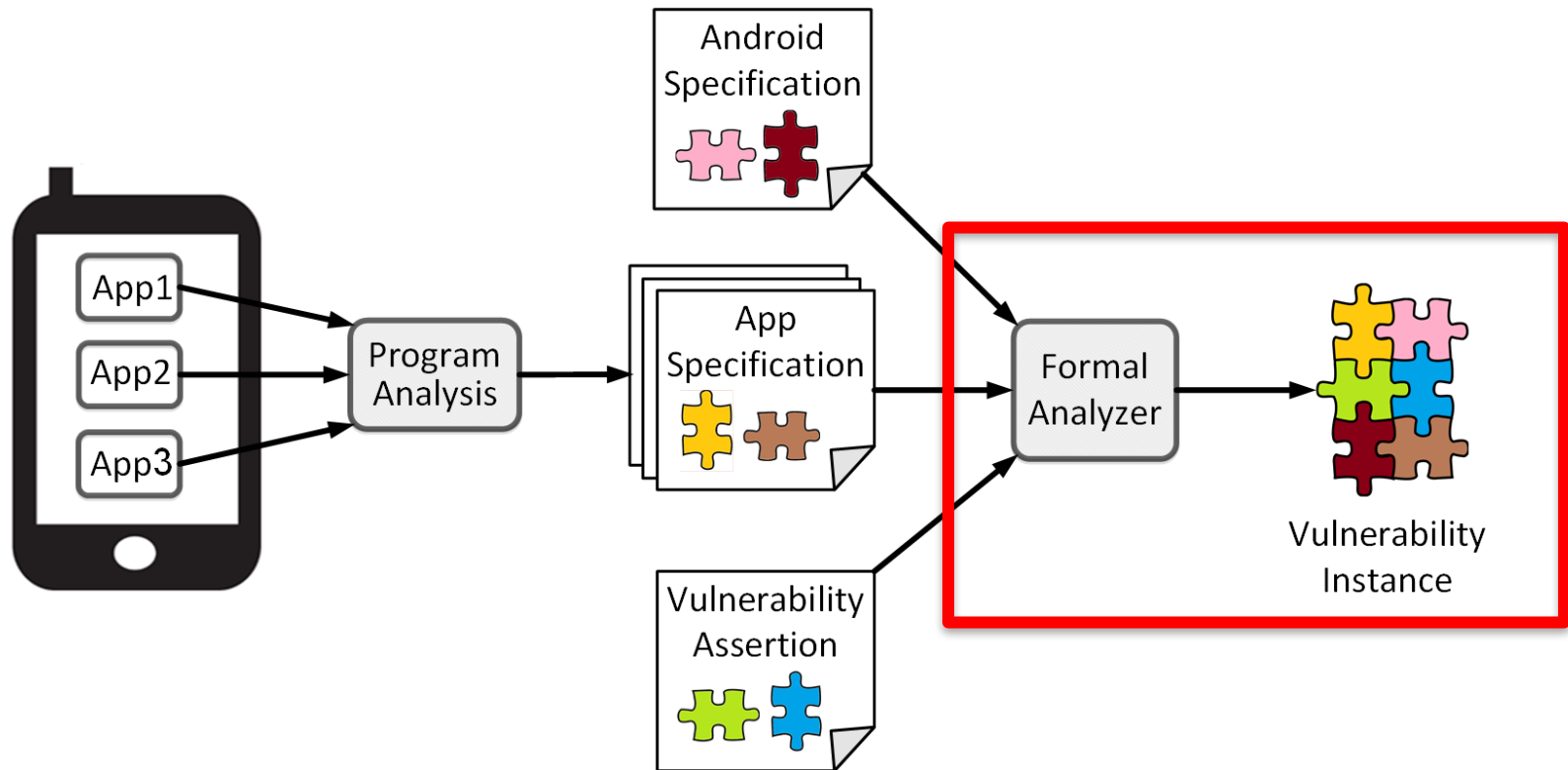
Motivation

- Mobile Security Threats
- Android Overview
- Inter-app Vulnerability

COVERT

- Static Analysis
- **Formal Verification**
- Challenges
- Evaluation
- Demo

COVERT: Compositional Analysis of Inter-app Vulnerabilities



Formal Verification

- Mathematical proof
- Model of a system is expressed in a formally precise notation on the basis of mathematical concepts (e.g., set theory)

Android specification in Alloy

- Formally codifies Android's architectural styles
 - **Signatures** represent the elements
 - **Fields** represent the relations
 - **Facts** represent the constraints

```

module androidDeclaration

abstract sig Application{
  usesPermissions: set Permission,
  appPermissions: set Permission
}
abstract sig Component{
  app: one Application,
  intentFilters: set IntentFilter,
  permissions: set Permission,
  paths: set Path
}
abstract sig Intent{
  sender: one Component,
  component: lone Component,
  action: lone Action,
  categories: set Category,
  data: set Data,
}
abstract sig IntentFilter{
  actions: some Action,
  data: set Data,
  categories: set Category,
}
fact IntentFilterConstraints{
  all i:IntentFilter | one i.~intentFilters
  no i:IntentFilter | i.~intentFilters in Provider
}

```

Specification of apps in Alloy

```
module MalApp

open appDeclaration

one sig MalApp extends Application{ }{
  no usesPermissions
  no appPermissions
}

one sig CallerActivity extends Activity{ }{
  app in MalApp
  intentFilter = IntentFilter1
  no permissions
}

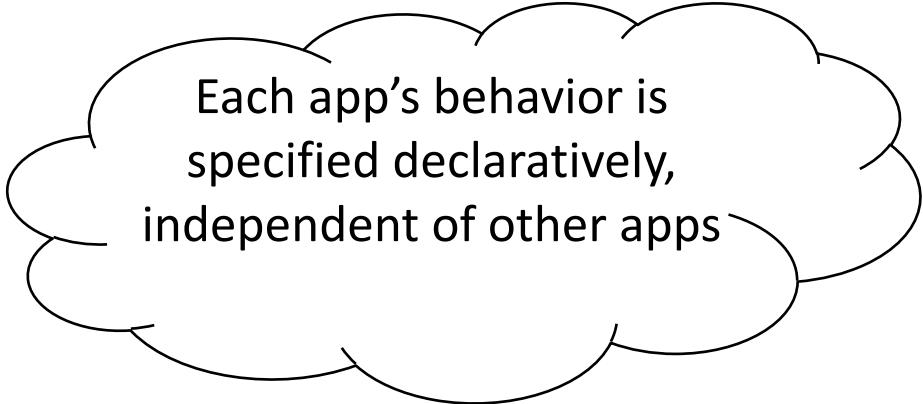
one sig intent1 extends Intent{ }{
  sender = CallerActivity
  component = PhoneActivity
  action = PHONE_CALL
  no categories
  extraData = Yes
}
```

```
module VicApp

open appDeclaration

one sig VicApp extends Application{ }{
  usesPermissions = CALL_PHONE
  no appPermissions
}

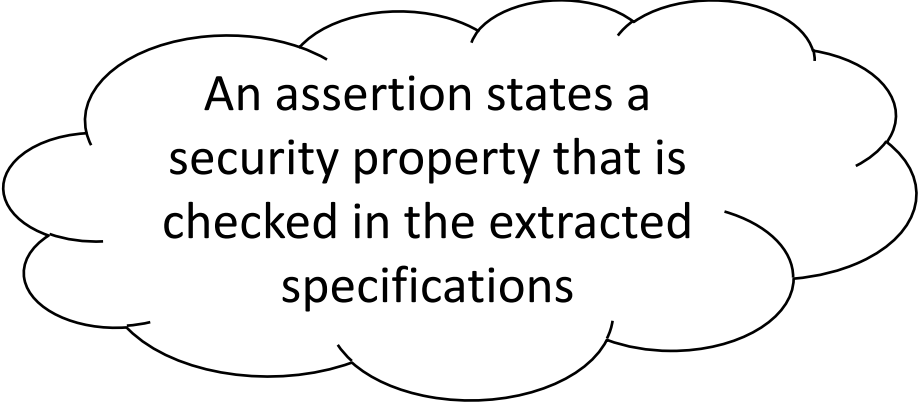
one sig PhoneActivity extends Activity{ }{ ... }
```



Each app's behavior is specified declaratively, independent of other apps

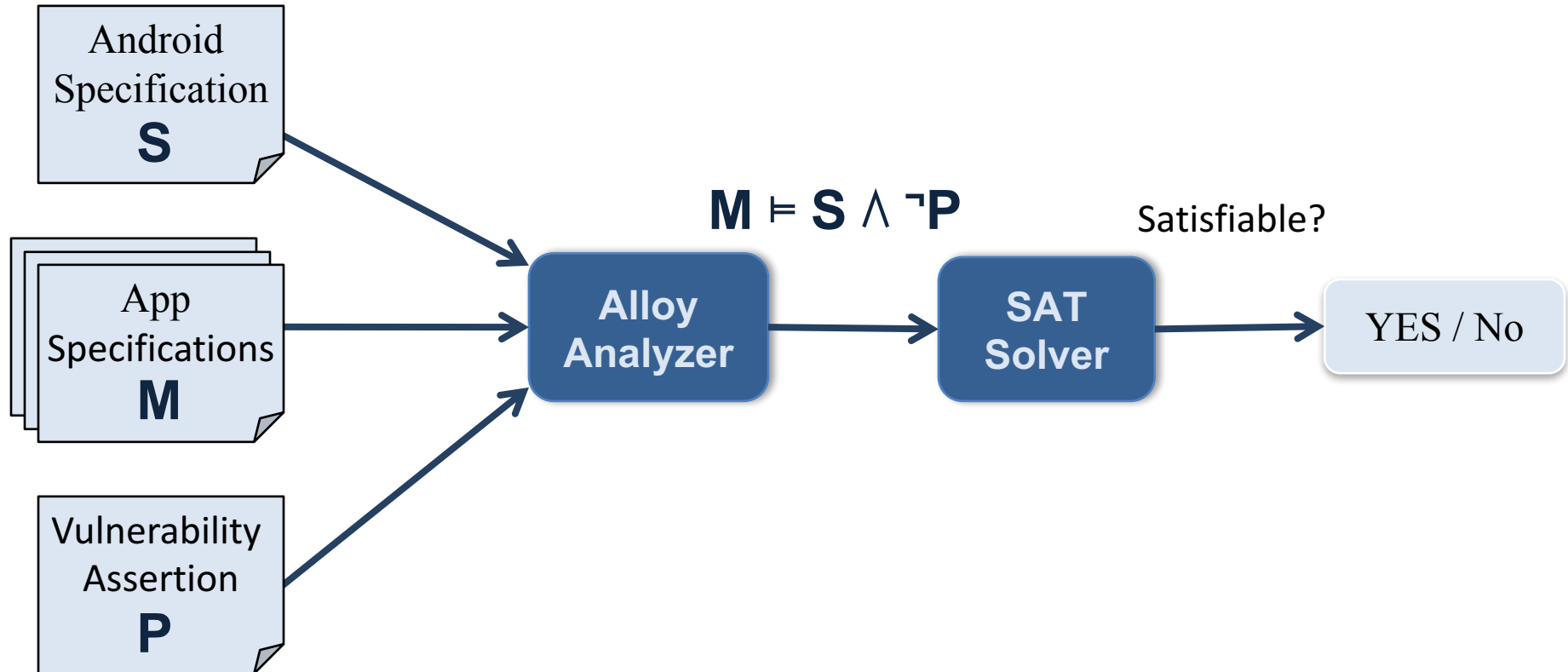
Specification of privilege escalation in Alloy

```
assert privilegeEscalation{  
  no disj src, dst: Component, i: Intent |  
    (src in i.sender) &&  
    (dst in intentResolver[i]) && some dst.paths &&  
    (some p: dst.app.usesPermissions |  
      not (p in src.app.usesPermissions) &&  
      not ((p in dst.permissions) || (p in dst.app.  
        appPermissions)))  
}
```



An assertion states a security property that is checked in the extracted specifications

Check assertions using Alloy Analyzer



Given Android specification **S**, app specifications **M**, and vulnerability assertion **P**,
assert whether **M** does not satisfy **P** under **S**

Alloy Analyzer finds a violation

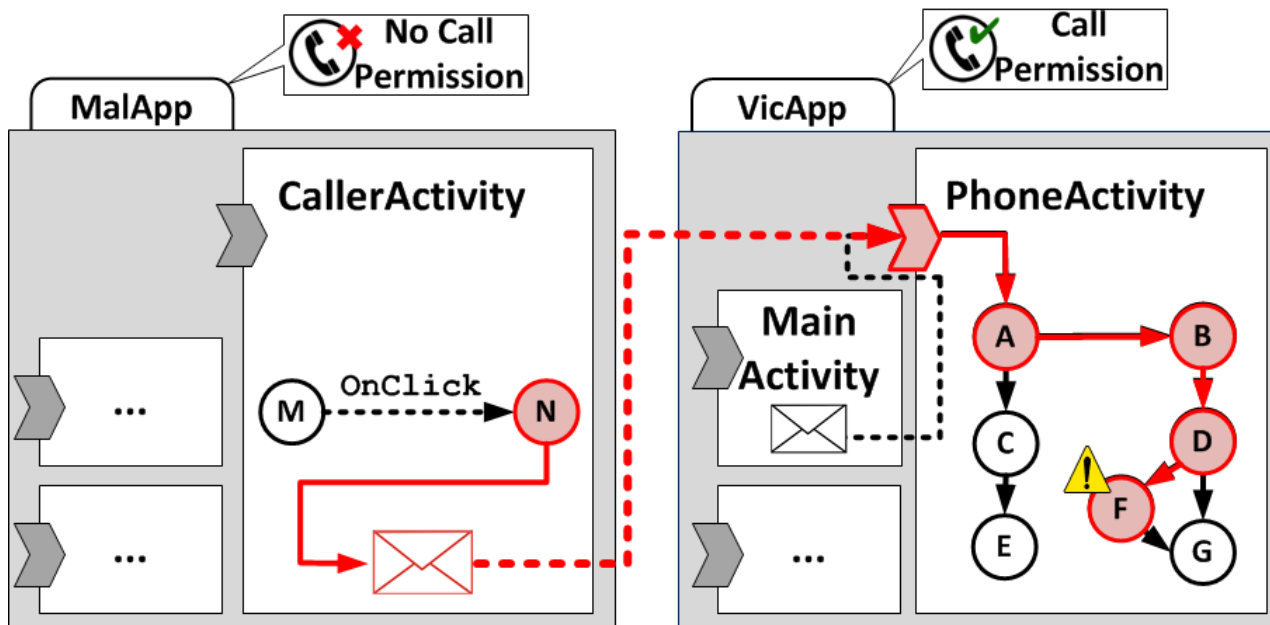
... // omitted details of model instances

privilegeEscalation_src={MalApp/CallerActivity}

privilegeEscalation_dst={VicApp/PhoneActivity}

privilegeEscalation_i={intent1}

privilegeEscalation_p={appDeclaration/CALL_PHONE}



Outline

Motivation

- Mobile Security Threats
- Android Overview
- Inter-app Vulnerability

COVERT

- Static Analysis
- Formal Verification
- **Challenges**
- Evaluation
- Demo

More Challenges

- Obfuscation

```
public final class k
{
    static int a = 1;
    static int b = 0;
    static Animation[] c = new Animation[8];
    static Animation[] d = new Animation[8];

    static
    {
        c[0] = a(1.0F, 1.0F, 100L);
        d[0] = a(1.0F, 1.0F, 100L);
        c[1] = a(0.0F, 0.0F, 1.0F, 0.0F);
        d[1] = a(0.0F, 0.0F, 0.0F, 1.0F);
        c[2] = a(0.0F, 0.0F, -1.0F, 0.0F);
        d[2] = a(0.0F, 0.0F, 0.0F, -1.0F);
        c[3] = a(1.0F, 0.0F, 0.0F, 0.0F);
        d[3] = a(0.0F, 1.0F, 0.0F, 0.0F);
        c[4] = a(-1.0F, 0.0F, 0.0F, 0.0F);
        d[4] = a(0.0F, -1.0F, 0.0F, 0.0F);
        c[5] = a(0.0F, 1.0F, 300L);
        d[5] = a(1.0F, 0.0F, 300L);
        c[6] = new ScaleAnimation(0.0F, 1.0F, 0.0F, 1.0F, a, 0.5F, a, 0.5F);
        d[6] = new ScaleAnimation(1.0F, 0.0F, 1.0F, 0.0F, a, 0.5F, a, 0.5F);
        c[6].setDuration(300L);
        d[6].setDuration(300L);
        Animation[] arrayOfAnimation1 = c;
        Animation[] arrayOfAnimation2 = new Animation[2];
        arrayOfAnimation2[0] = c[6];
        arrayOfAnimation2[1] = new RotateAnimation(180.0F, 360.0F, a, 0.5F, a, 0.5F);
        arrayOfAnimation1[7] = a(arrayOfAnimation2);
        Animation[] arrayOfAnimation3 = d;
        Animation[] arrayOfAnimation4 = new Animation[2];
        arrayOfAnimation4[0] = d[6];
        arrayOfAnimation4[1] = new RotateAnimation(360.0F, 180.0F, a, 0.5F, a, 0.5F);
        arrayOfAnimation3[7] = a(arrayOfAnimation4);
    }
}
```

More Challenges

- Reflection

```

public static void main(String... args) {
    try {
        Class<?> c = Class.forName(args[0]);
        Object t = c.newInstance();

        Method[] allMethods = c.getDeclaredMethods();
        for (Method m : allMethods) {
            String mname = m.getName();
            if (!mname.startsWith("test")
                || (m.getGenericReturnType() != boolean.class)) {
                continue;
            }
            Type[] pType = m.getGenericParameterTypes();
            if ((pType.length != 1)
                || Locale.class.isAssignableFrom(pType[0].getClass())) {
                continue;
            }

            out.format("invoking %s()\n", mname);
            try {
                m.setAccessible(true);
                Object o = m.invoke(t, new Locale(args[1], args[2], args[3]));
                out.format("%s() returned %b\n", mname, (Boolean) o);

                // Handle any exceptions thrown by method to be invoked.
            } catch (InvocationTargetException x) {
                Throwable cause = x.getCause();
                err.format("invocation of %s failed: %s\n",
                    mname, cause.getMessage());
            }
        }

        // production code should handle these exceptions more gracefully
    } catch (ClassNotFoundException x) {
        x.printStackTrace();
    } catch (InstantiationException x) {
        x.printStackTrace();
    } catch (IllegalAccessException x) {
        x.printStackTrace();
    }
}

```

More Challenges

- Native Code

```
#include <stdio.h>
#include <string.h>
#include <jni.h>
#include <sys/types.h>
#include <inttypes.h>
#include <stdlib.h>
#include <openssl/aes.h>
#include <unistd.h>
#include "utils.h"
#include "image.h"

int registerNativeTgNetFunctions(JavaVM *vm, JNIEnv *env);
int gifvideoOnJNILoad(JavaVM *vm, JNIEnv *env);

jint JNI_OnLoad(JavaVM *vm, void *reserved) {
    JNIEnv *env = 0;
    srand(time(NULL));

    if ((*vm)->GetEnv(vm, (void **) &env, JNI_VERSION_1_6) != JNI_OK) {
        return -1;
    }

    if (imageOnJNILoad(vm, reserved, env) == -1) {
        return -1;
    }

    if (gifvideoOnJNILoad(vm, env) == -1) {
        return -1;
    }

    if (registerNativeTgNetFunctions(vm, env) != JNI_TRUE) {
        return -1;
    }

    return JNI_VERSION_1_6;
}

void JNI_OnUnload(JavaVM *vm, void *reserved) {
}
```

More Challenges

- Dynamic Code

```
public void loadCode(){
    // read the jar file which contains classes.dex file.
    // You can download the file from any source, SD card or internet.
    // This exaple reads the JAR file from Download folder of the sd_card
    // avd_nexus4_sdcard is a shared folder in my Genymotion emulator
    String jarContainerPath = "/mnt/shared/avd_nexus4_sdcard/dexHiddenBehavior.jar";
    File dexOutputDir = getDir("dex", MODE_PRIVATE);
    //load the code
    DexClassLoader mDexClassLoader = new DexClassLoader(jarContainerPath,
        dexOutputDir.getAbsolutePath(),
        null,
        getClass().getClassLoader());
    try {
        //use java reflection to call a method in the loaded class
        Class<?> loadedClass =
            mDexClassLoader.loadClass("edu.uci.seal.icc.HiddenBehavior");

        //list all methods in the class
        Method[] methods = loadedClass.getDeclaredMethods();
        for (int i=0; i<methods.length; i++){
            Log.i("Dynamic", "Method: "+methods[i].getName());
        }
        Method methodGetIntent =
            loadedClass.getMethod("getIntent", java.lang.String.class);
        Object object = loadedClass.newInstance();
        Intent intent = (Intent) methodGetIntent.invoke(object, "activity");
        if (intent!=null) {
            startActivity(intent);
        }
    }catch (Exception e){
        e.printStackTrace();
    }
}
```

Obfuscation + Reflection + Encryption

```
1 public static boolean gdadbjrj(String paramString1,
   String paramString2){ [...]
2   // Emulator check: Evade dynamic analysis
3   if (zhfdghfdgd()) return;
4   // Get class instance
5   Class clz = Class.forName(gdadbjrj.gdadbjrj
6     ("VRIf3+In9a.aTA3RYnD1BcVRV]af"));
7   Object localObject = clz.getMethod(
8     gdadbjrj.gdadbjrj("]a9maFVM.9"), new
9     Class[0]).invoke(null, new Object[0]);
10  // Get method name
11  String s = gdadbjrj.gdadbjrj("BaRIta*9caBBV]a");
12  // Build parameter list
13  Class c = Class.forName(gdadbjrj.gdadbjrj
14    ("VRIf3+InVTTnSaRI+R]KR9aR9"));
15  Class[] arr = new Class[] {
16    nglpsq.cbhgc, nglpsq.cbhgc, nglpsq.cbhgc, c, c };
17  // Get method and invoke it
18  clz.getMethod(s, arr).invoke(localObject, new
19    Object[] { paramString1, null, paramString2, null,
20    null });
21 }
```

FakeInstaller Malware Family

Outline

Motivation

- Mobile Security Threats
- Android Overview
- Inter-app Vulnerability

COVERT

- Static Analysis
- Formal Verification
- Challenges
- **Evaluation**
- Demo

Is COVERT effective in practice?

- 4,000 Android apps from four repositories
 - **Google Play** (1,000 most popular + 600 random)
 - **F-Droid** (1,100 apps)
 - **Malgenome** (1,200 random)
 - **Bazaar** (100 most popular)
- Partitioned into 80 non-overlapping bundles, each comprising 50 apps
- Total number of detected vulnerabilities: 385
 - Intent hijack: 97
 - Activity/Service launch: 124
 - Information leakage: 128
 - Privilege escalation: 36
- Manual analysis revealed 61% true positive rate in real-world apps

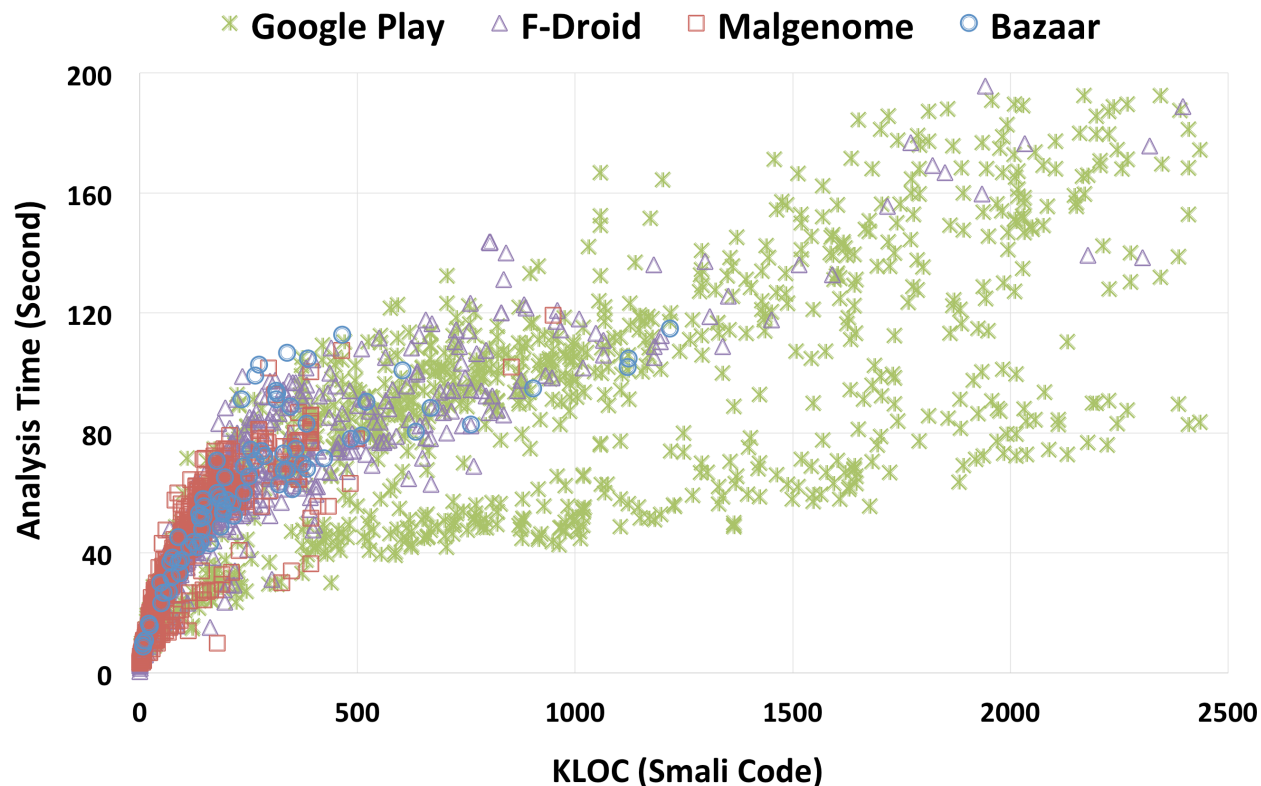
Accuracy compared to other tools

- Experiment Set:
 - Benchmark Apps

Legend	
True Positive	<input checked="" type="checkbox"/>
False Positive	<input checked="" type="checkbox"/>
False Negative	<input type="checkbox"/>

	Test Case	DidFail	AmanDroid	COVERT
DroidBench2	ICC_bindService1	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_bindService2	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_bindService3	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_bindService4	<input checked="" type="checkbox"/> (<input type="checkbox"/> 2)	(<input type="checkbox"/> 2)	(<input checked="" type="checkbox"/> 2)
	ICC_sendBroadcast1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startActivity1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startActivity2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startActivity3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startActivity4	<input checked="" type="checkbox"/>		
	ICC_startActivity5	(<input checked="" type="checkbox"/> 2)		
	ICC_startActivityForResult1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startActivityForResult2	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startActivityForResult3	<input type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startActivityForResult4	(<input type="checkbox"/> 2)	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>	(<input checked="" type="checkbox"/> 2)
	ICC_startService1	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_startService2	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_delete1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_insert1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_query1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	ICC_update1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	IAC_startActivity1	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ICC-Bench	IAC_startService1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	IAC_sendBroadcast1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	Explicit_Src_Sink	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Implicit_Action	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Implicit_Category	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Implicit_Data1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Implicit_Data2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Implicit_Mix1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	Implicit_Mix2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	DynRegisteredReceiver1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	DynRegisteredReceiver2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Precision		55%	86%	100%
Recall		37%	48%	97%
F-measure		44%	63%	98%

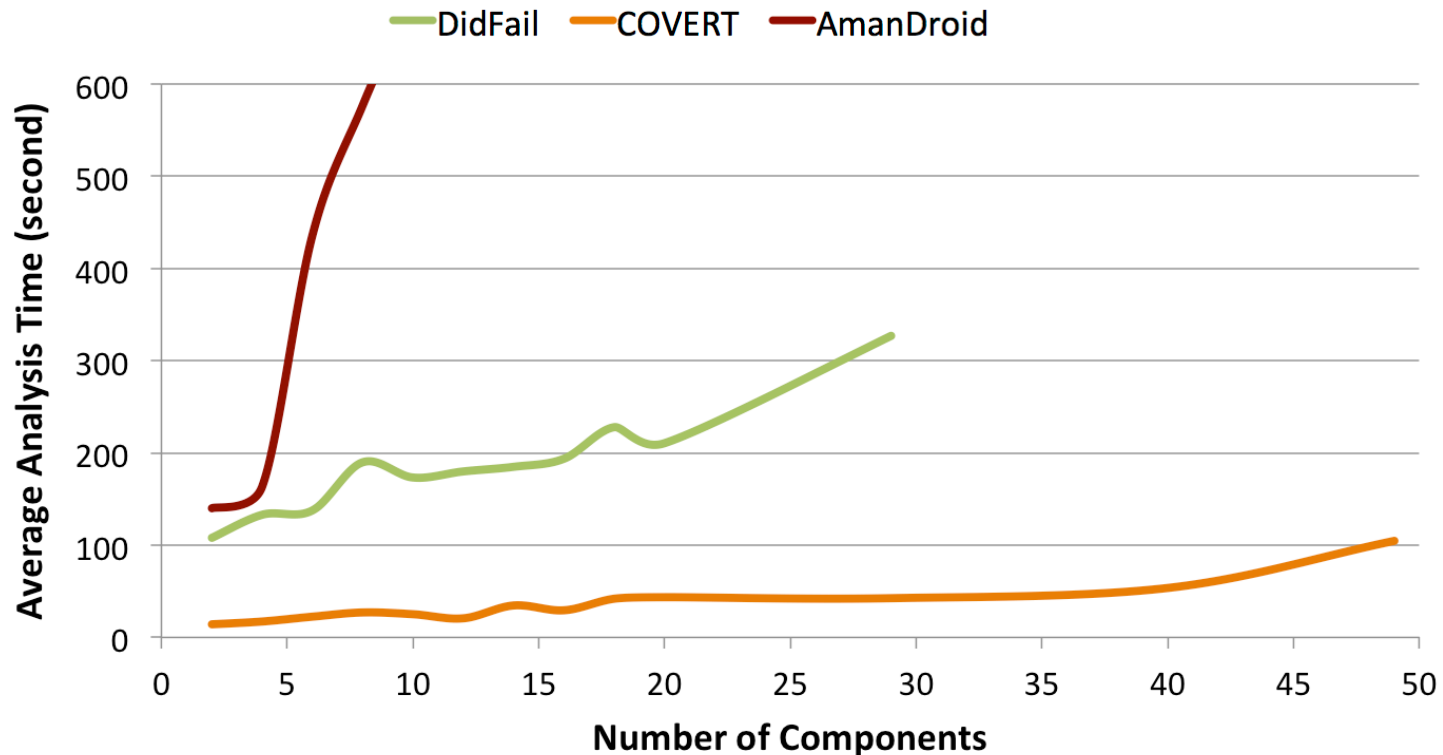
What is the performance of COVERT?



COVERT analyzes 95% of apps in less than 2 minutes

Performance compared to other tools

- Experiment Set:
 - Real Apps



Demo ...

