INVESTIGATING STATE-OF-THE-ART ANDROID MALWARE WITH CODEINSPECT

Fraunhofer
SIT

Partner in CRISP
Center for Research in Security and Privacy

AGENDA

- Android Malware: Binary-Only Investigation
- Dissecting Malware with CodeInspect
- Advanced Static Analysis
- Conclusions

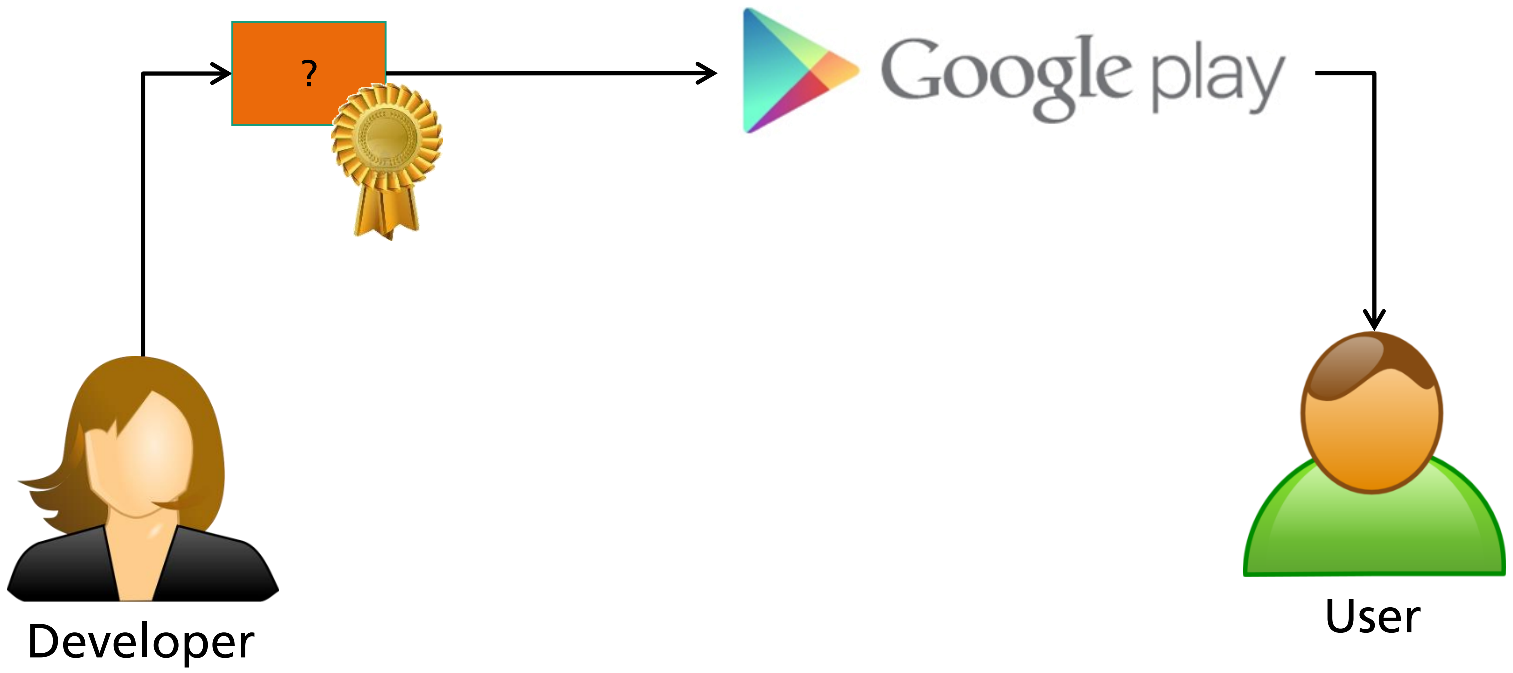
Who Am I?

- 4th year PhD Student at TU Darmstadt
- Researcher at Fraunhofer SIT

- Research interests:
  - Static analysis
  - IT security

- Community service
  - Reviewer for conferences & journals
  - Maintainer of Soot and FlowDroid

The Android Ecosystem (1)



The Android Ecosystem (2)

amazon underground

Google play

安卓市场
apk.hiapk.com
— 百度旗下品牌 —

Yandex Store

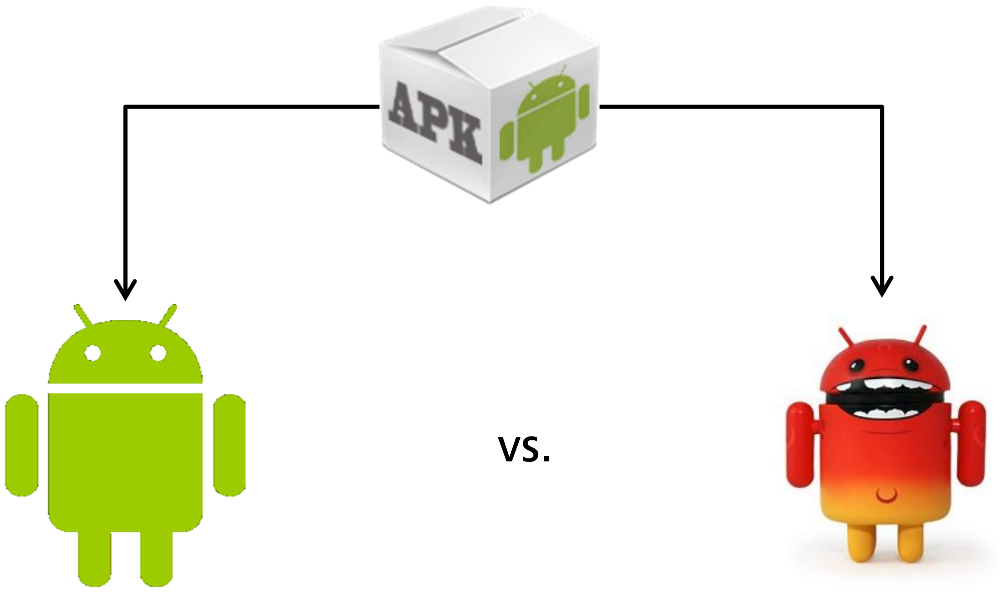
百度手机助手
人 气 安 卓 应 用 商 店

应用宝

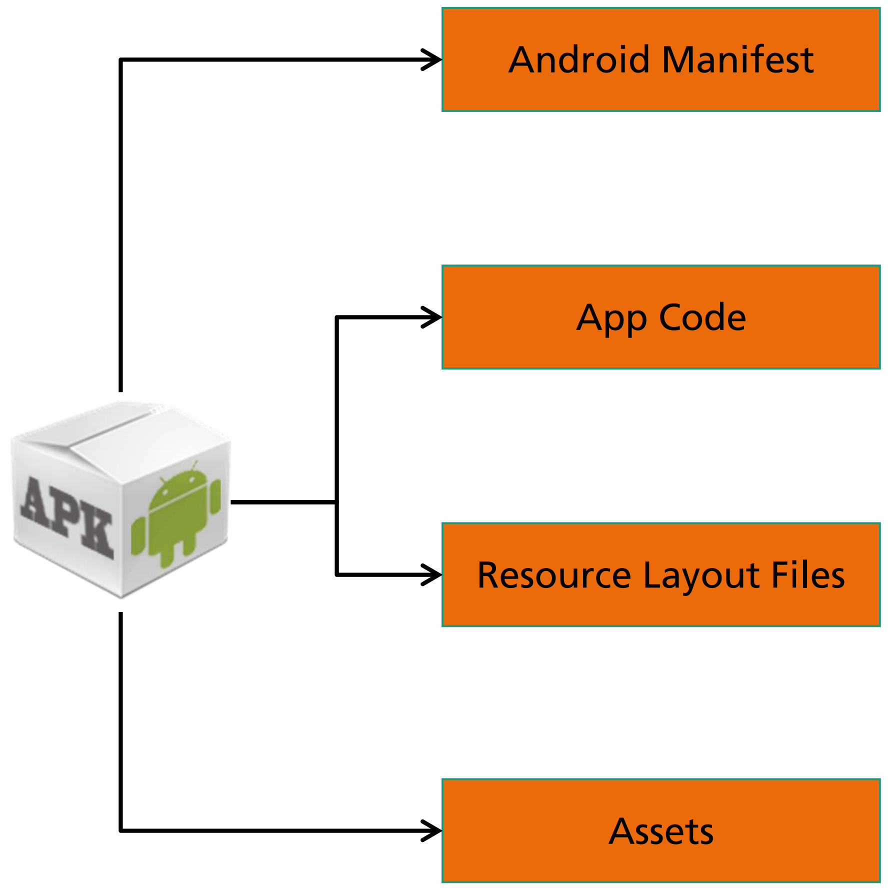
小米应用商店
APP.MI.COM

华为应用市场
appstore.huawei.com | 放心下，畅快玩

The Android Ecosystem (3)



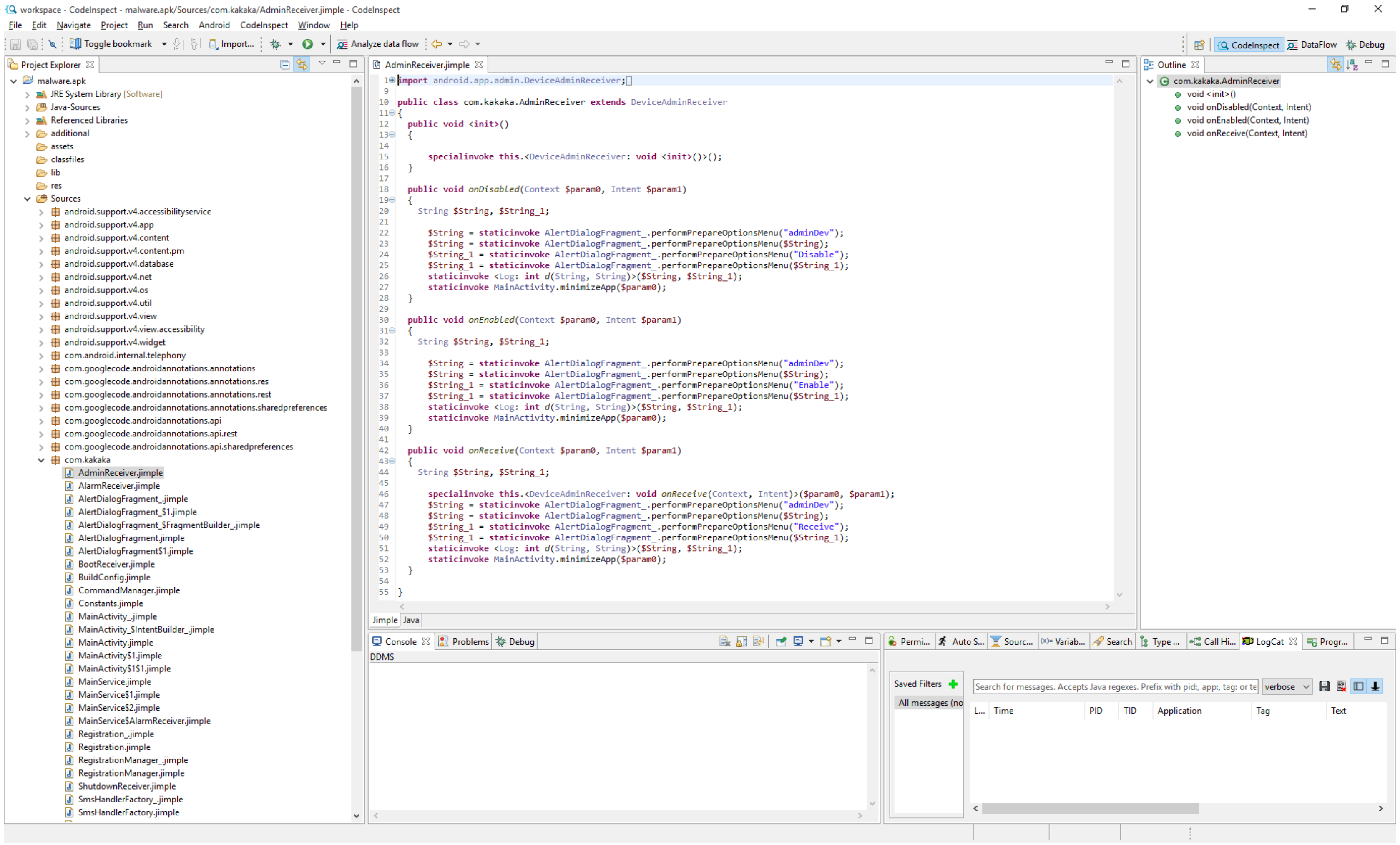
Android Black-Box App Analysis



CodeInspect At A Glance (1)

- Based on Eclipse RCP
- Work as you would on source code in Eclipse
  - Navigate through the code
  - Add, change, and remove code
  - Inject arbitrary Java code
  - Start and debug your app
  - Inspect and change runtime values

CodeInspect At A Glance (2)



```
malware.apk
  JRE System Library [Software]
  Java-Sources
  Referenced Libraries
  additional
  assets
  classfiles
  lib
  res
  Sources
    android.support.v4.accessibilityservice
    android.support.v4.app
    android.support.v4.content
    android.support.v4.content.pm
    android.support.v4.database
    android.support.v4.net
    android.support.v4.os
    android.support.v4.util
    android.support.v4.view
    android.support.v4.view.accessibility
    android.support.v4.widget
    com.android.internal.telephony
    com.googlecode.androidannotations.annotations
    com.googlecode.androidannotations.annotations.res
    com.googlecode.androidannotations.annotations.rest
    com.googlecode.androidannotations.annotations.sharedpreferences
    com.googlecode.androidannotations.api
    com.googlecode.androidannotations.api.rest
    com.googlecode.androidannotations.api.sharedpreferences
    com.kakaka
      AdminReceiver.jimple
      AlarmReceiver.jimple
      AlertDialogFragment_.jimple
      AlertDialogFragment_$1.jimple
      AlertDialogFragment_$FragmentBuilder_.jimple
      AlertDialogFragment.jimple
      AlertDialogFragment$1.jimple
      BootReceiver.jimple
      BuildConfig.jimple
      CommandManager.jimple
      Constants.jimple
      MainActivity_.jimple
      MainActivity_$IntentBuilder_.jimple
      MainActivity.jimple
      MainActivity$1.jimple
      MainActivity$1$1.jimple
      MainService.jimple
      MainService$1.jimple
      MainService$2.jimple
      MainService$AlarmReceiver.jimple
      Registration_.jimple
      Registration.jimple
      RegistrationManager_.jimple
      RegistrationManager.jimple
      ShutdownReceiver.jimple
      SmsHandlerFactory_.jimple
      SmsHandlerFactory.jimple
```

```
import android.app.admin.DeviceAdminReceiver;

public class com.kakaka.AdminReceiver extends DeviceAdminReceiver
{
  public void <init>()
  {

      specialinvoke this.<DeviceAdminReceiver: void <init>()>();
  }

  public void onDisabled(Context $param0, Intent $param1)
  {
    String $String, $String_1;

      $String = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu("adminDev");
      $String = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu($String);
      $String_1 = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu("Disable");
      $String_1 = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu($String_1);
      staticinvoke <Log: int d(String, String)>($String, $String_1);
      staticinvoke MainActivity.minimizeApp($param0);
  }

  public void onEnabled(Context $param0, Intent $param1)
  {
    String $String, $String_1;

      $String = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu("adminDev");
      $String = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu($String);
      $String_1 = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu("Enable");
      $String_1 = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu($String_1);
      staticinvoke <Log: int d(String, String)>($String, $String_1);
      staticinvoke MainActivity.minimizeApp($param0);
  }

  public void onReceive(Context $param0, Intent $param1)
  {
    String $String, $String_1;

      specialinvoke this.<DeviceAdminReceiver: void onReceive(Context, Intent)>($param0, $param1);
      $String = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu("adminDev");
      $String = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu($String);
      $String_1 = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu("Receive");
      $String_1 = staticinvoke AlertDialogFragment_.performPrepareOptionsMenu($String_1);
      staticinvoke <Log: int d(String, String)>($String, $String_1);
      staticinvoke MainActivity.minimizeApp($param0);
  }

}
```

```
com.kakaka.AdminReceiver
  void <init>()
  void onDisabled(Context, Intent)
  void onEnabled(Context, Intent)
  void onReceive(Context, Intent)
```

Partner in

CodeInspect At A Glance (3)

- Sophisticated Static and Dynamic Analysis
  - Permission Use Analysis
  - Sensitive API Call Detection
  - Data Flow Tracking
  - Runtime Code Injection
  - App Communication Analysis

The Jimple IR

Method Declaration

```
public void foo() {
```

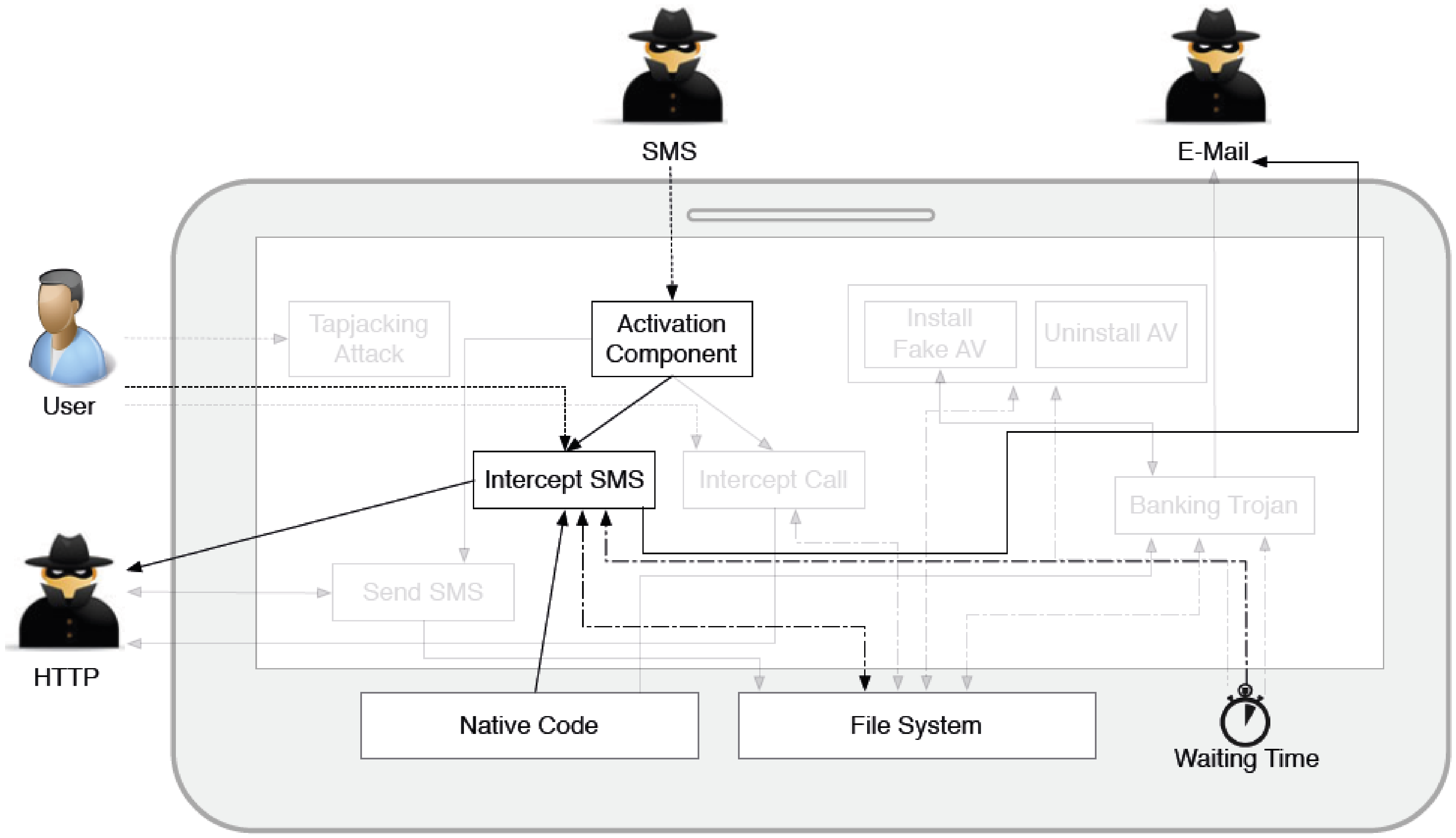
Variable Declarations

```
    byte[] $arrbyte;
    java.io.FileOutputStream $FileOutputStream;
    …

    specialinvoke this.<android.app.Service: void onCreate()>();

    $File = new java.io.File;
    specialinvoke $File.<java.io.File: void <init>(java.lang.String)>("/sdcard/test.apk");
    specialinvoke $FileOutputStream.<java.io.FileOutputStream: void <init>(java.io.File)>($File);

    $arrbyte = newarray (byte)[1024];
    $int = virtualinvoke $InputStream.<java.io.InputStrea
    …
```

Implementation

Live Demo (1)

Live Demo (2)


SMS
E-Mail
User
Tapjacking Attack
Activation Component
Install Fake AV
Uninstall AV
Intercept SMS
Intercept Call
Banking Trojan
HTTP
Send SMS
Native Code
File System
Waiting Time

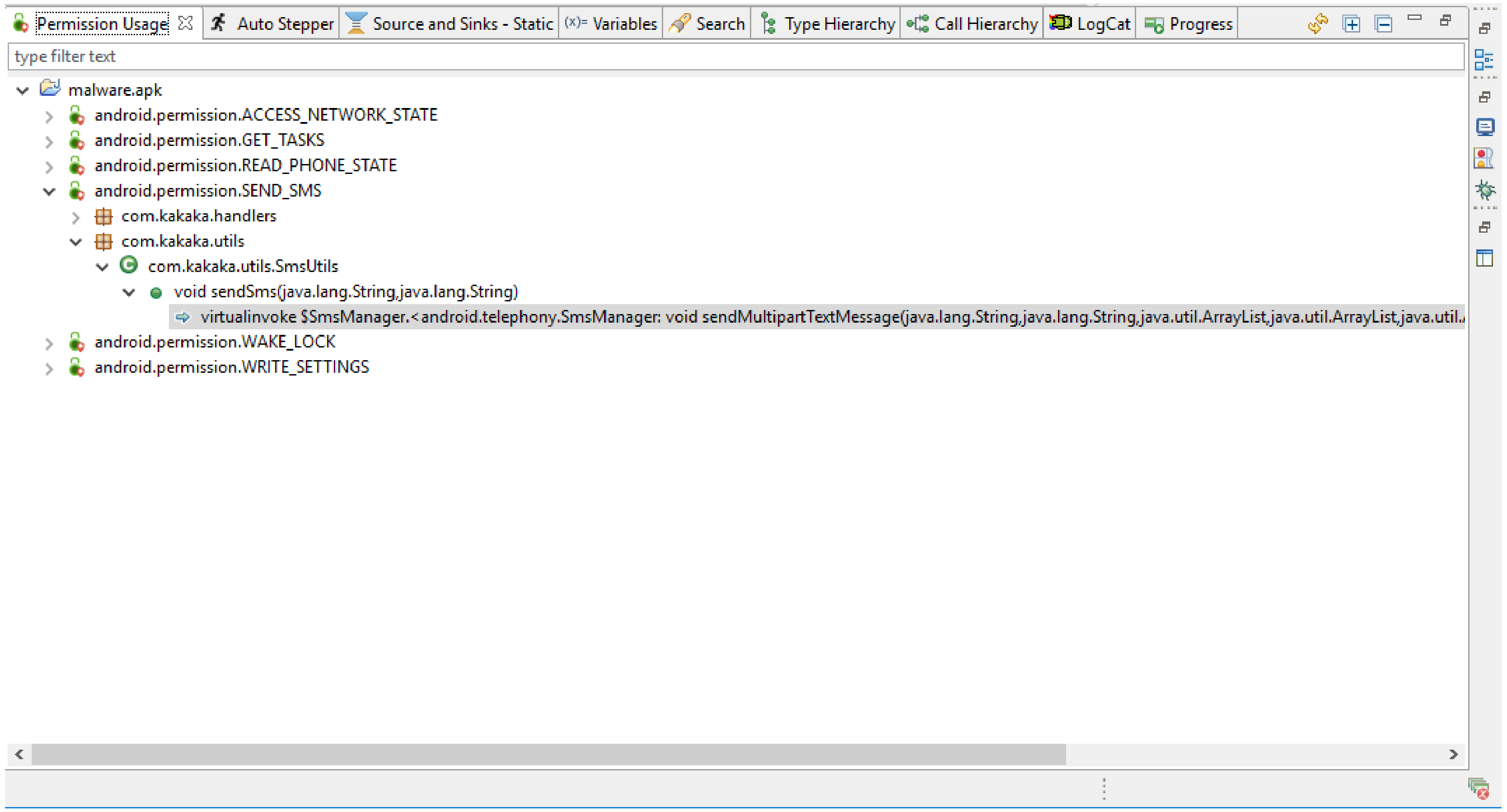
Live Demo (3)

Live Demo Wrap-Up

1. Find interesting starting points
   - External guidance (network sniff, etc.)
   - Text search
   - Manifest analysis: main activity, application class, etc.
   - Permission uses

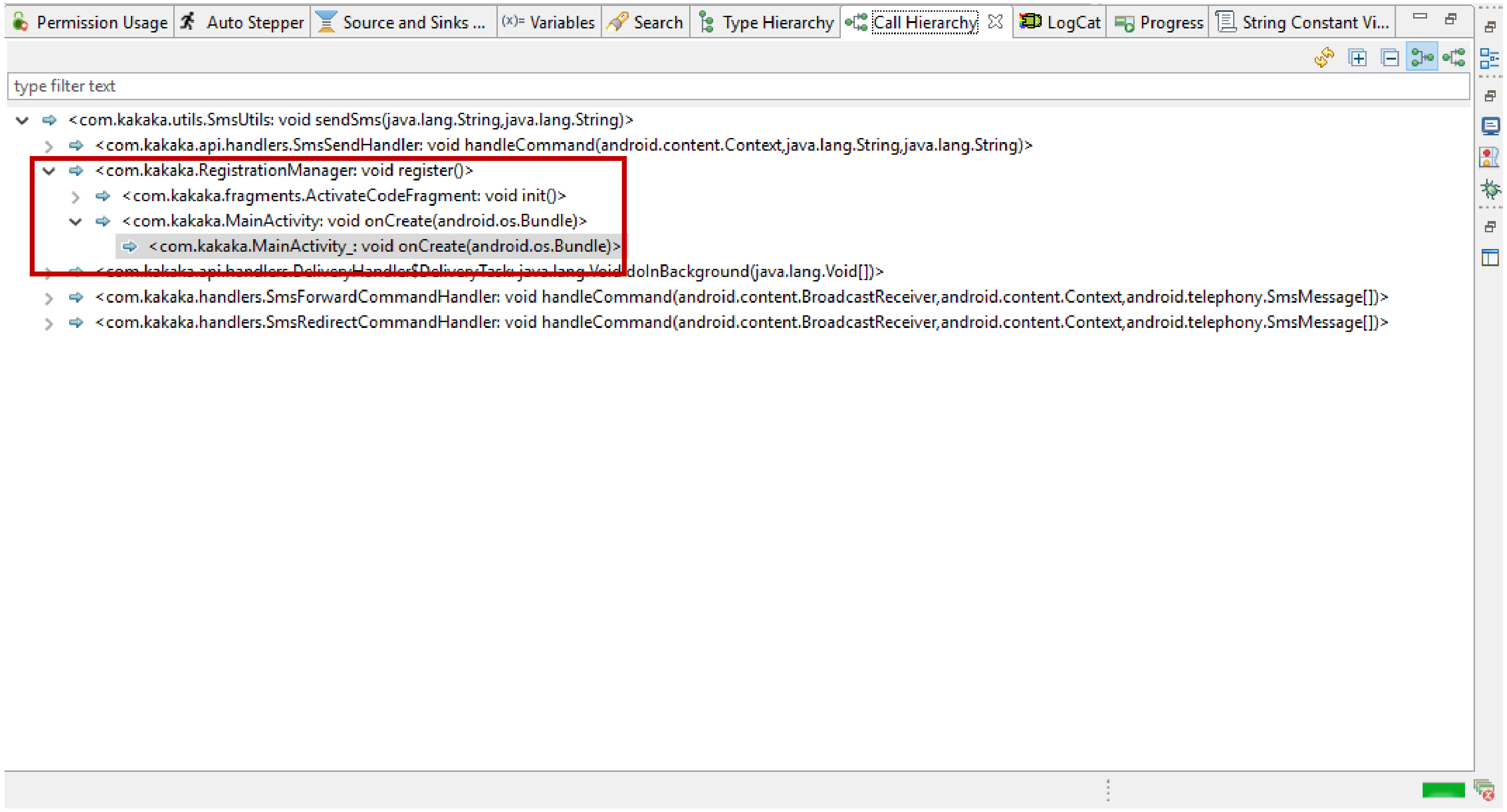
2. Debug the app for the details
   - Circumvent environment checks (e.g., emulator)
   - Step over reflective calls for free
   - URLs, IP addresses, e-mail addresses, telephone numbers, etc.

Advanced Static Analysis: Permission Usage



```
malware.apk
  android.permission.ACCESS_NETWORK_STATE
  android.permission.GET_TASKS
  android.permission.READ_PHONE_STATE
  android.permission.SEND_SMS
    com.kakaka.handlers
    com.kakaka.utils
      com.kakaka.utils.SmsUtils
        void sendSms(java.lang.String,java.lang.String)
          virtualinvoke $SmsManager.<android.telephony.SmsManager: void sendMultipartTextMessage(java.lang.String,java.lang.String,java.util.ArrayList,java.util.ArrayList,java.util.
  android.permission.WAKE_LOCK
  android.permission.WRITE_SETTINGS
```

Where is this called?

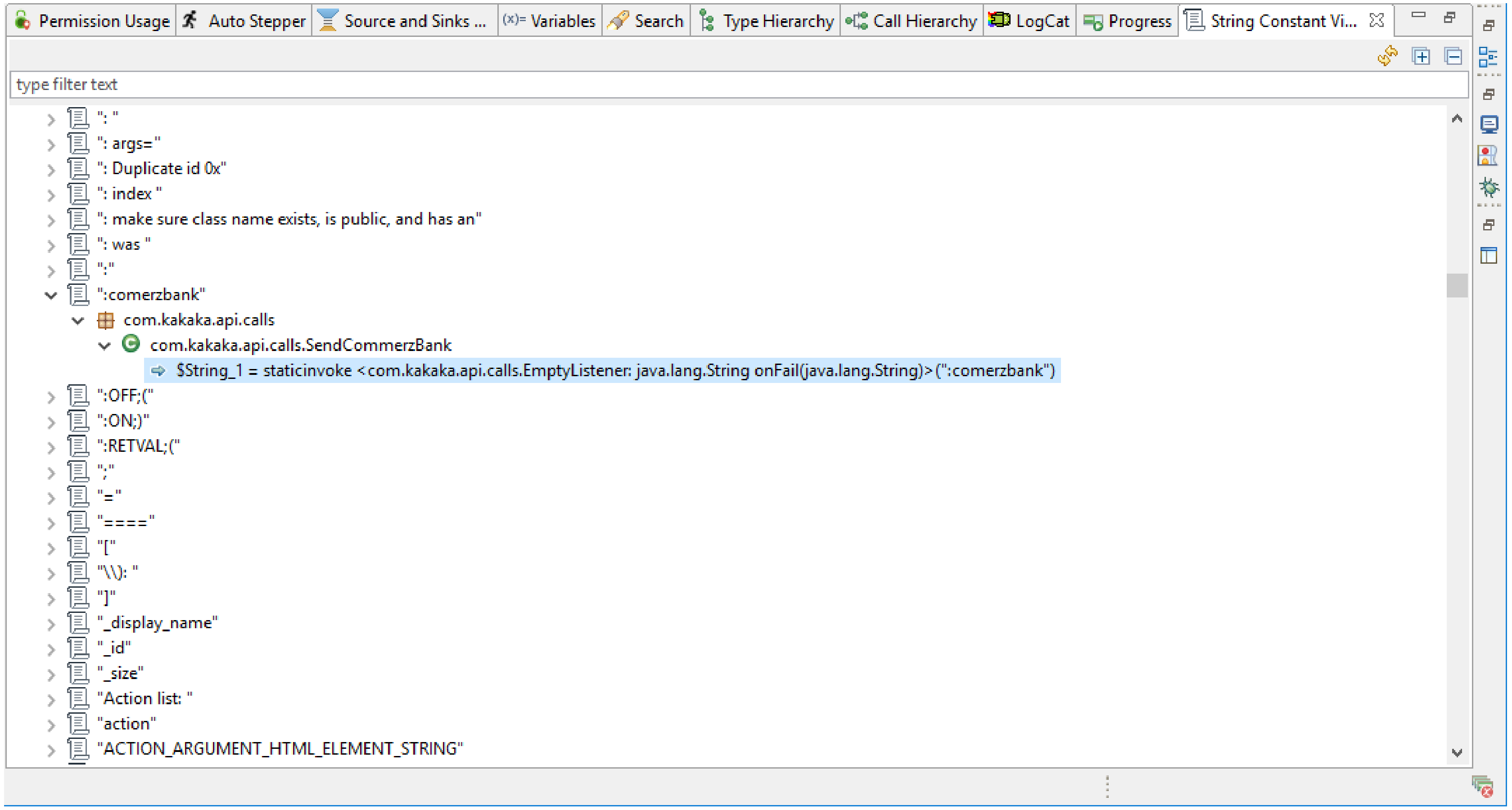


```
<com.kakaka.utils.SmsUtils: void sendSms(java.lang.String,java.lang.String)>
  <com.kakaka.api.handlers.SmsSendHandler: void handleCommand(android.content.Context,java.lang.String,java.lang.String)>
  <com.kakaka.RegistrationManager: void register()>
    <com.kakaka.fragments.ActivateCodeFragment: void init()>
    <com.kakaka.MainActivity: void onCreate(android.os.Bundle)>
      <com.kakaka.MainActivity_: void onCreate(android.os.Bundle)>
  <com.kakaka.api.handlers.DeliveryHandler$DeliveryTask: java.lang.Void doInBackground(java.lang.Void[])>
  <com.kakaka.handlers.SmsForwardCommandHandler: void handleCommand(android.content.BroadcastReceiver,android.content.Context,android.telephony.SmsMessage[])>
  <com.kakaka.handlers.SmsRedirectCommandHandler: void handleCommand(android.content.BroadcastReceiver,android.content.Context,android.telephony.SmsMessage[])>
```

Investigating the SMS Message

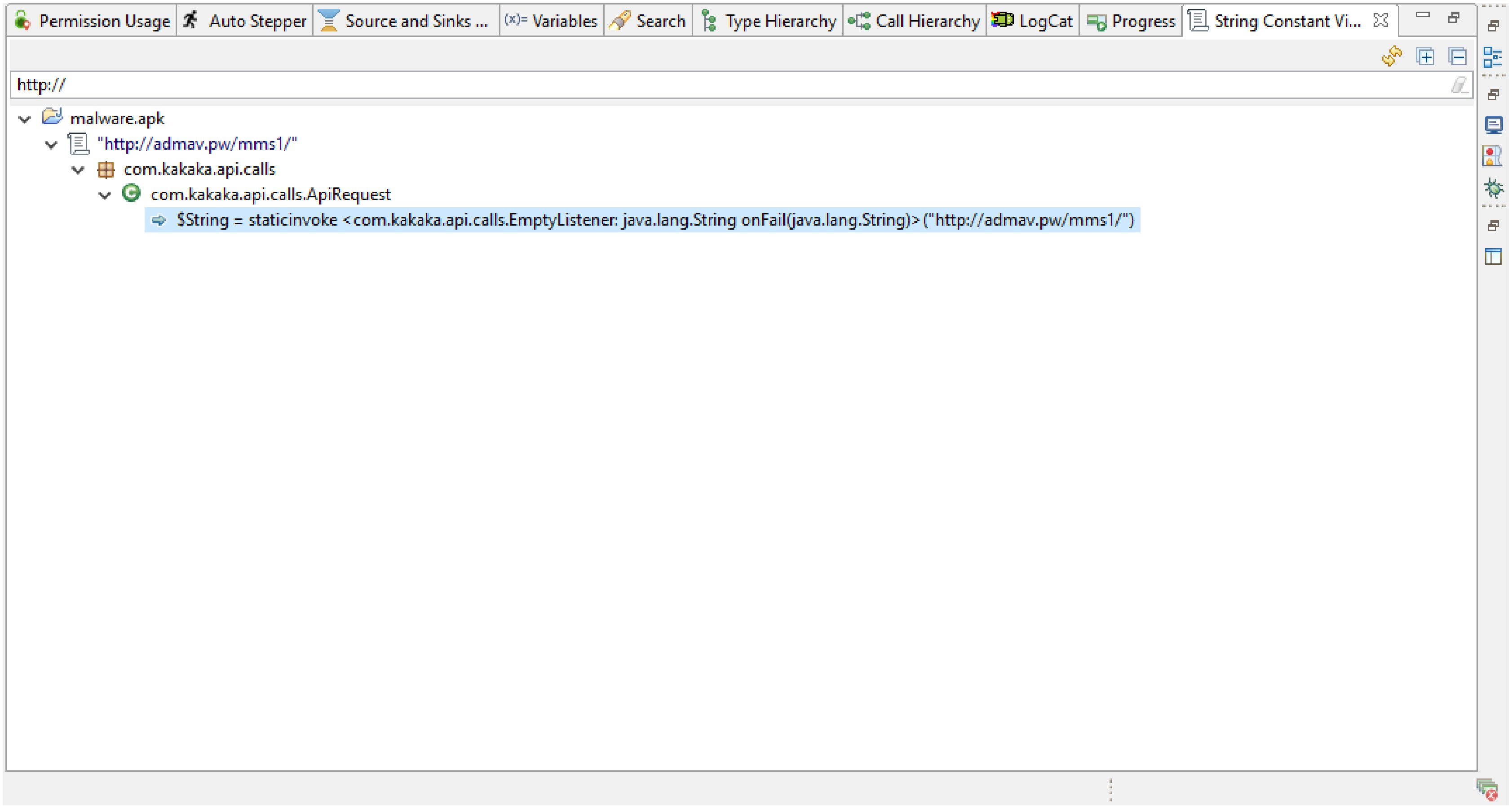
- Set breakpoints
  - in onCreate()
  - in sendSms()
- Look at the path in between
  - Conditions?
  - Remote triggers?
  - Runtime values?
- Emulate necessary events
  - Incoming SMS message, location change, etc.

Advanced Static Analysis: String Constants (1)



```
": "
": args="
": Duplicate id 0x"
": index "
": make sure class name exists, is public, and has an"
": was "
":"
":comerzbank"
    com.kakaka.api.calls
        com.kakaka.api.calls.SendCommerzBank
            $String_1 = staticinvoke <com.kakaka.api.calls.EmptyListener: java.lang.String onFail(java.lang.String)>(":comerzbank")
":OFF;("
":ON;)"
":RETVAL;("
";"
"="
"===="
"["
"\\): "
"]"
"_display_name"
"_id"
"_size"
"Action list: "
"action"
"ACTION_ARGUMENT_HTML_ELEMENT_STRING"
```

Advanced Static Analysis: String Constants (2)

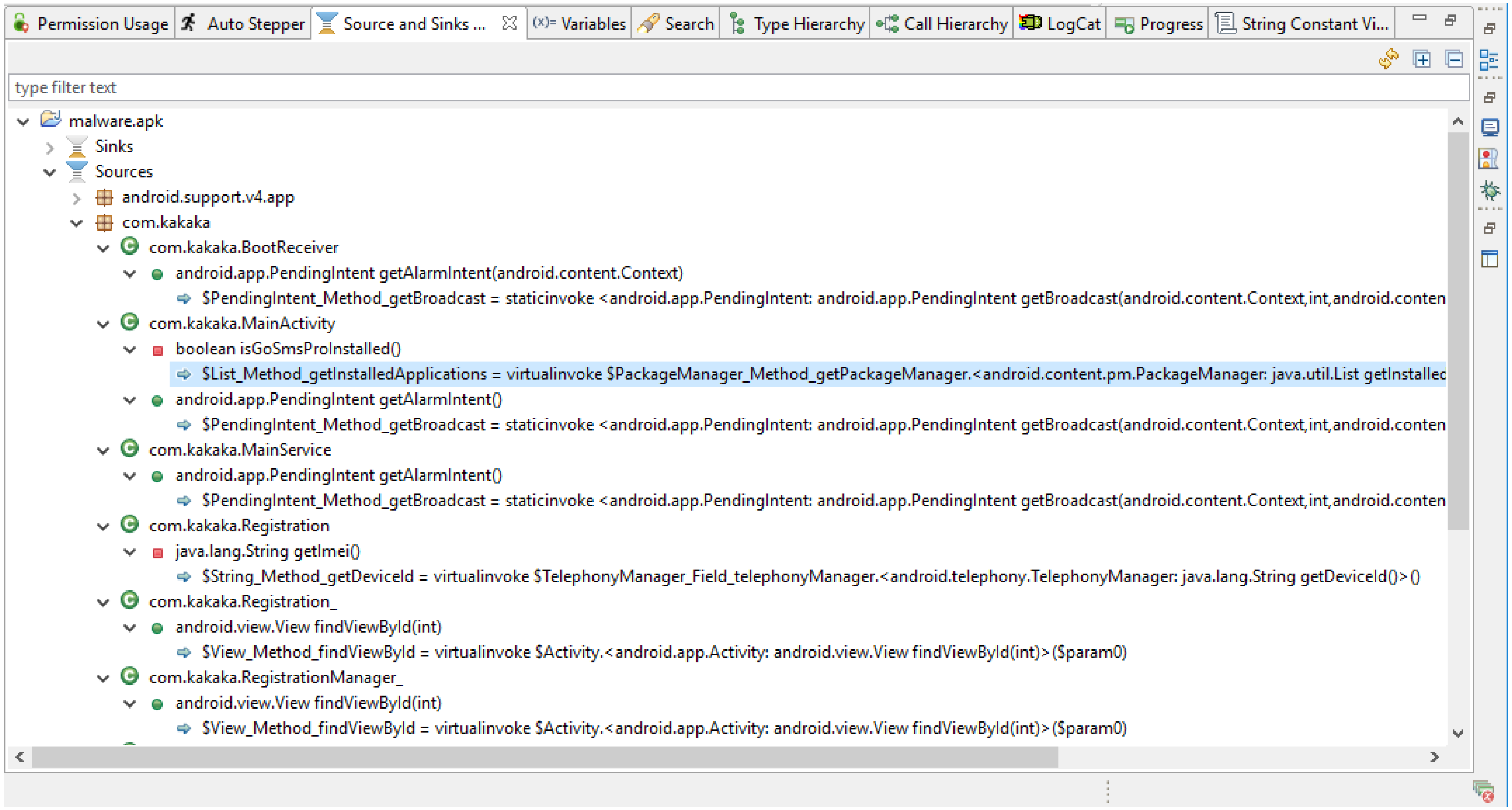


```
http://
malware.apk
  "http://admav.pw/mms1/"
    com.kakaka.api.calls
      com.kakaka.api.calls.ApiRequest
        $String = staticinvoke <com.kakaka.api.calls.EmptyListener: java.lang.String onFail(java.lang.String)>("http://admav.pw/mms1/")
```

Advanced Static Analysis: String Constants (3)

- Look for common patterns
  - http:// and https:// connections
  - Telephone Numbers
  - File paths (/sdcard/)
- Case-specific patterns
  - Bank names
  - Country names
  - Strings from SMS messages or e-mails

Advanced Static Analysis: Sensitive API Calls



```
malware.apk
  Sinks
  Sources
    android.support.v4.app
    com.kakaka
      com.kakaka.BootReceiver
        android.app.PendingIntent getAlarmIntent(android.content.Context)
          $PendingIntent_Method_getBroadcast = staticinvoke <android.app.PendingIntent: android.app.PendingIntent getBroadcast(android.content.Context,int,android.conten
      com.kakaka.MainActivity
        boolean isGoSmsProInstalled()
          $List_Method_getInstalledApplications = virtualinvoke $PackageManager_Method_getPackageManager.<android.content.pm.PackageManager: java.util.List getInstalled
        android.app.PendingIntent getAlarmIntent()
          $PendingIntent_Method_getBroadcast = staticinvoke <android.app.PendingIntent: android.app.PendingIntent getBroadcast(android.content.Context,int,android.conten
      com.kakaka.MainService
        android.app.PendingIntent getAlarmIntent()
          $PendingIntent_Method_getBroadcast = staticinvoke <android.app.PendingIntent: android.app.PendingIntent getBroadcast(android.content.Context,int,android.conten
      com.kakaka.Registration
        java.lang.String getImei()
          $String_Method_getDeviceId = virtualinvoke $TelephonyManager_Field_telephonyManager.<android.telephony.TelephonyManager: java.lang.String getDeviceId()> ()
      com.kakaka.Registration_
        android.view.View findViewById(int)
          $View_Method_findViewById = virtualinvoke $Activity.<android.app.Activity: android.view.View findViewById(int)> ($param0)
      com.kakaka.RegistrationManager_
        android.view.View findViewById(int)
          $View_Method_findViewById = virtualinvoke $Activity.<android.app.Activity: android.view.View findViewById(int)> ($param0)
```

Partner in CRISP Center for Research in Security and Privacy

Fraunhofer SIT

Conclusions

- Unified environment for Android app analysis

- Get an overview
  - Sensitive API calls
  - Permission use
  - Suspicious strings

- Go deep with the debugger
  - Avoid environment checks
  - Extract runtime values

Code Inspect

www.codeinspect.de

Free Demo Version Available!