Static and Dynamic Software Security Analysis:

Analysing the KoopaApp

Presented by: Fares KECHID and Mashal ZAINAB
Presented on: 14th December, 2023.

UNIVERSITÉ DU LUXEMBOURG

Table of contents

01 Tools investigation

02 Instrument the app

03 Location Source and Shortest Path Identification

04 Data Leak Detection

05 LLM Experiment

06 Conclusion

01

Tools investigation

The chosen tools:

Java Soot Framework

A framework for analyzing and transforming Java and Android applications.

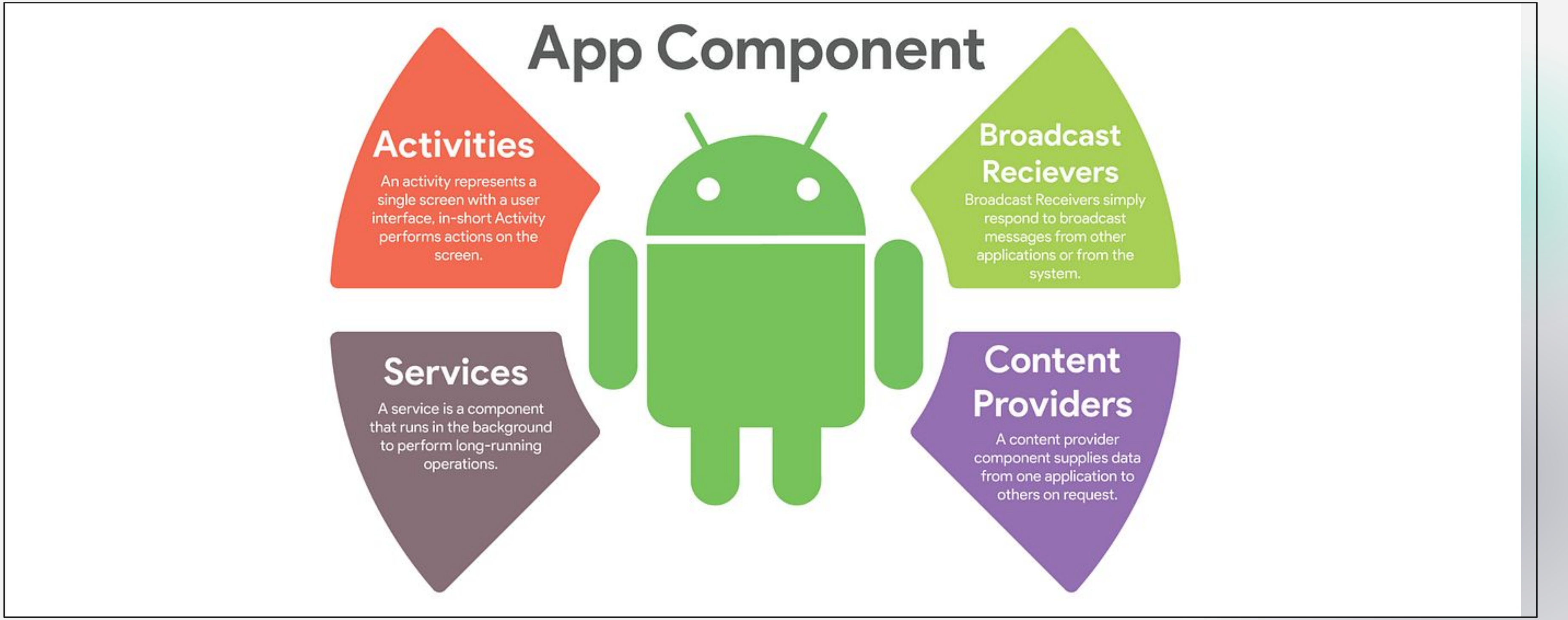
FlowDroid

a context-, flow-, field-, object-sensitive and lifecycle-aware static taint analysis tool for Android applications

Why?

- Cover all the features Needed.
- One of the best tools.
- Compatible with each other.
- Prior experience with Soot

Android app components and their lifecycles.



App Component

Activities

An activity represents a single screen with a user interface, in-short Activity performs actions on the screen.

Broadcast Recievers

Broadcast Receivers simply respond to broadcast messages from other applications or from the system.

Services

A service is a component that runs in the background to perform long-running operations.

Content Providers

A content provider component supplies data from one application to others on request.

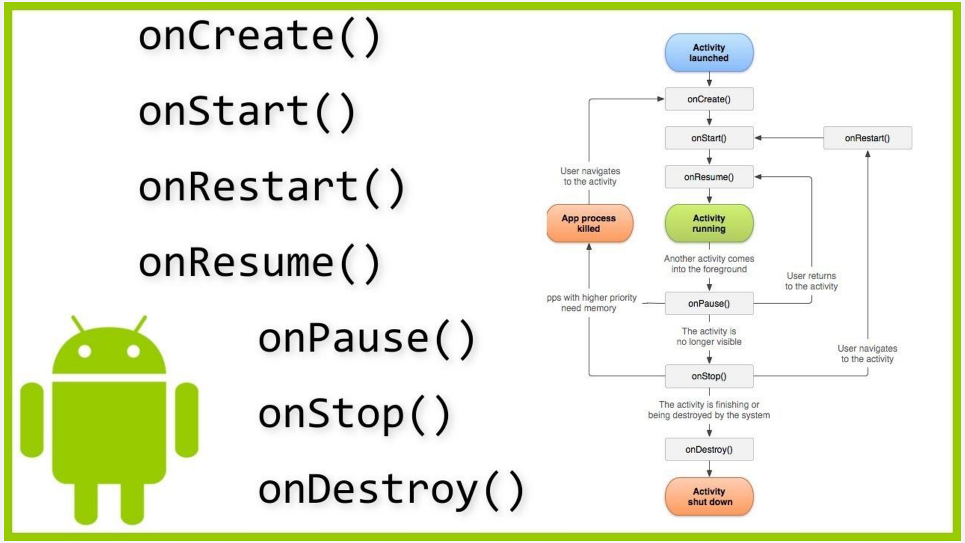
Additional Android app components

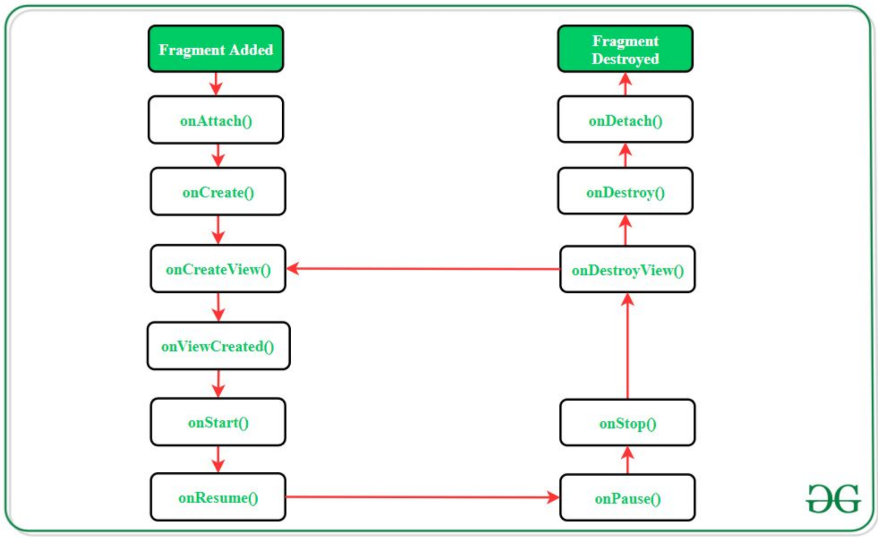
Fragments	Represents a portion of the user interface in an Activity.
Views UI	Elements that are drawn on-screen including buttons, lists forms etc.
Layouts View	Hierarchies that control screen format and appearance of the views.
Intents	Messages wiring components together.
Resources	External elements, such as strings, constants and drawable pictures.
Manifest	Configuration file for the application.

Activity\Fragment lifecycles

Activity

Fragment

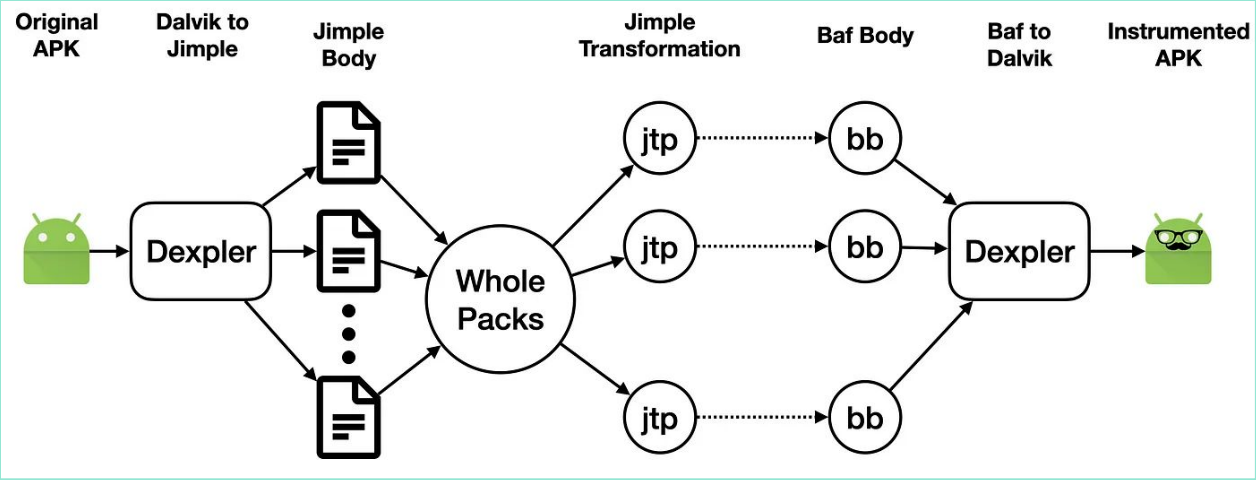

onCreate()
onStart()
onRestart()
onResume()
onPause()
onStop()
onDestroy()
Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
pps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down



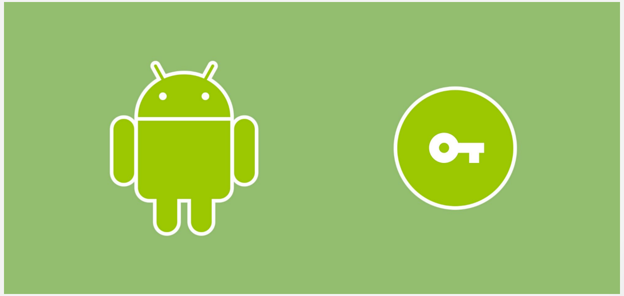
02

Instrument the app Using Soot

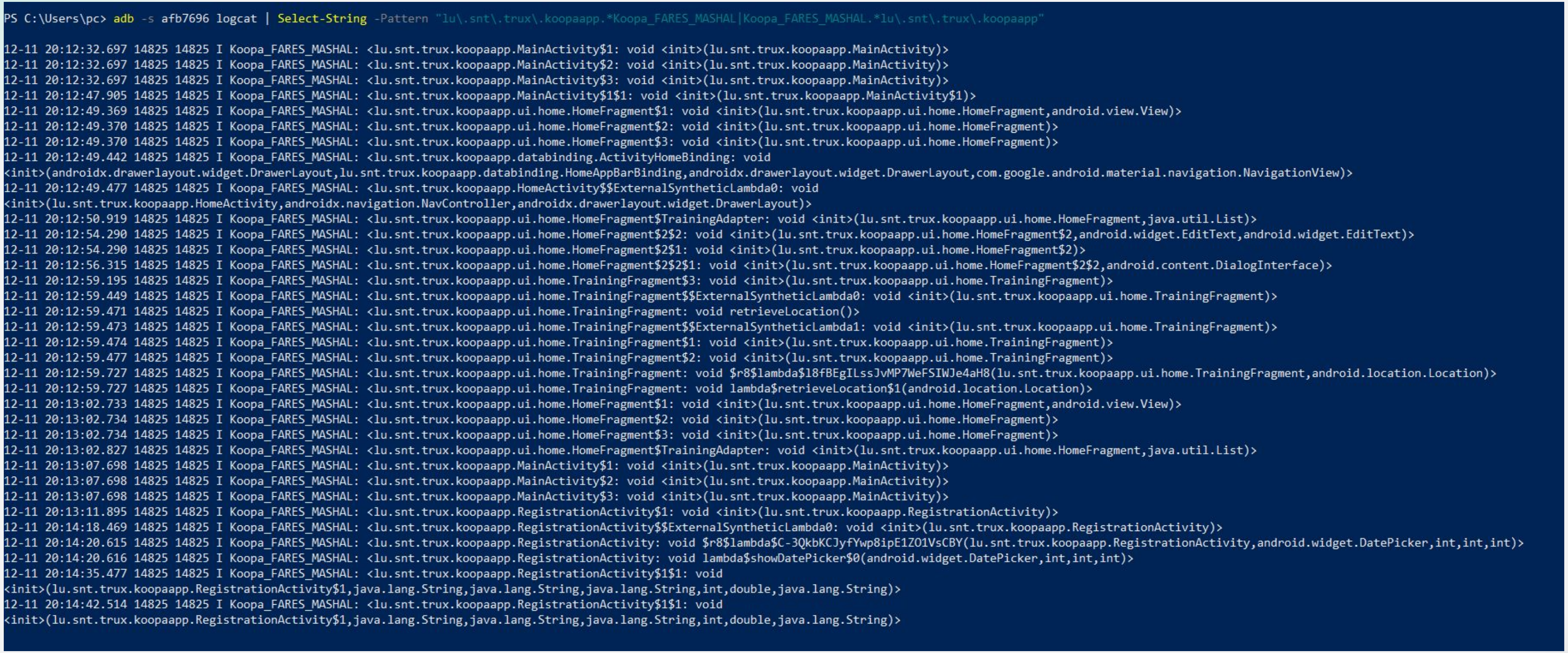
Instrumenting Using Soot



Signing the app



Instrumenting Using Soot: Output



```
PS C:\Users\pc> adb -s afb7696 logcat | Select-String -Pattern "lu\.snt\.trux\.koopaapp.*Koopa_FARES_MASHAL|Koopa_FARES_MASHAL.*lu\.snt\.trux\.koopaapp"

12-11 20:12:32.697 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.MainActivity$1: void <init>(lu.snt.trux.koopaapp.MainActivity)>
12-11 20:12:32.697 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.MainActivity$2: void <init>(lu.snt.trux.koopaapp.MainActivity)>
12-11 20:12:32.697 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.MainActivity$3: void <init>(lu.snt.trux.koopaapp.MainActivity)>
12-11 20:12:47.905 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.MainActivity$1$1: void <init>(lu.snt.trux.koopaapp.MainActivity$1)>
12-11 20:12:49.369 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$1: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment,android.view.View)>
12-11 20:12:49.370 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$2: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment)>
12-11 20:12:49.370 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$3: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment)>
12-11 20:12:49.442 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.databinding.ActivityHomeBinding: void
<init>(androidx.drawerlayout.widget.DrawerLayout,lu.snt.trux.koopaapp.databinding.HomeAppBarBinding,androidx.drawerlayout.widget.DrawerLayout,com.google.android.material.navigation.NavigationView)>
12-11 20:12:49.477 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.HomeActivity$$ExternalSyntheticLambda0: void
<init>(lu.snt.trux.koopaapp.HomeActivity,androidx.navigation.NavController,androidx.drawerlayout.widget.DrawerLayout)>
12-11 20:12:50.919 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$TrainingAdapter: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment,java.util.List)>
12-11 20:12:54.290 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$2$2: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment$2,android.widget.EditText,android.widget.EditText)>
12-11 20:12:54.290 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$2$1: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment$2)>
12-11 20:12:56.315 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$2$2$1: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment$2$2,android.content.DialogInterface)>
12-11 20:12:59.195 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment$3: void <init>(lu.snt.trux.koopaapp.ui.home.TrainingFragment)>
12-11 20:12:59.449 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment$$ExternalSyntheticLambda0: void <init>(lu.snt.trux.koopaapp.ui.home.TrainingFragment)>
12-11 20:12:59.471 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()>
12-11 20:12:59.473 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment$$ExternalSyntheticLambda1: void <init>(lu.snt.trux.koopaapp.ui.home.TrainingFragment)>
12-11 20:12:59.474 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment$1: void <init>(lu.snt.trux.koopaapp.ui.home.TrainingFragment)>
12-11 20:12:59.477 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment$2: void <init>(lu.snt.trux.koopaapp.ui.home.TrainingFragment)>
12-11 20:12:59.727 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment: void $r8$lambda$18fBEgILssJvMP7WeFSIWJe4aH8(lu.snt.trux.koopaapp.ui.home.TrainingFragment,android.location.Location)>
12-11 20:12:59.727 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.TrainingFragment: void lambda$retrieveLocation$1(android.location.Location)>
12-11 20:13:02.733 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$1: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment,android.view.View)>
12-11 20:13:02.734 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$2: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment)>
12-11 20:13:02.734 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$3: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment)>
12-11 20:13:02.827 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.ui.home.HomeFragment$TrainingAdapter: void <init>(lu.snt.trux.koopaapp.ui.home.HomeFragment,java.util.List)>
12-11 20:13:07.698 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.MainActivity$1: void <init>(lu.snt.trux.koopaapp.MainActivity)>
12-11 20:13:07.698 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.MainActivity$2: void <init>(lu.snt.trux.koopaapp.MainActivity)>
12-11 20:13:07.698 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.MainActivity$3: void <init>(lu.snt.trux.koopaapp.MainActivity)>
12-11 20:13:11.895 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.RegistrationActivity$1: void <init>(lu.snt.trux.koopaapp.RegistrationActivity)>
12-11 20:14:18.469 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.RegistrationActivity$$ExternalSyntheticLambda0: void <init>(lu.snt.trux.koopaapp.RegistrationActivity)>
12-11 20:14:20.615 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.RegistrationActivity: void $r8$lambda$C-3QkbKCJyfYwp8ipE1ZO1VsCBY(lu.snt.trux.koopaapp.RegistrationActivity,android.widget.DatePicker,int,int,int)>
12-11 20:14:20.616 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.RegistrationActivity: void lambda$showDatePicker$0(android.widget.DatePicker,int,int,int)>
12-11 20:14:35.477 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.RegistrationActivity$1$1: void
<init>(lu.snt.trux.koopaapp.RegistrationActivity$1,java.lang.String,java.lang.String,java.lang.String,int,double,java.lang.String)>
12-11 20:14:42.514 14825 14825 I Koopa_FARES_MASHAL: <lu.snt.trux.koopaapp.RegistrationActivity$1$1: void
<init>(lu.snt.trux.koopaapp.RegistrationActivity$1,java.lang.String,java.lang.String,java.lang.String,int,double,java.lang.String)>
```

Exploring the app and analysing logs

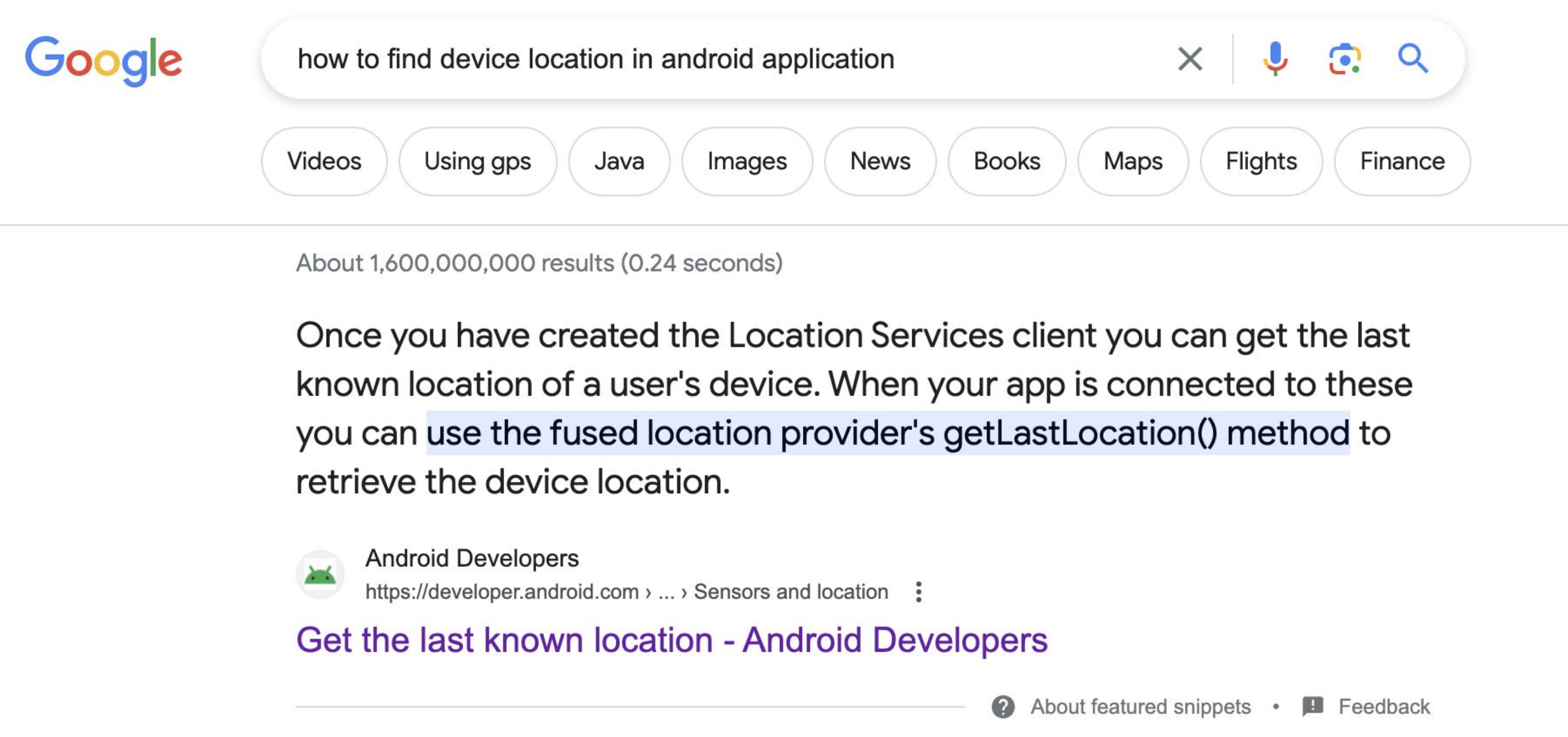
The main Activities/Fragments existing on the app based on the logs

Activities	APIs(like Google)	Fragment
MainActivity	Other activities	HomeFragment
HomeActivity	Other Fragments	TrainingFragment
RegistrationActivity		MyAccountFragment

03

Location Source Identification and Finding the Shortest Path

Finding the device's location in Android



About 1,600,000,000 results (0.24 seconds)

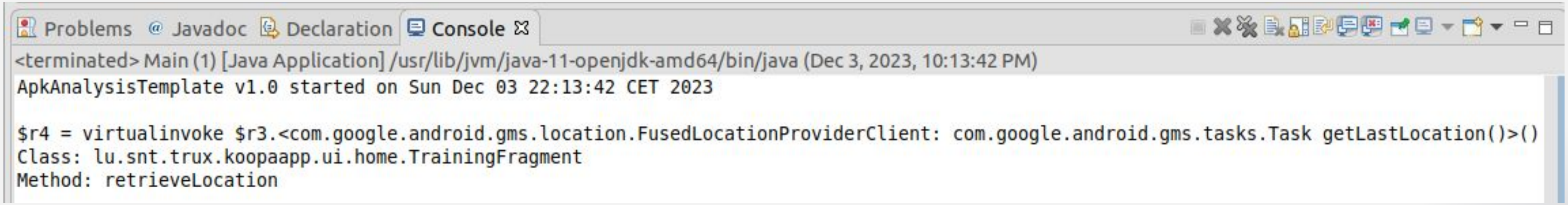
Once you have created the Location Services client you can get the last known location of a user's device. When your app is connected to these you can use the fused location provider's getLastLocation() method to retrieve the device location.

Android Developers
https://developer.android.com › ... › Sensors and location

Get the last known location - Android Developers

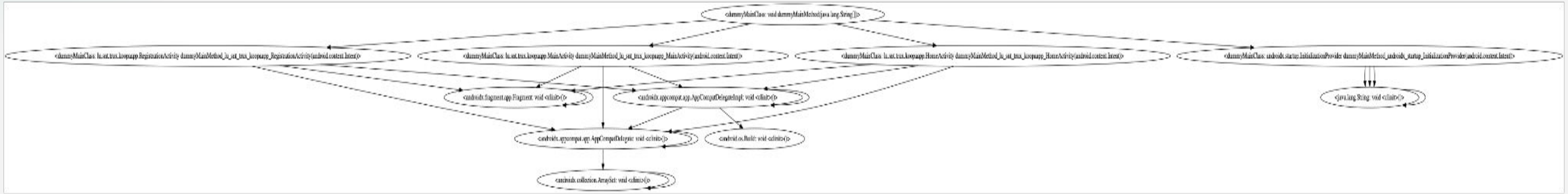
fineLocationMethod()

- Iterates over all classes in the Scene.
- Filters classes belonging to the "lu.snt.trux.koopaapp" package.
- For each method in these packages:
  - Checks if method names contain the substring "location."
  - If found, retrieves the method body.
  - Iterates over each statement in the method body.
  - Checks if the statement contains an invoke statement.
  - If invoking the "getLastLocation" method, prints method name, class (Activity/Fragment), and the identified statement.



```
<terminated> Main (1) [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Dec 3, 2023, 10:13:42 PM)
ApkAnalysisTemplate v1.0 started on Sun Dec 03 22:13:42 CET 2023

$r4 = virtualinvoke $r3.<com.google.android.gms.location.FusedLocationProviderClient: com.google.android.gms.tasks.Task getLastLocation()>()
Class: lu.snt.trux.koopaapp.ui.home.TrainingFragment
Method: retrieveLocation
```

Finding the Shortest Path: Constructing the CallGraph


<dummyMainClass: void dummyMainMethod(java.lang.String[])>
<dummyMainClass: lu.uni.trux.loopapp.RegistrationActivity dummyMainMethod_lu_uni_trux_loopapp_RegistrationActivity(android.content.Intent)>
<dummyMainClass: lu.uni.trux.loopapp.MainActivity dummyMainMethod_lu_uni_trux_loopapp_MainActivity(android.content.Intent)>
<dummyMainClass: lu.uni.trux.loopapp.HomeActivity dummyMainMethod_lu_uni_trux_loopapp_HomeActivity(android.content.Intent)>
<dummyMainClass: androidx.startup.InitializationProvider dummyMainMethod_androidx_startup_InitializationProvider(android.content.Intent)>
<androidx.fragment.app.Fragment: void <clinit>()>
<androidx.appcompat.app.AppCompatDelegateImpl: void <clinit>()>
<java.lang.String: void <clinit>()>
<androidx.appcompat.app.AppCompatDelegate: void <clinit>()>
<android.os.Build: void <clinit>()>
<androidx.collection.ArraySet: void <clinit>()>

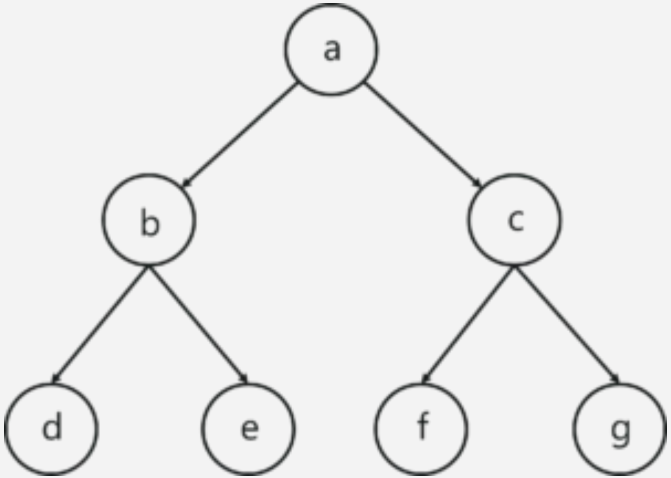
- Android apps have complicated parts like Activities and Services with tricky lifecycles.
- Certain code parts get called automatically, making it hard to follow the flow.

Constructing the CallGraph using Flowdroid on top of Soot.

- InfoflowAndroidConfiguration
- SetupApplication
- Using CHA algorithm
- Contains 2299176 nodes.

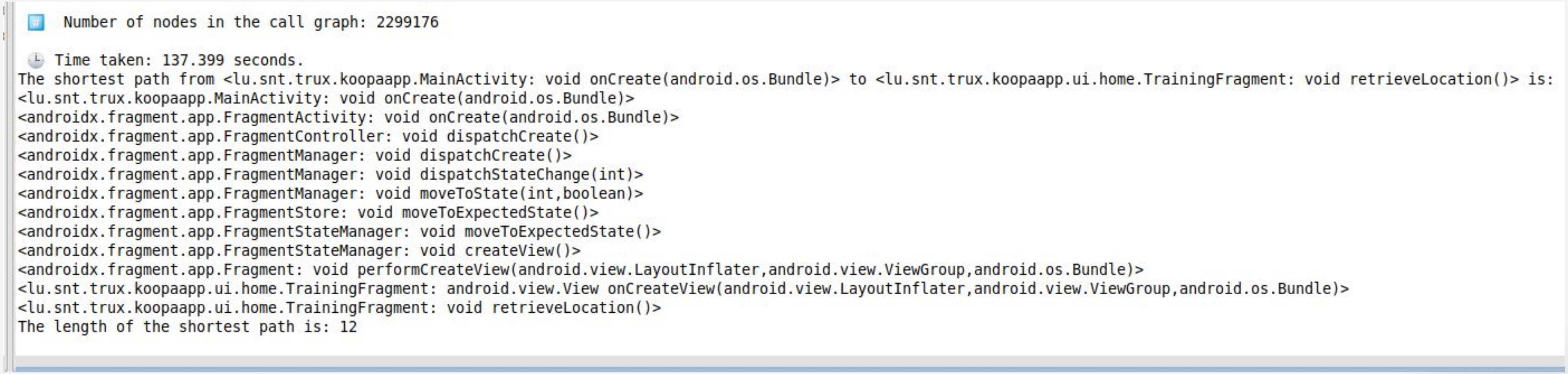
Finding the Shortest Path: Breadth First Search Algorithm

- **BFS Algorithm Overview:**
  - Traverses a graph level by level.
  - Systematically explores nodes starting from the initial node.
  - Ensures that all nodes at the current level are visited before moving to the next level.
- **Using BFS for Shortest Path Between 2 Methods:**
  - Represent methods as nodes in a graph.
  - Create edges based on method calls.
  - Start BFS from the source method.
  - Explore neighbors (methods called) and enqueue unvisited methods.
  - Record parent information during traversal.
  - Stop when the destination method is reached.
  - Reconstruct the shortest path using parent information.



Finding the Shortest Path: Output

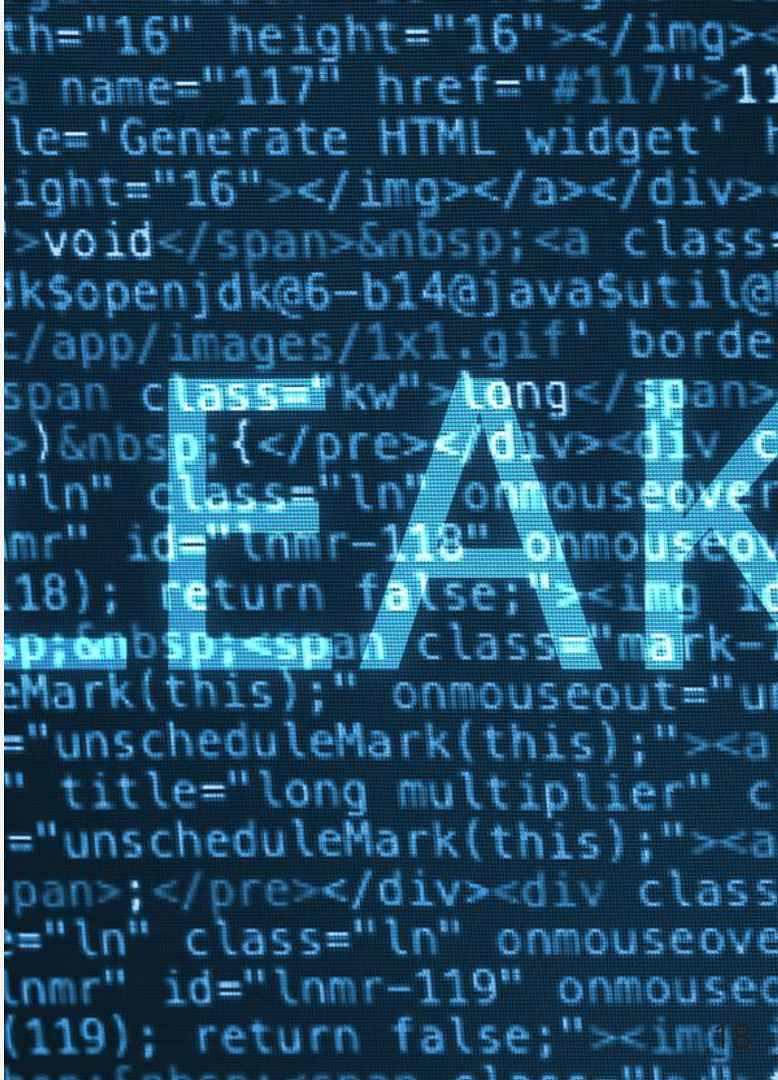
We find the shortest path from the entry point, the onCreate() method of the MainActivity, to the retrieveLocation() of the TrainingFragment.



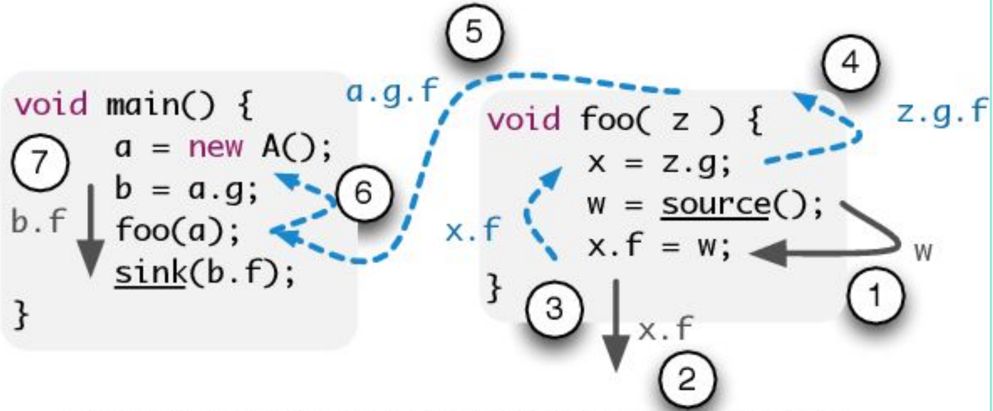
```
#️⃣  Number of nodes in the call graph: 2299176

🕒 Time taken: 137.399 seconds.
The shortest path from <lu.snt.trux.koopaapp.MainActivity: void onCreate(android.os.Bundle)> to <lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()> is:
<lu.snt.trux.koopaapp.MainActivity: void onCreate(android.os.Bundle)>
<androidx.fragment.app.FragmentActivity: void onCreate(android.os.Bundle)>
<androidx.fragment.app.FragmentController: void dispatchCreate()>
<androidx.fragment.app.FragmentManager: void dispatchCreate()>
<androidx.fragment.app.FragmentManager: void dispatchStateChange(int)>
<androidx.fragment.app.FragmentManager: void moveToState(int,boolean)>
<androidx.fragment.app.FragmentStore: void moveToExpectedState()>
<androidx.fragment.app.FragmentStateManager: void moveToExpectedState()>
<androidx.fragment.app.FragmentStateManager: void createView()>
<androidx.fragment.app.Fragment: void performCreateView(android.view.LayoutInflater,android.view.ViewGroup,android.os.Bundle)>
<lu.snt.trux.koopaapp.ui.home.TrainingFragment: android.view.View onCreateView(android.view.LayoutInflater,android.view.ViewGroup,android.os.Bundle)>
<lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()>
The length of the shortest path is: 12
```

04

Data leaks Detection



FlowDroid Taint analysis



```
void main() {
    a = new A();
    b = a.g;
    foo(a);
    sink(b.f);
}

void foo( z ) {
    x = z.g;
    w = source();
    x.f = w;
}
```

Figure 2: Taint analysis under realistic aliasing

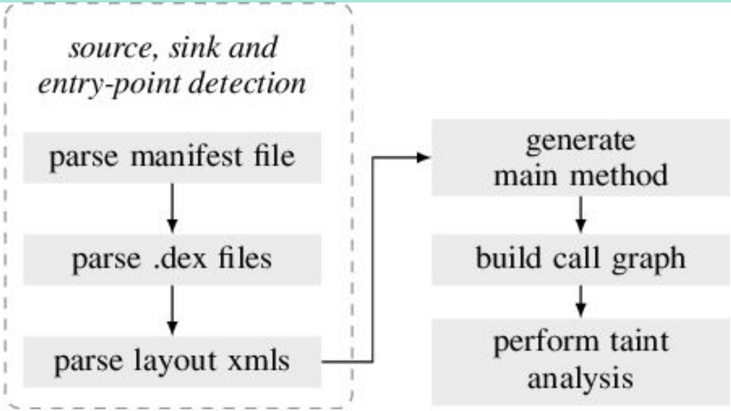


Figure 4: Overview of FlowDroid

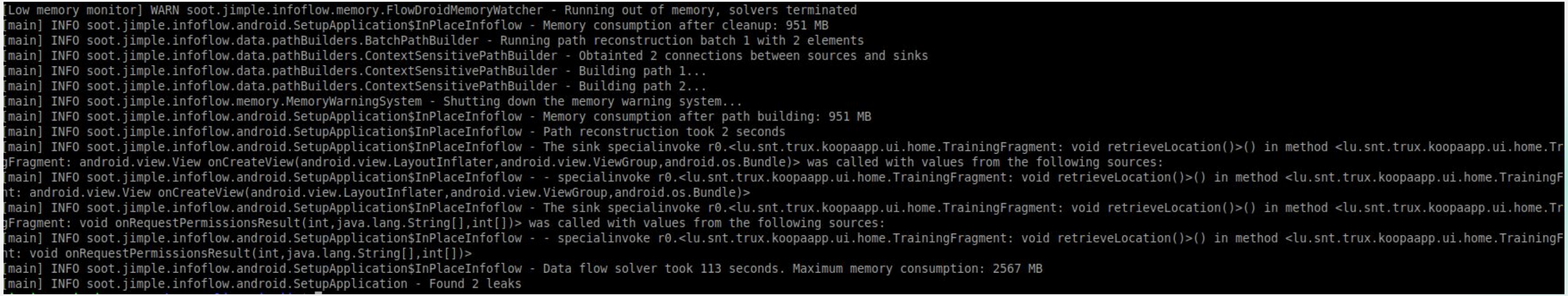
FlowDroid Taint analysis: Implementation

1. Building the Flowdroid Project with Maven
2. Generating the flowDroid.jar File
3. Executing the command with the necessary arguments

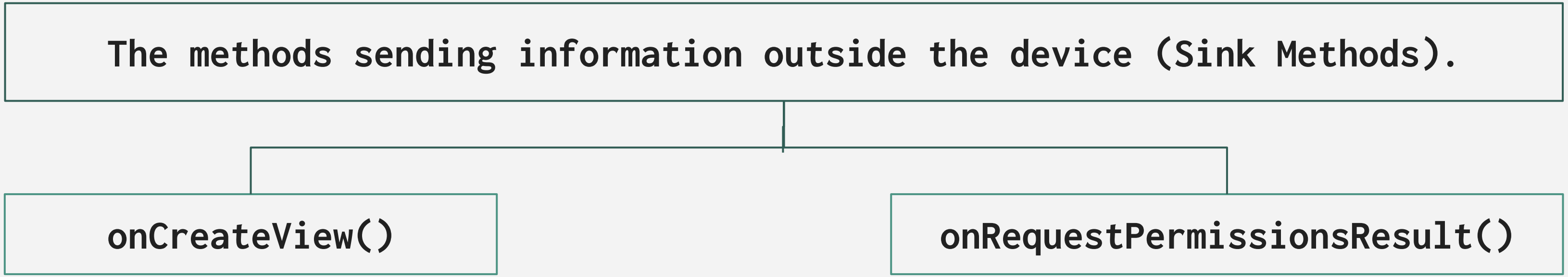
The command:

```
java -jar FlowDroid.jar -a koopaApp.apk -p
/home/micsispa/Desktop/Android-platforms/jars/stub -s SourcesAndSinks.txt
\ -d -cg CHA -t EasyTaintWrapperSource.txt -o analysis-results.xml
```

FlowDroid Taint analysis: Result



```
[Low memory monitor] WARN soot.jimple.infoflow.memory.FlowDroidMemoryWatcher - Running out of memory, solvers terminated
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - Memory consumption after cleanup: 951 MB
[main] INFO soot.jimple.infoflow.data.pathBuilders.BatchPathBuilder - Running path reconstruction batch 1 with 2 elements
[main] INFO soot.jimple.infoflow.data.pathBuilders.ContextSensitivePathBuilder - Obtainted 2 connections between sources and sinks
[main] INFO soot.jimple.infoflow.data.pathBuilders.ContextSensitivePathBuilder - Building path 1...
[main] INFO soot.jimple.infoflow.data.pathBuilders.ContextSensitivePathBuilder - Building path 2...
[main] INFO soot.jimple.infoflow.memory.MemoryWarningSystem - Shutting down the memory warning system...
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - Memory consumption after path building: 951 MB
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - Path reconstruction took 2 seconds
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - The sink specialinvoke r0.<lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()>() in method <lu.snt.trux.koopaapp.ui.home.Tr
gFragment: android.view.View onCreateView(android.view.LayoutInflater,android.view.ViewGroup,android.os.Bundle)> was called with values from the following sources:
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - - specialinvoke r0.<lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()>() in method <lu.snt.trux.koopaapp.ui.home.TrainingF
nt: android.view.View onCreateView(android.view.LayoutInflater,android.view.ViewGroup,android.os.Bundle)>
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - The sink specialinvoke r0.<lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()>() in method <lu.snt.trux.koopaapp.ui.home.Tr
gFragment: void onRequestPermissionsResult(int,java.lang.String[],int[])> was called with values from the following sources:
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - - specialinvoke r0.<lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()>() in method <lu.snt.trux.koopaapp.ui.home.TrainingF
nt: void onRequestPermissionsResult(int,java.lang.String[],int[])>
[main] INFO soot.jimple.infoflow.android.SetupApplication$InPlaceInfoflow - Data flow solver took 113 seconds. Maximum memory consumption: 2567 MB
[main] INFO soot.jimple.infoflow.android.SetupApplication - Found 2 leaks
```

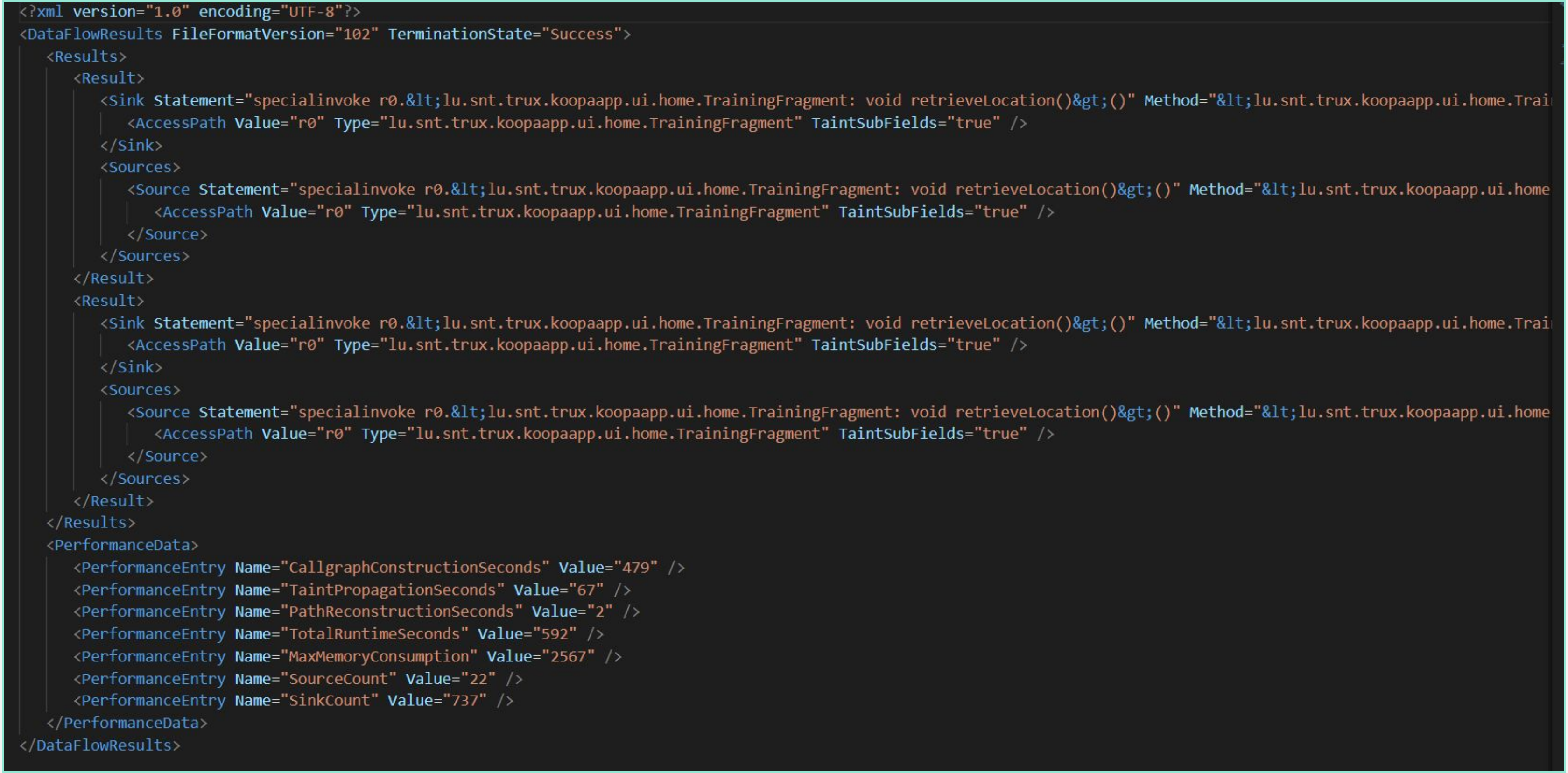


The methods sending information outside the device (Sink Methods).

onCreateView()

onRequestPermissionsResult()

FlowDroid Taint analysis: Result XML File

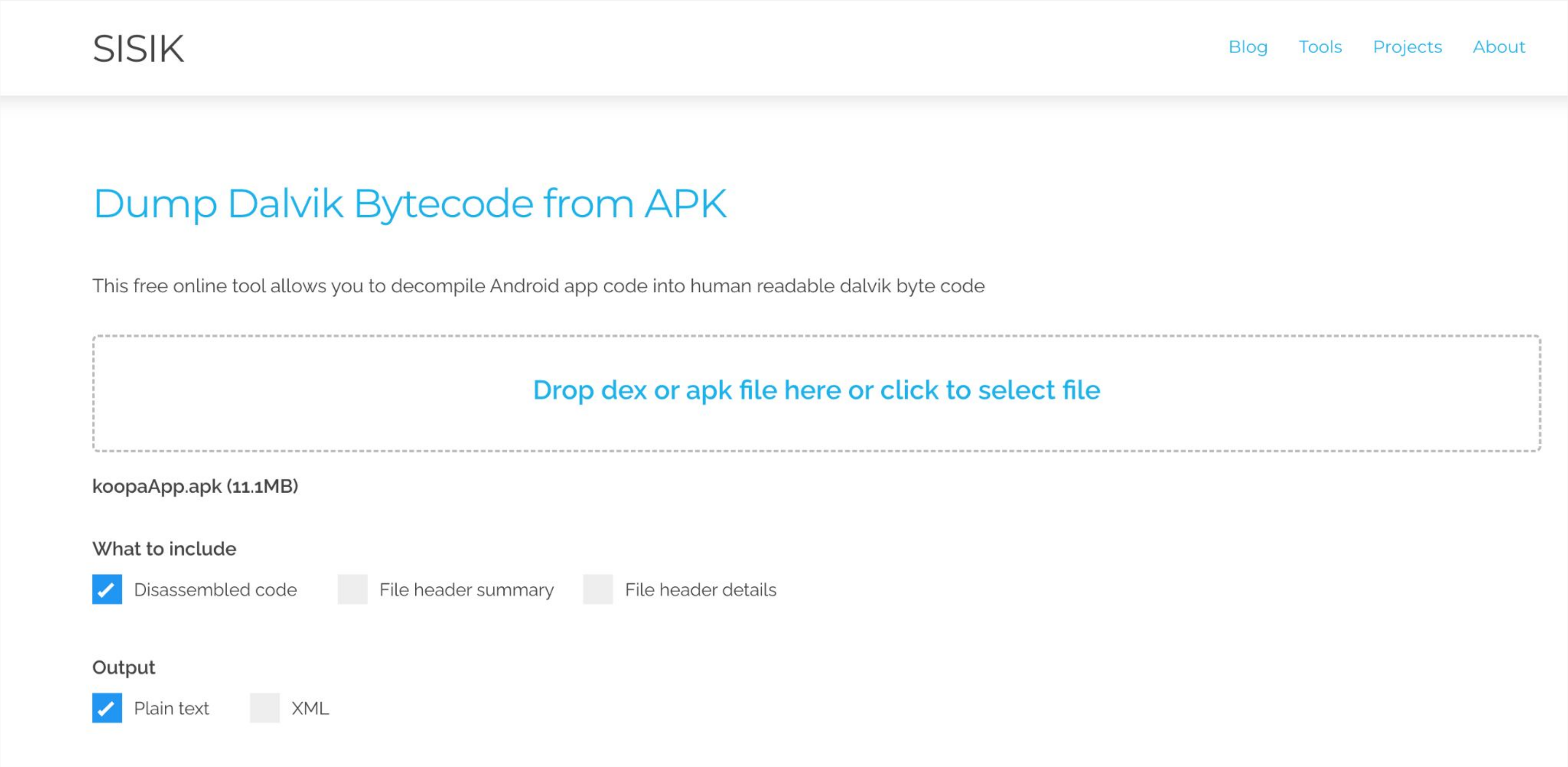


```xml
<?xml version="1.0" encoding="UTF-8"?>
<DataFlowResults FileFormatVersion="102" TerminationState="Success">
   <Results>
      <Result>
         <Sink Statement="specialinvoke r0.&lt;lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()&gt;()" Method="&lt;lu.snt.trux.koopaapp.ui.home.Trai
            <AccessPath Value="r0" Type="lu.snt.trux.koopaapp.ui.home.TrainingFragment" TaintSubFields="true" />
         </Sink>
         <Sources>
            <Source Statement="specialinvoke r0.&lt;lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()&gt;()" Method="&lt;lu.snt.trux.koopaapp.ui.home
               <AccessPath Value="r0" Type="lu.snt.trux.koopaapp.ui.home.TrainingFragment" TaintSubFields="true" />
            </Source>
         </Sources>
      </Result>
      <Result>
         <Sink Statement="specialinvoke r0.&lt;lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()&gt;()" Method="&lt;lu.snt.trux.koopaapp.ui.home.Trai
            <AccessPath Value="r0" Type="lu.snt.trux.koopaapp.ui.home.TrainingFragment" TaintSubFields="true" />
         </Sink>
         <Sources>
            <Source Statement="specialinvoke r0.&lt;lu.snt.trux.koopaapp.ui.home.TrainingFragment: void retrieveLocation()&gt;()" Method="&lt;lu.snt.trux.koopaapp.ui.home
               <AccessPath Value="r0" Type="lu.snt.trux.koopaapp.ui.home.TrainingFragment" TaintSubFields="true" />
            </Source>
         </Sources>
      </Result>
   </Results>
   <PerformanceData>
      <PerformanceEntry Name="CallgraphConstructionSeconds" Value="479" />
      <PerformanceEntry Name="TaintPropagationSeconds" Value="67" />
      <PerformanceEntry Name="PathReconstructionSeconds" Value="2" />
      <PerformanceEntry Name="TotalRuntimeSeconds" Value="592" />
      <PerformanceEntry Name="MaxMemoryConsumption" Value="2567" />
      <PerformanceEntry Name="SourceCount" Value="22" />
      <PerformanceEntry Name="SinkCount" Value="737" />
   </PerformanceData>
</DataFlowResults>
```

05

LLM Experiment

Dumping Bytecode from the koopaAPP APK file



Dump Dalvik Bytecode from APK

This free online tool allows you to decompile Android app code into human readable dalvik byte code

koopaApp.apk (11.1MB)

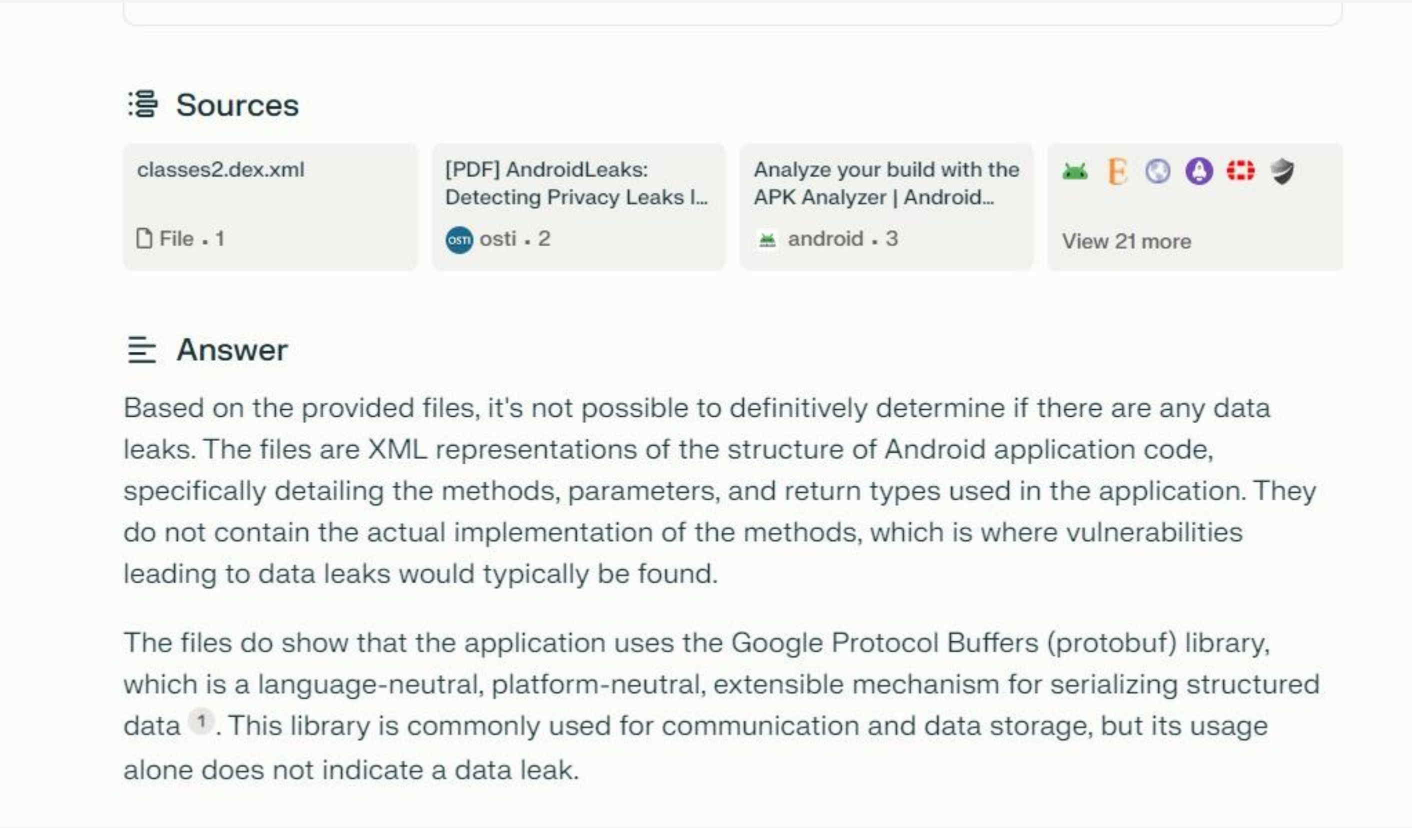
What to include

☑ Disassembled code ☐ File header summary ☐ File header details

Output

☑ Plain text ☐ XML

LLM 1: Perplexity AI



Sources

classes2.dex.xml

File . 1

[PDF] AndroidLeaks: Detecting Privacy Leaks I...

osti . 2

Analyze your build with the APK Analyzer | Android...

android . 3

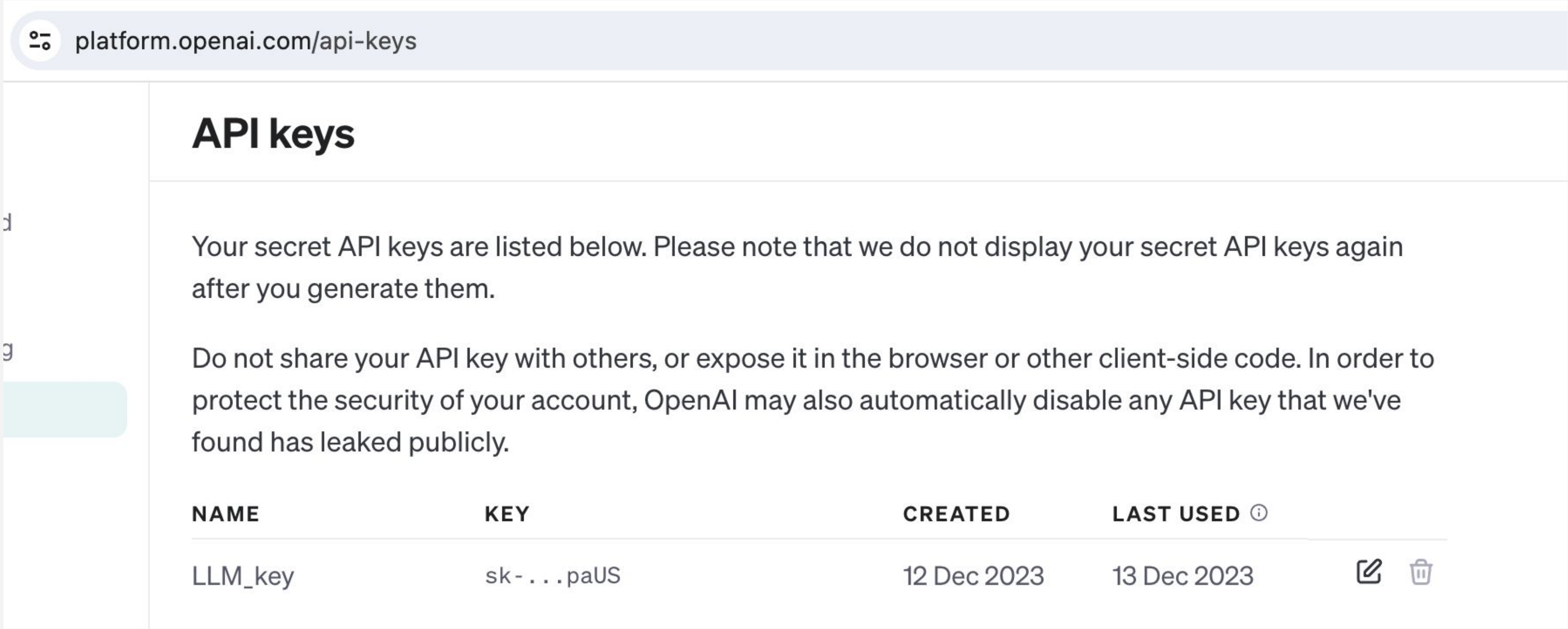
View 21 more

Answer

Based on the provided files, it's not possible to definitively determine if there are any data leaks. The files are XML representations of the structure of Android application code, specifically detailing the methods, parameters, and return types used in the application. They do not contain the actual implementation of the methods, which is where vulnerabilities leading to data leaks would typically be found.

The files do show that the application uses the Google Protocol Buffers (protobuf) library, which is a language-neutral, platform-neutral, extensible mechanism for serializing structured data [1]. This library is commonly used for communication and data storage, but its usage alone does not indicate a data leak.

LLM 2: ChatGPT



platform.openai.com/api-keys

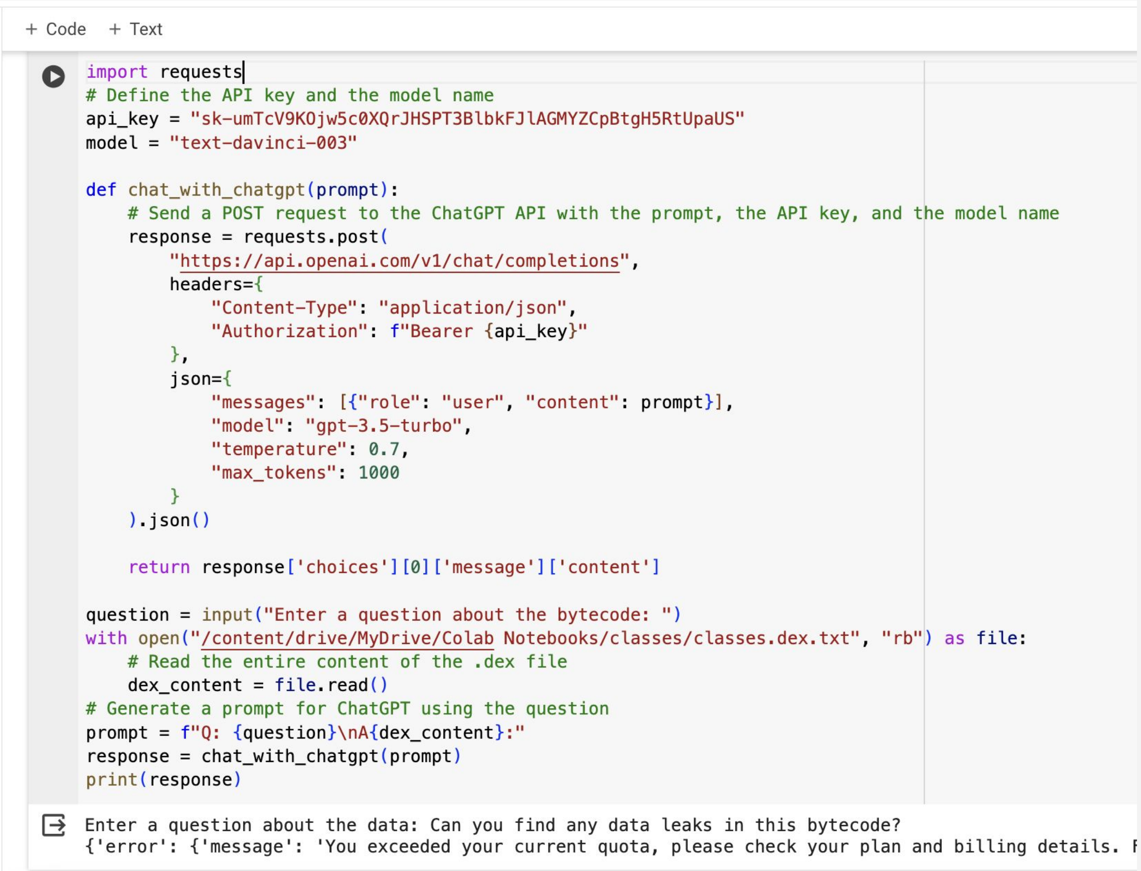
API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically disable any API key that we've found has leaked publicly.

NAME	KEY	CREATED	LAST USED ⓘ
LLM_key	`sk-...paUS`	12 Dec 2023	13 Dec 2023

LLM 2: ChatGPT



```
import requests
# Define the API key and the model name
api_key = "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"
model = "text-davinci-003"

def chat_with_chatgpt(prompt):
    # Send a POST request to the ChatGPT API with the prompt, the API key, and the model name
    response = requests.post(
        "https://api.openai.com/v1/chat/completions",
        headers={
            "Content-Type": "application/json",
            "Authorization": f"Bearer {api_key}"
        },
        json={
            "messages": [{"role": "user", "content": prompt}],
            "model": "gpt-3.5-turbo",
            "temperature": 0.7,
            "max_tokens": 1000
        }
    ).json()

    return response['choices'][0]['message']['content']

question = input("Enter a question about the bytecode: ")
with open("/content/drive/MyDrive/Colab Notebooks/classes/classes.dex.txt", "rb") as file:
    # Read the entire content of the .dex file
    dex_content = file.read()
# Generate a prompt for ChatGPT using the question
prompt = f"Q: {question}\nA{dex_content}:"
response = chat_with_chatgpt(prompt)
print(response)
```

```
Enter a question about the data: Can you find any data leaks in this bytecode?
{'error': {'message': 'You exceeded your current quota, please check your plan and billing details.
```

Thank you!