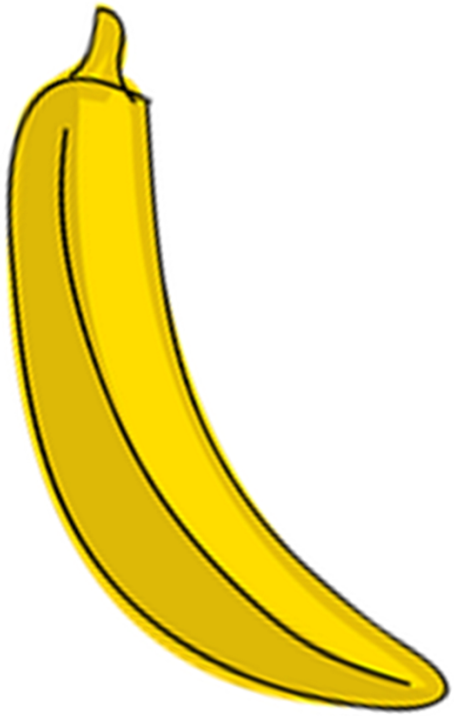


Project Group:

BANANA

Bare **N**ative **A**ndroid **Ana**lysis

2019/20

http://go.upb.de/BANANA

28.01.2019

BANANA: Motivation

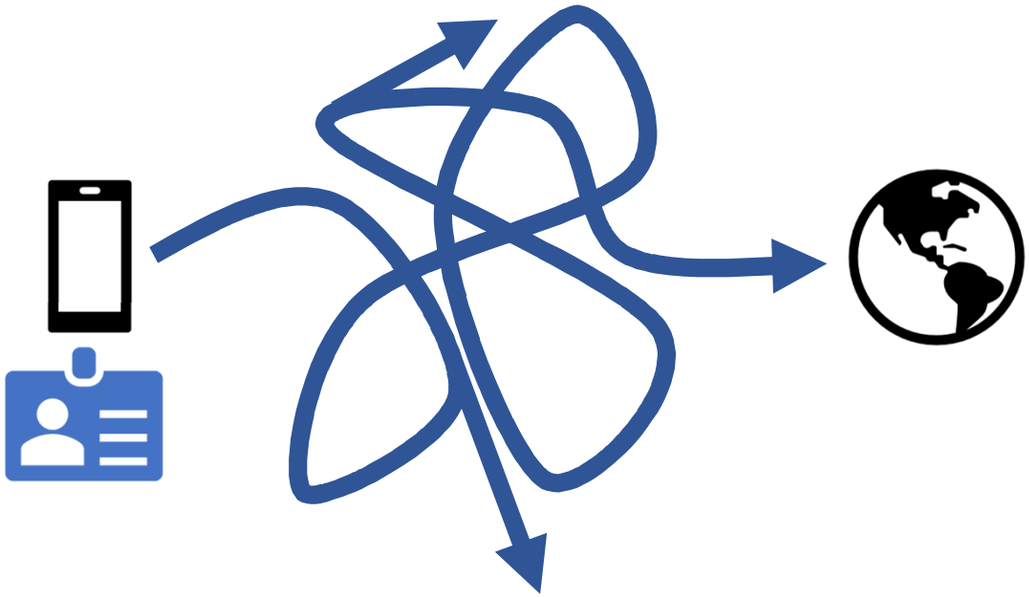
Reports for motivation
(omitted for upload)

BANANA: Overview

- **Topics:**
  - Taint-flow analysis
  - Slicing
  - Java + Assembler
  - Static + Dynamic

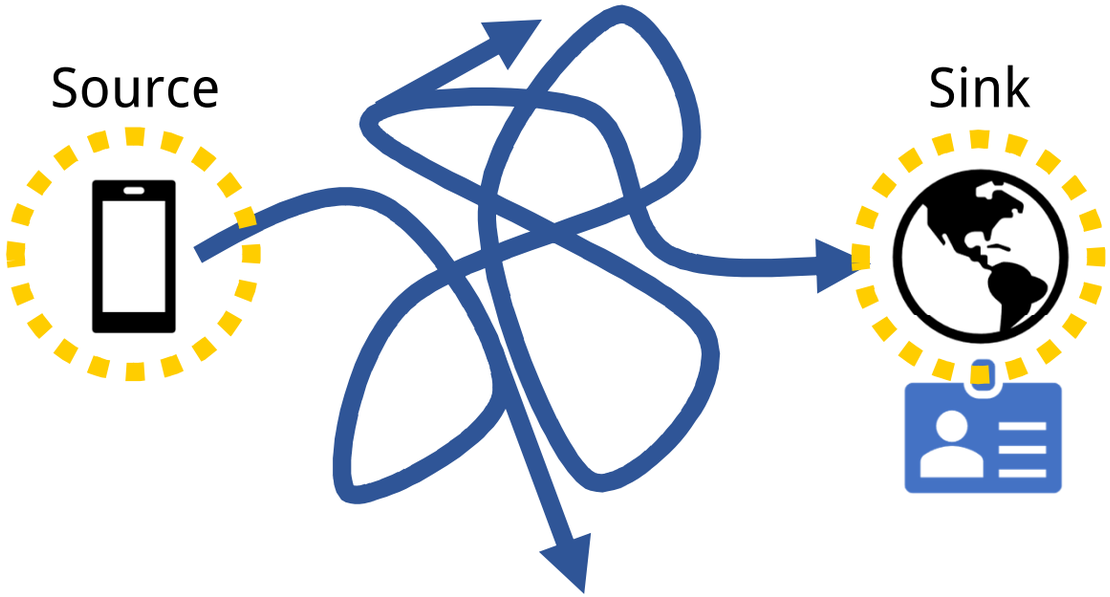
BANANA: Overview

- **Topics:**
  - **Taint-flow analysis** 🍌
  - Slicing
  - Java + Assembler
  - Static + Dynamic



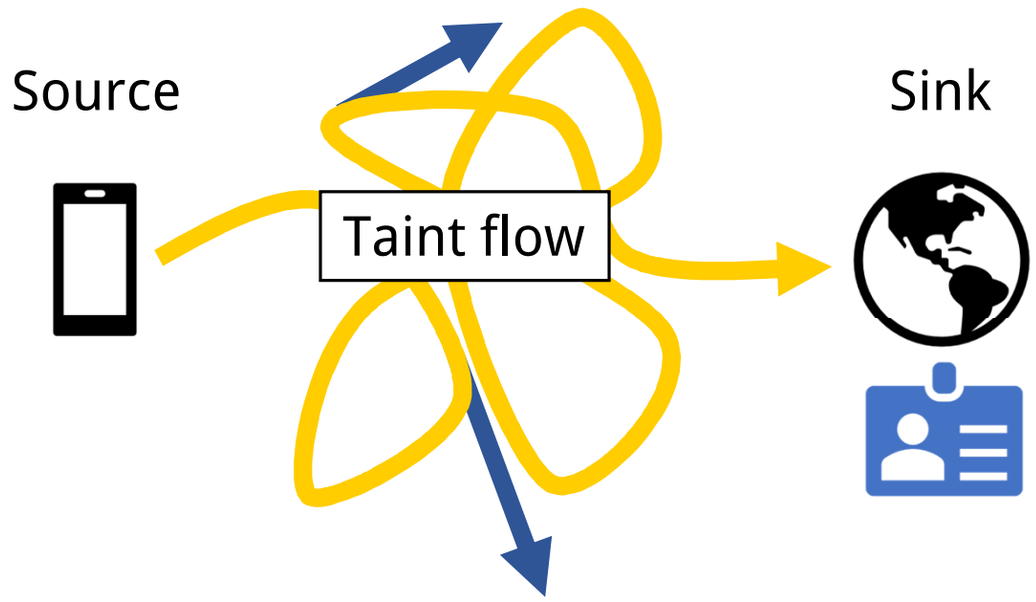
BANANA: Overview

- **Topics:**
  - **Taint-flow analysis**
  - Slicing
  - Java + Assembler
  - Static + Dynamic



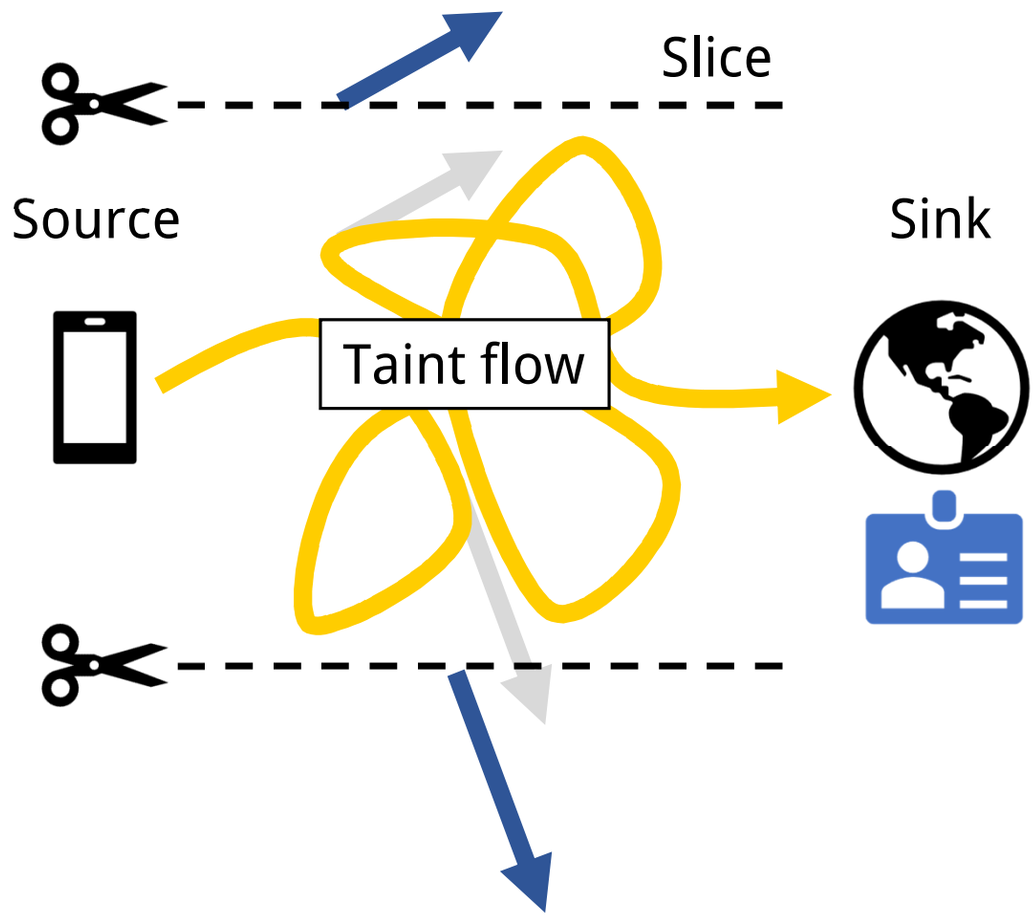
BANANA: Overview

- **Topics:**
  - **Taint-flow analysis**
  - Slicing
  - Java + Assembler
  - Static + Dynamic



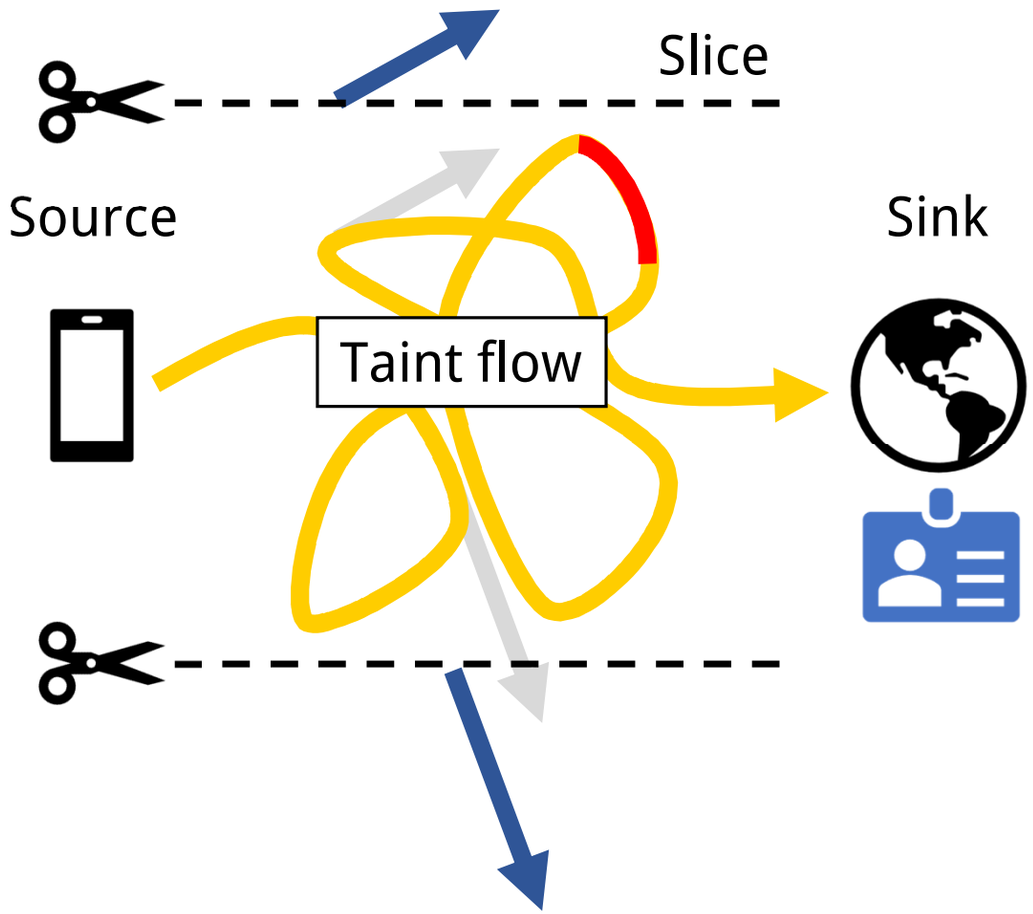
BANANA: Overview

- **Topics:**
  - Taint-flow analysis
  - **Slicing**
  - Java + Assembler
  - Static + Dynamic



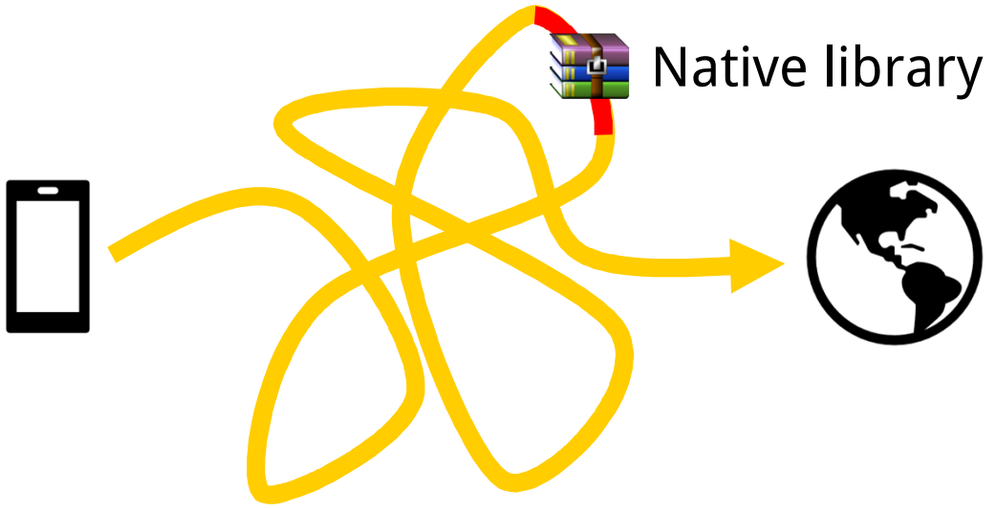
BANANA: Overview

- **Topics:**
  - Taint-flow analysis
  - Slicing
  - **Java + Assembler**
  - Static + Dynamic



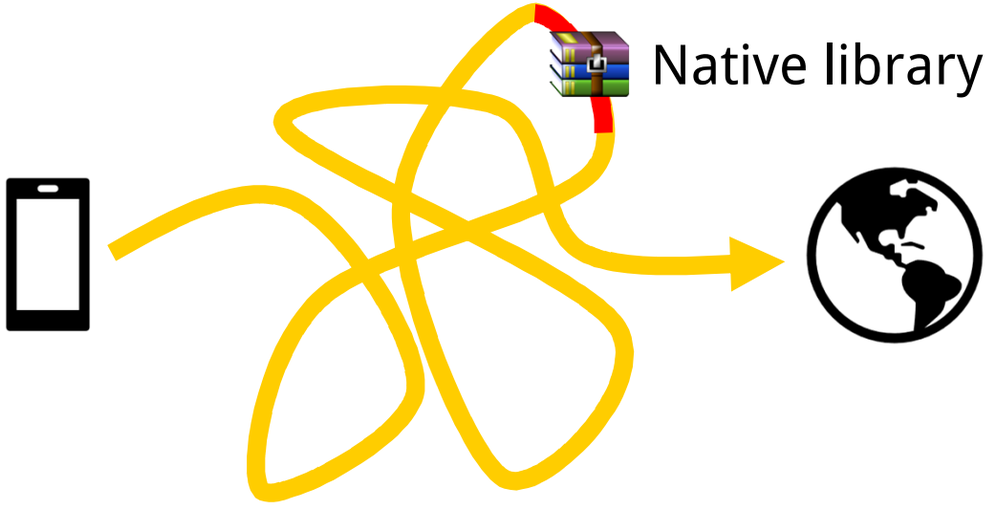
BANANA: Overview

- **Topics:**
  - Taint-flow analysis
  - Slicing
  - **Java + Assembler**
  - Static + Dynamic



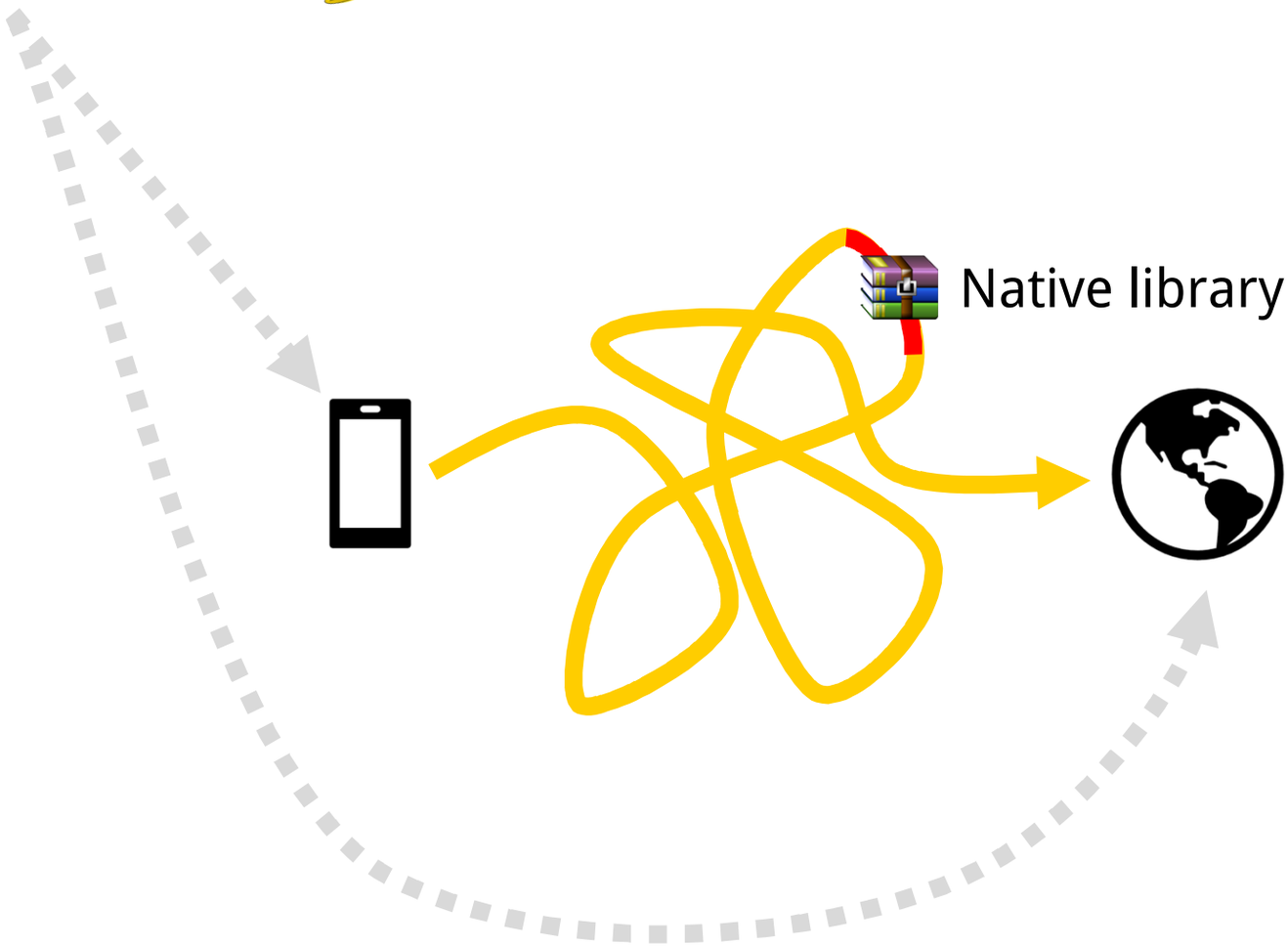
BANANA: Overview

- **Topics:**
  - Taint-flow analysis
  - Slicing
  - Java + Assembler
  - **Static + Dynamic**



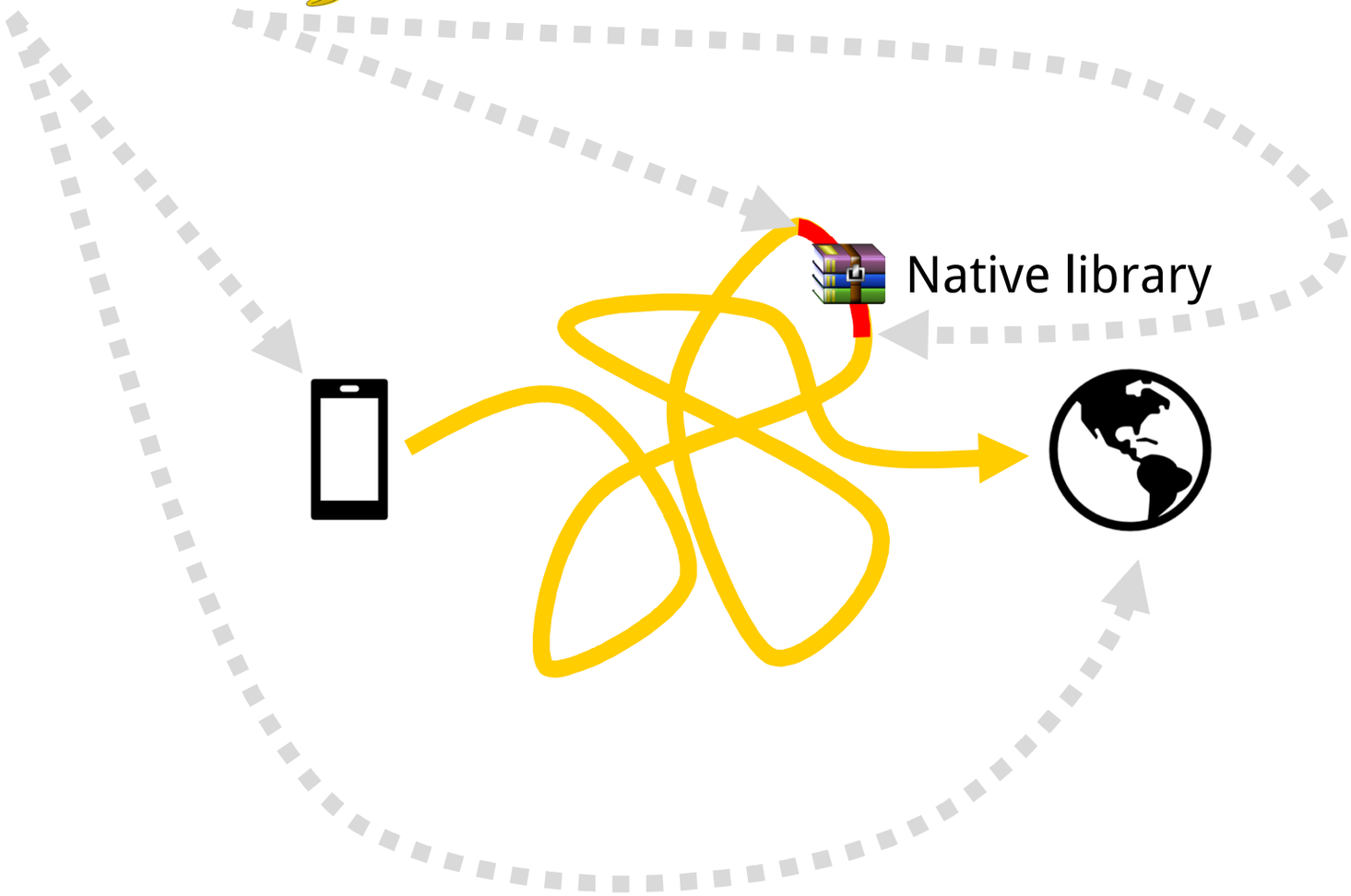
BANANA: Overview

- **Topics:**
  - Taint-flow analysis
  - Slicing
  - Java + Assembler
  - **Static + Dynamic**



BANANA: Overview

- **Topics:**
  - Taint-flow analysis
  - Slicing
  - Java + Assembler
  - **Static + Dynamic**



BANANA: Challenge

- **1. Java/Android + C source code**

BANANA: Challenge

- **1. Java/Android + C source code**

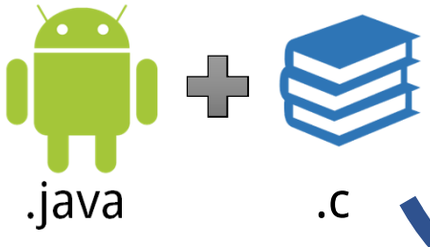

```
public class MainActivity extends Activity {

    ...

    static {
        System.loadLibrary("native-lib");
    }

    public native boolean doSomethingJNI();

    private void iAmExecuted() {
        Log.wtf(doSomethingJNI());
    }

    ...
}
```

- **1. Java/Android + C source code**


```
// Get the TELEPHONY_SERVICE constant
jfieldID TELEPHONY_SERVICE = env->GetStaticFieldID(clsCtx, "TELEPHONY_SERVICE", "Ljava/lang/String;");
jstring str = (jstring) env->GetStaticObjectField(clsCtx,TELEPHONY_SERVICE);

// Get the TelephonyManager
jmethodID getSystemService = env->GetMethodID(clsCtx, "getSystemService", "(Ljava/lang/String;)Ljava/lang/Object;");
jobject telephonymanager = env->CallObjectMethod(cxt, getSystemService,str);

// Get the device ID
jclass clsTm = env->FindClass("android/telephony/TelephonyManager");
jmethodID getDeviceId = env->GetMethodID(clsTm, "getDeviceId", "()Ljava/lang/String;");
jstring strIMEI = (jstring) env->CallObjectMethod(telephonymanager,getDeviceId);

// Write it back
return strIMEI;
```

BANANA: Challenge

- **1. Java/Android + C source code**



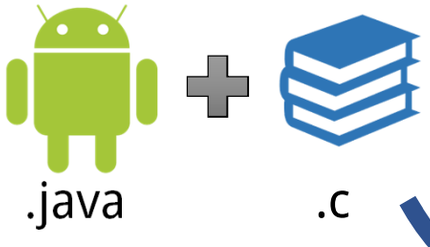
```
public class MainActivity extends Activity {
    ...

    static {
        System.loadLibrary("native-lib");
    }

    public native boolean doSomethingJNI();

    private void iAmExecuted() {
        Log.wtf(doSomethingJNI());
    }

    ...
    }
```

- **2. Android Application Package**

.apk

BANANA: Challenge

- **1. Java/Android + C source code**

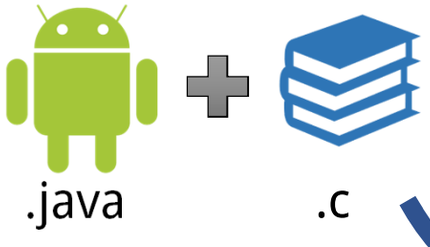


- **2. Android Application Package**

- **3. Unzipped APK**

BANANA: Challenge

- **1. Java/Android + C source code**



- **2. Android Application Package**

- **3. Unzipped APK**

BANANA: Challenge

- Identify Sources & Sinks

```
// Get the TELEPHONY_SERVICE constant
jfieldID TELEPHONY_SERVICE = env->GetStaticFieldID(clsCtx, "TELEPHONY_SERVICE", "Ljava/lang/String;");
jstring str = (jstring) env->GetStaticObjectField(clsCtx,TELEPHONY_SERVICE);

// Get the TelephonyManager
jmethodID getSystemService = env->GetMethodID(clsCtx, "getSystemService", "(Ljava/lang/String;)Ljava/lang/Object;");
jobject telephonymanager = env->CallObjectMethod(cxt, getSystemService,str);

// Get the device ID
jclass clsTm = env->FindClass("android/telephony/TelephonyManager");
jmethodID getDeviceId = env->GetMethodID(clsTm, "getDeviceId", "()Ljava/lang/String;");
jstring strIMEI = (jstring) env->CallObjectMethod(telephonymanager,getDeviceId);

// Write it back
return strIMEI;
```

- Identify Sources & Sinks

```
// Get the TELEPHONY_SERVICE constant
jfieldID TELEPHONY_SERVICE = env->GetStaticFieldID(clsCtx, "TELEPHONY_SERVICE", "Ljava/lang/String;");
jstring str = (jstring) env->GetStaticObjectField(clsCtx,TELEPHONY_SERVICE);

// Get the TelephonyManager
jmethodID getSystemService = env->GetMethodID(clsCtx, "getSystemService", "(Ljava/lang/String;)Ljava/lang/Object;");
jobject telephonymanager = env->CallObjectMethod(cxt, getSystemService,str);

// Get the device ID
jclass clsTm = env->FindClass("android/telephony/TelephonyManager");
jmethodID getDeviceId = env->GetMethodID(clsTm, "getDeviceId", "()Ljava/lang/String;");
jstring strIMEI = (jstring) env->CallObjectMethod(telephonymanager,getDeviceId);

// Write it back
return strIMEI;
```

BANANA: Challenge

- Identify Sources & Sinks

```
237c 4e444b5f 4558414d 504c4500 25730061  NDK_EXAMPLE.%s.a
238c 6e64726f 69642f63 6f6e7465 6e742f43  ndroid/content/C
239c 6f6e7465 78740054 454c4550 484f4e59  ontext.TELEPHONY
23ac 5f534552 56494345 004c6a61 76612f6c  _SERVICE.Ljava/l
23bc 616e672f 53747269 6e673b00 67657453  ang/String;.getS
23cc 79737465 6d536572 76696365 00284c6a  ystemService.(Lj
23dc 6176612f 6c616e67 2f537472 696e673b  ava/lang/String;
23ec 294c6a61 76612f6c 616e672f 4f626a65  )Ljava/lang/Obje
23fc 63743b00 616e6472 6f69642f 74656c65  ct;.android/tele
240c 70686f6e 792f5465 6c657068 6f6e794d  phony/TelephonyM
241c 616e6167 65720067 65744465 76696365  anager.getDevice
242c 49640028 294c6a61 76612f6c 616e672f  Id.()Ljava/lang/
243c 53747269 6e673b00 0a2d2063 46756e63  String;..- cFunc
244c 4a676574 494d4549 28293a5b 25735d00  JgetIMEI():[%s].
```

```
// Get the TELEPHONY_SERVICE constant
jfieldID TELEPHONY_SERVICE = env->GetSt
jstring str = (jstring) env->GetStaticO

// Get the TelephonyManager
jmethodID getSystemService = env->GetMethodID(clsCtx, "getSystemService", "(Ljava/lang/String;)Ljava/lang/Object;");
jobject telephonymanager = env->CallObjectMethod(cxt, getSystemServi,str);

// Get the device ID
jclass clsTm = env->FindClass("android/telephony/TelephonyManager");
jmethodID getDeviceId = env->GetMethodID(clsTm, "getDeviceId", "()Ljava/lang/String;");
jstring strIMEI = (jstring) env->CallObjectMethod(telephonymanager,getDeviceId);

// Write it back
return strIMEI;
```

BANANA: Challenge

- Identify Sources & Sinks

```
; ============== F U N C T I O N ==================================

unknown Java_mod_ndk_ActMain_cFuncGetIMEI (unknown)

mrcmi 5, 1, fp, cr1, cr0, {7}
ldmdbmi r1!, {r0, r2, r6, r7, ip, sp, pc}
ldmdavs r6!, {r1, r2, r3, r4, r5, r6, sl, lr}
ldrbtmi r9, [r9], #-513 ; 0x201
stcne 8, cr6, [r5], {51} ; 0x33
stmdavs r3, {r0, r1, r6, r8, r9, ip, pc}
; <UNDEFINED> instruction: 0x4798699b
stcne 10, cr4, [r7], {44} ; 0x2c
stmdavs r8!, {r2, r3, r5, r8, r9, fp, lr}
umulleq r2, r9, r0, r1
ldrbtmi r5, [sl], #-2116 ; 0x844
ldrbtmi r1, [fp], #-3129 ; 0xc39
strmi r1, [r0, r8, lsr #24]!
orrscs r6, r1, #2686976 ; 0x290000
stcne 0, cr0, [r2], {155} ; 0x9b
stcne 8, cr5, [r8], #-812 ; 0xfffffcd4
; <UNDEFINED> instruction: 0x47981c39
bmi loc_0091ae1c
tstcc r8, r4, lsr #22
andls r6, r0, ip, asr #31
ldrbtmi r1, [sl], #-3129 ; 0xc39
cfstrsne mvf4, [r8], #-492 ; 0xfffffe14
stmdbls r1, {r5, r7, r8, r9, sl, lr}
blls func_00007d98
; <UNDEFINED> instruction: 0xf7ff1c28
stmdavs fp!, {r0, r4, r5, r7, r8, r9, sl, fp, ip,
stcne 9, cr4, [r7], {29}
ldrbtmi r6, [r9], #-2459 ; 0x99b
ldrmi r1, [r8, r8, lsr #24]
stmdavs r8!, {r0, sl, fp, ip}
blmi func_006d3614
svcvs 0x00c43008
ldrbtmi r4, [sl], #-1147 ; 0x47b
strmi r1, [r0, r8, lsr #24]!
stcne 12, cr1, [r2], {57} ; 0x39
; <UNDEFINED> instruction: 0xf7ff1c28
stmdavs sl!, {r0, r1, r3, r4, r7, r8, r9, sl, fp,
stcne 3, cr2, [r4], {169} ; 0xa9
ldmpl r3, {r0, r1, r3, r4, r7}^
andcs r1, r0, #8448 ; 0x2100
ldrmi r1, [r8, r8, lsr #24]
stcne 9, cr4, [r2], {17}
ldrbtmi sl, [r9], #-2051 ; 0x803
svc 0x0050f7ff
; <UNDEFINED> instruction: 0xf7ffa803
bls loc_01100c44
stcne 8, cr6, [r0], #-204 ; 0xffffff34
mulle r1, sl, r2
svc 0x004cf7ff
ldcllt 0, cr11, [r0, #276]! ; 0x114
andeq r3, r0, r4, ror r2
andeq r1, r0, sp, asr #12
andeq r1, r0, r9, asr #12
andeq r1, r0, r7, asr r6
andeq r1, r0, r6, asr #12
andeq r1, r0, r5, asr r6
andeq r1, r0, r2, ror #12
andeq r1, r0, r1, ror r6
andeq r1, r0, pc, ror r6
andeq r1, r0, sl, ror #12
```

```
237c 4e444b5f 4558414d 504c4500 25730061  NDK_EXAMPLE.%s.a
238c 6e64726f 69642f63 6f6e7465 6e742f43  ndroid/content/C
239c 6f6e7465 78740054 454c4550 484f4e59  ontext.TELEPHONY
23ac 5f534552 56494345 004c6a61 76612f6c  _SERVICE.Ljava/l
23bc 616e672f 53747269 6e673b00 67657453  ang/String;.getS
23cc 79737465 6d536572 76696365 00284c6a  ystemService.(Lj
23dc 6176612f 6c616e67 2f537472 696e673b  ava/lang/String;
23ec 294c6a61 76612f6c 616e672f 4f626a65  )Ljava/lang/Obje
23fc 63743b00 616e6472 6f69642f 74656c65  ct;.android/tele
240c 70686f6e 792f5465 6c657068 6f6e794d  phony/TelephonyM
241c 616e6167 65720067 65744465 76696365  anager.getDevice
242c 49640028 294c6a61 76612f6c 616e672f  Id.()Ljava/lang/
243c 53747269 6e673b00 0a2d2063 46756e63  String;..- cFunc
244c 4a676574 494d4549 28293a5b 25735d00  JgetIMEI():[%s].
```

```
dID(clsCtx, "getSystemService", "(Ljava/lang/String;)Ljava/lang/Object;");
Method(cxt, getSystemService,str);

ephony/TelephonyManager");
lsTm, "getDeviceId", "()Ljava/lang/String;");
tMethod(telephonymanager,getDeviceId);
```

BANANA: Task & Goal

- **Exemplary approach:**
  - *(static)* Detect flows in the native library
    - Identify sources & sinks
    - Detect connections: sources & sinks, inputs & sinks, sources & outputs

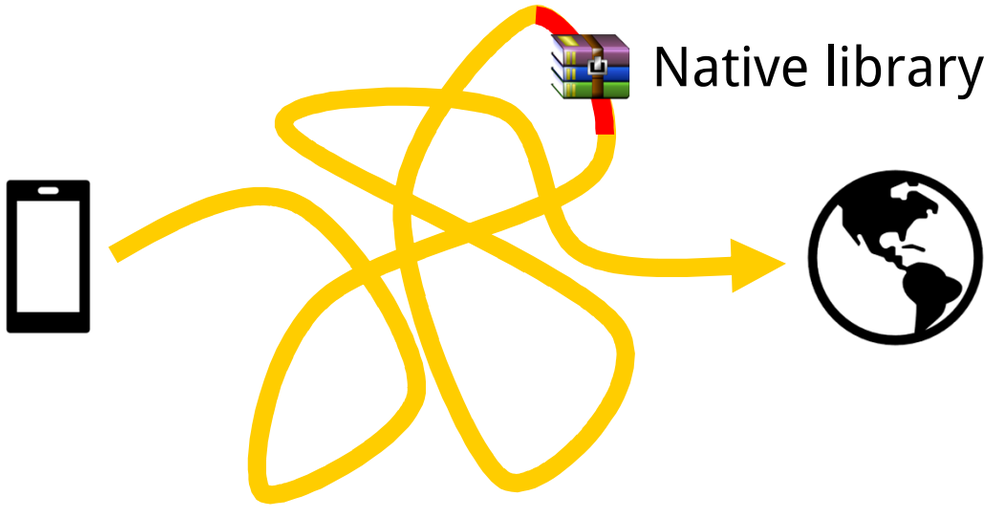
BANANA: Task & Goal

- **Exemplary approach:**
  - *(static)* Detect flows in the native library
    - Identify sources & sinks
    - Detect connections: sources & sinks, inputs & sinks, sources & outputs
  - *(dynamic)* Observe behavior:
    - Slice app to JNI call
    - Simulate with different/all-possible inputs
    - Observe outputs

BANANA: Task & Goal

- **Exemplary approach:**
  - *(static)* Detect flows in the native library
    - Identify sources & sinks
    - Detect connections: sources & sinks, inputs & sinks, sources & outputs
  - *(dynamic)* Observe behavior:
    - Slice app to JNI call
    - Simulate with different/all-possible inputs
    - Observe outputs

- **Goals:**
  - Less false-positives than over-approximation (only) techniques



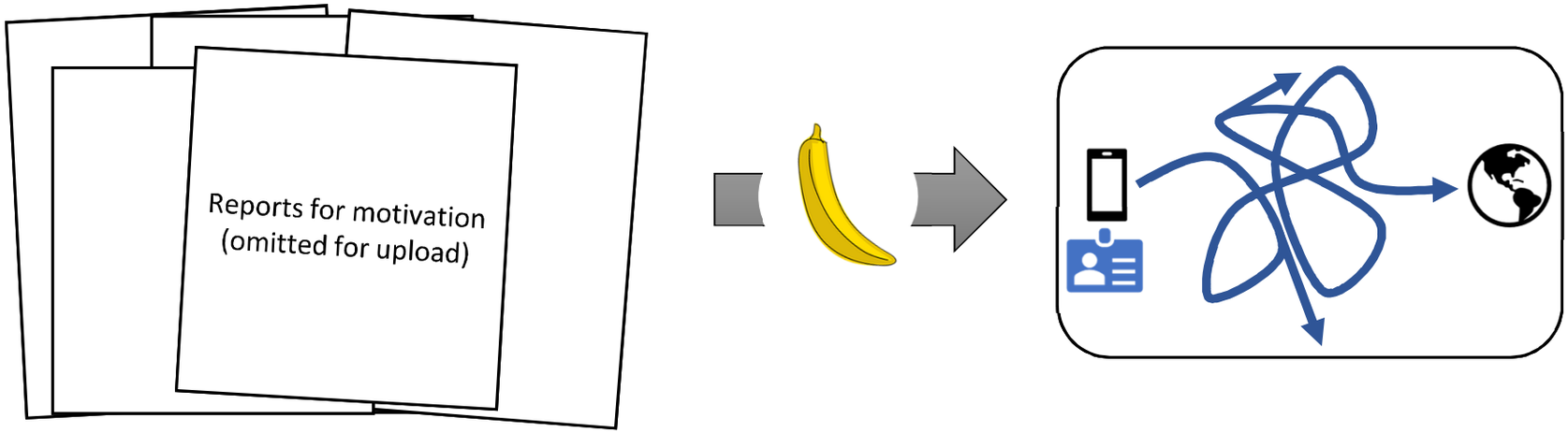
BANANA: Requirements

- **Required**
  - Ability to read and understand **Java/Android** programs
  - Knowledge about software design and efficient programming

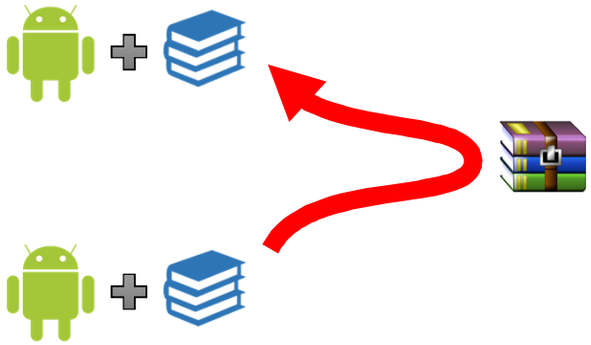
- **Nice-To-Have**
  - Interest in **reverse engineering** and **software analysis**
  - Experience with software analysis frameworks and tools such as **Soot** and **FlowDroid**
  - Knowledge about the JNI

BANANA: Summary

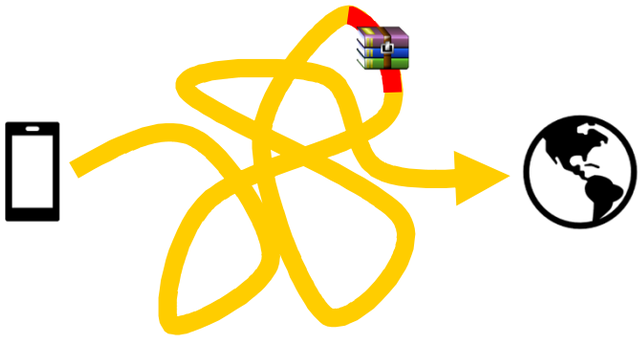
- What is **BANANA** about?



- What is the **challenge** to be overcome?

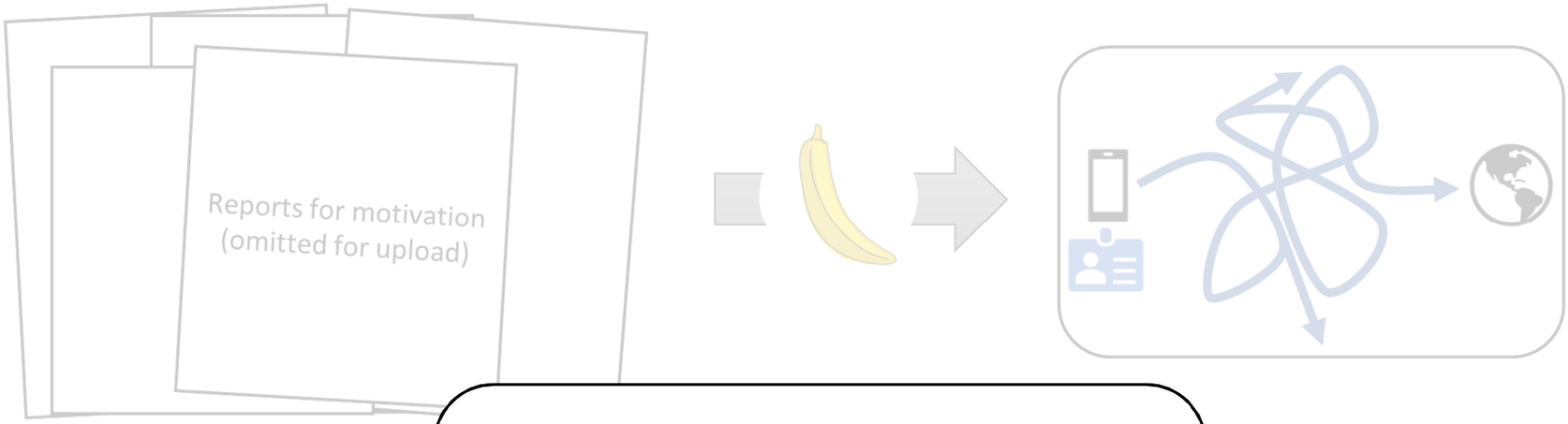


- What is the **goal**?



BANANA: Summary

- What is **BANANA** about?



https://go.upb.de/BANANA

Felix Pauck
fpauck@mail.upb.de

- What is the **challenge** to be overcome?

- What is the **goal**?