

HOOKER

A solution to analyze Android markets

Dimitri Kirchner – @Tibapbedoum

Georges Bossert – @Lapeluche

AMOSSYS

GEORGES BOSSERT

PhD candidate AMOSSYS / Supelec
IT security engineer AMOSSYS
Protocole Reverse Engineering
Android

DIMITRI KIRCHNER

IT security engineer at AMOSSYS since 2010

Android

Informatique de confiance







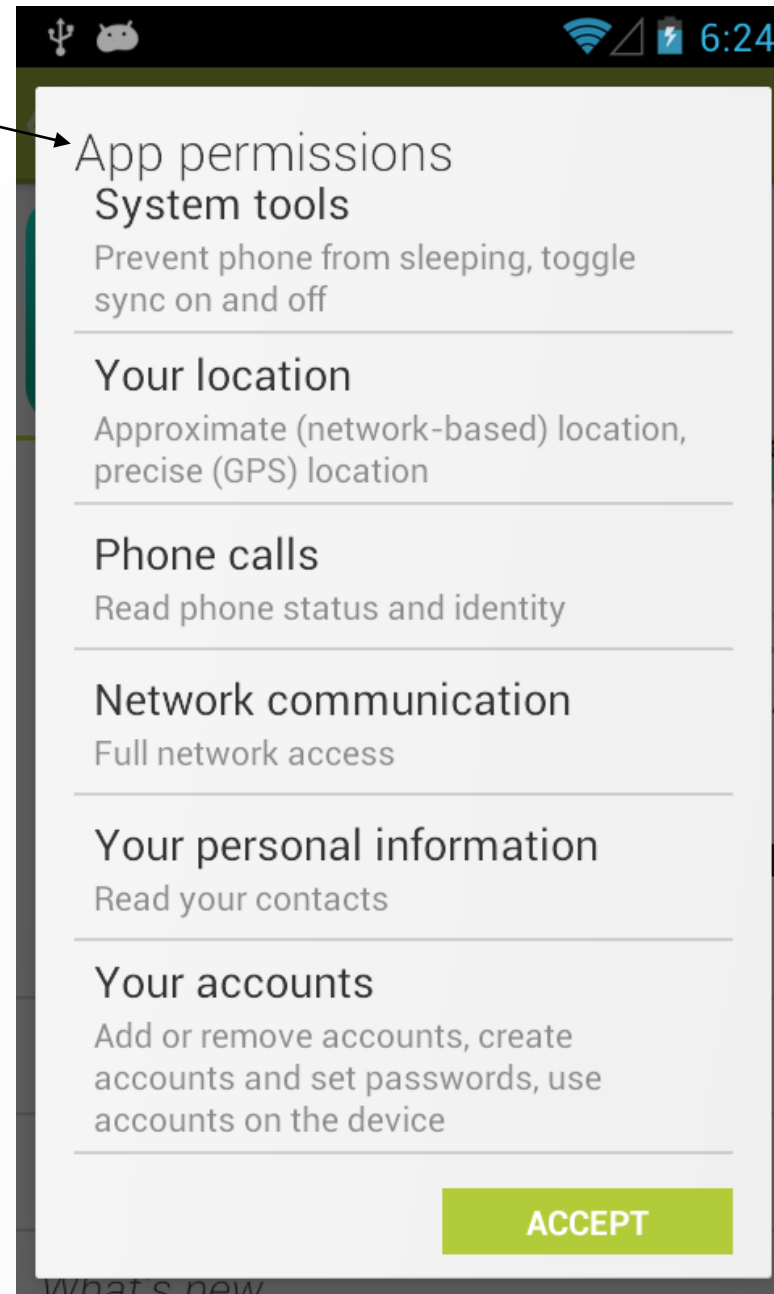




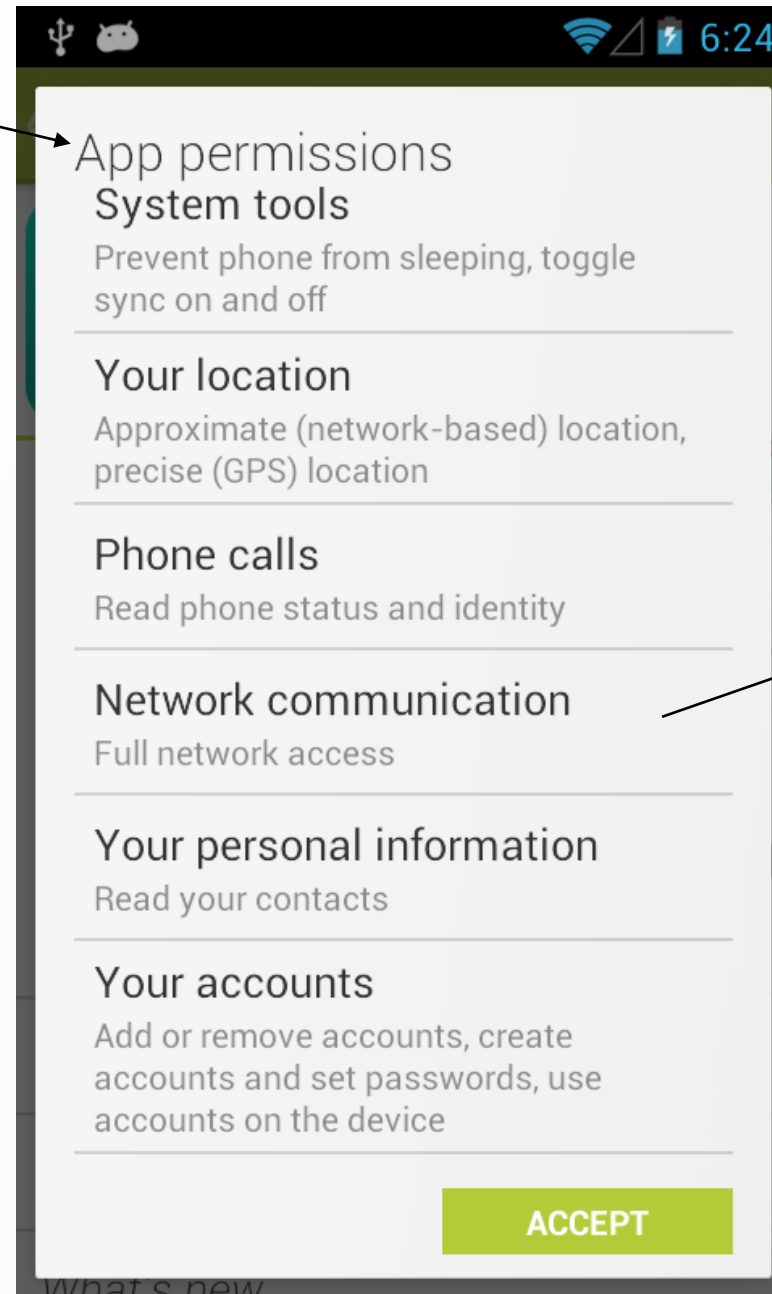
Android security model

Ask the user for permissions in order to access phones
ressources (texts, GPS, etc.)

RATP

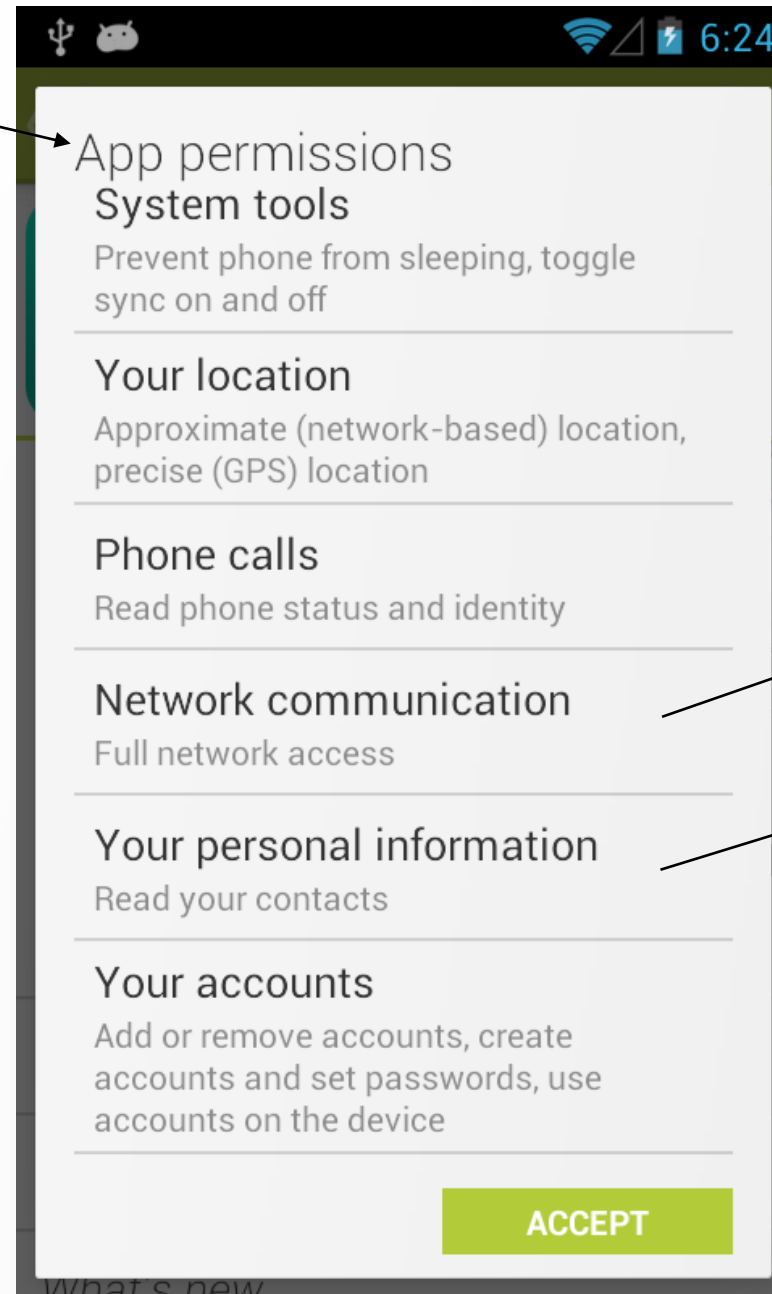


RATP



Access the internet

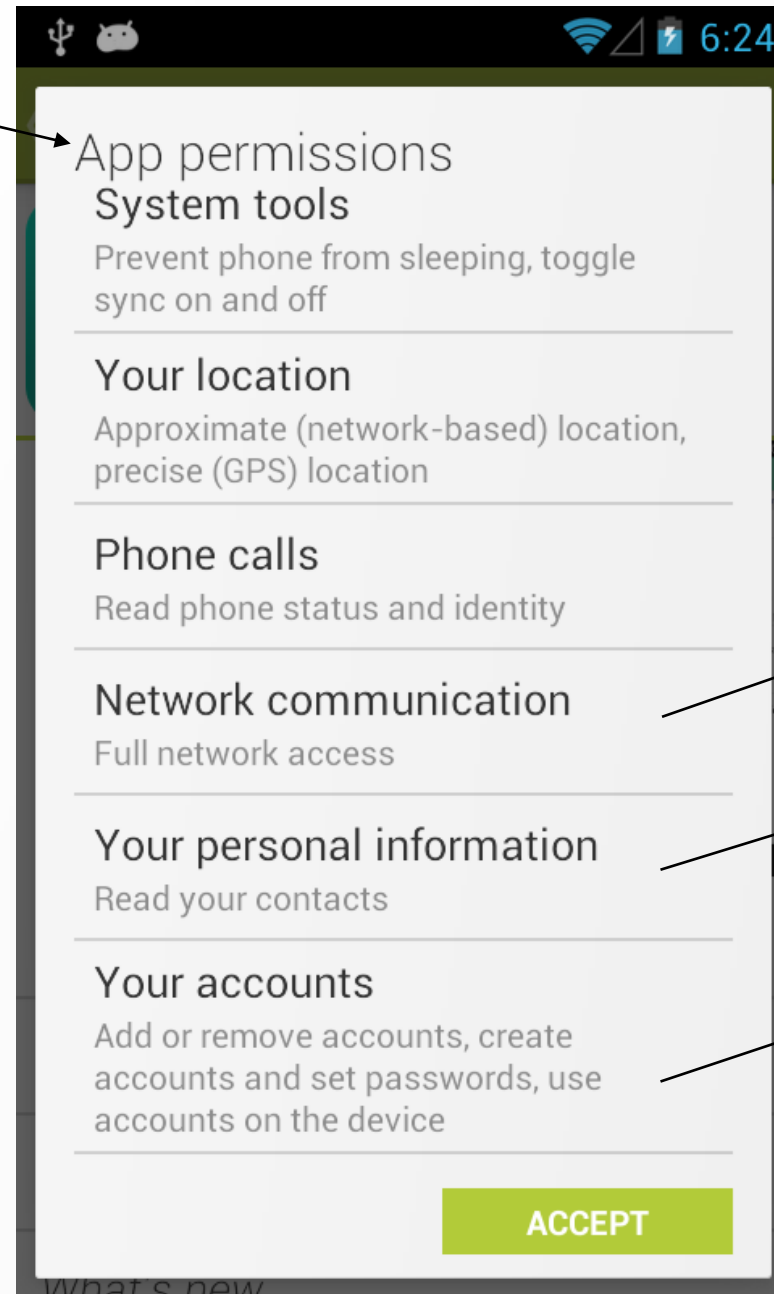
RATP



Access the internet

Read my contacts

RATP



Let's say, I really need this app...

What the application does with its resources ?

Are resources really used by the application ?

Are resources used in a legitimate way ?

You already have solutions for that

Static versus dynamic analysis tools



Androguard
JD-Core/GUI
Etc.

Dynamic analysis

Solution 1: Build a custom Android ROM (Droidbox) to instrument the kernel

Dynamic analysis

Solution 1: Build a custom Android ROM (Droidbox) to instrument the kernel

Solution 2: Modify APK before install (APIMonitor / Fino) to instrument the APK

Dynamic analysis

Solution 1: Build a custom Android ROM (Droidbox) to instrument the kernel

Solution 2: Modify APK before install (APIMonitor / Fino) to instrument the APK

Solution 3: API hooking framework (Substrate / Xposed)

Online scanners

Mix of static and dynamic

Fancy user interface and reports



→ COOL logo

Error 524

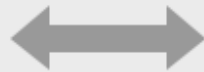
Ray ID: 12fa2ab8911204a3

A timeout occurred



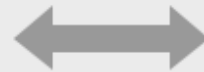
You

Browser
Working



Paris

CloudFlare
Working



www.apk-analyzer.net

Host
Error

Analysis are centered on one application

Is it possible to analyze more than one application ?

Can you analyze an entire market ?



Introducing hooker

What is Hooker

A solution to analyze Android applications

Centralize and aggregate analysis of thousands of
different applications

Come in WE'RE

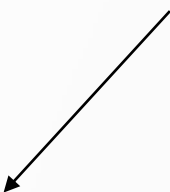
OPEN
SOURCE

How Hooker works

Microanalysis versus Macroanalysis

How Hooker works

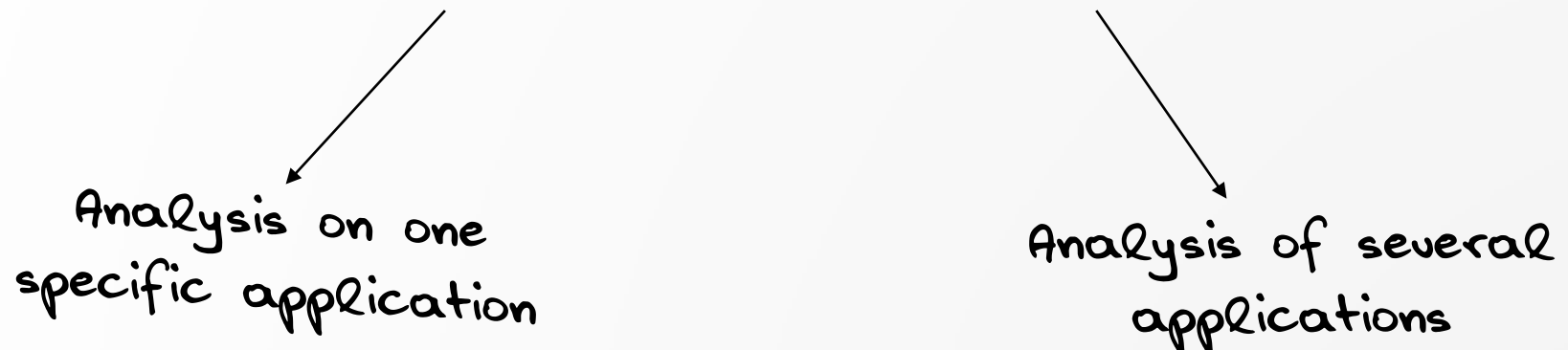
Microanalysis versus Macroanalysis



Analysis on one
specific application

How Hooker works

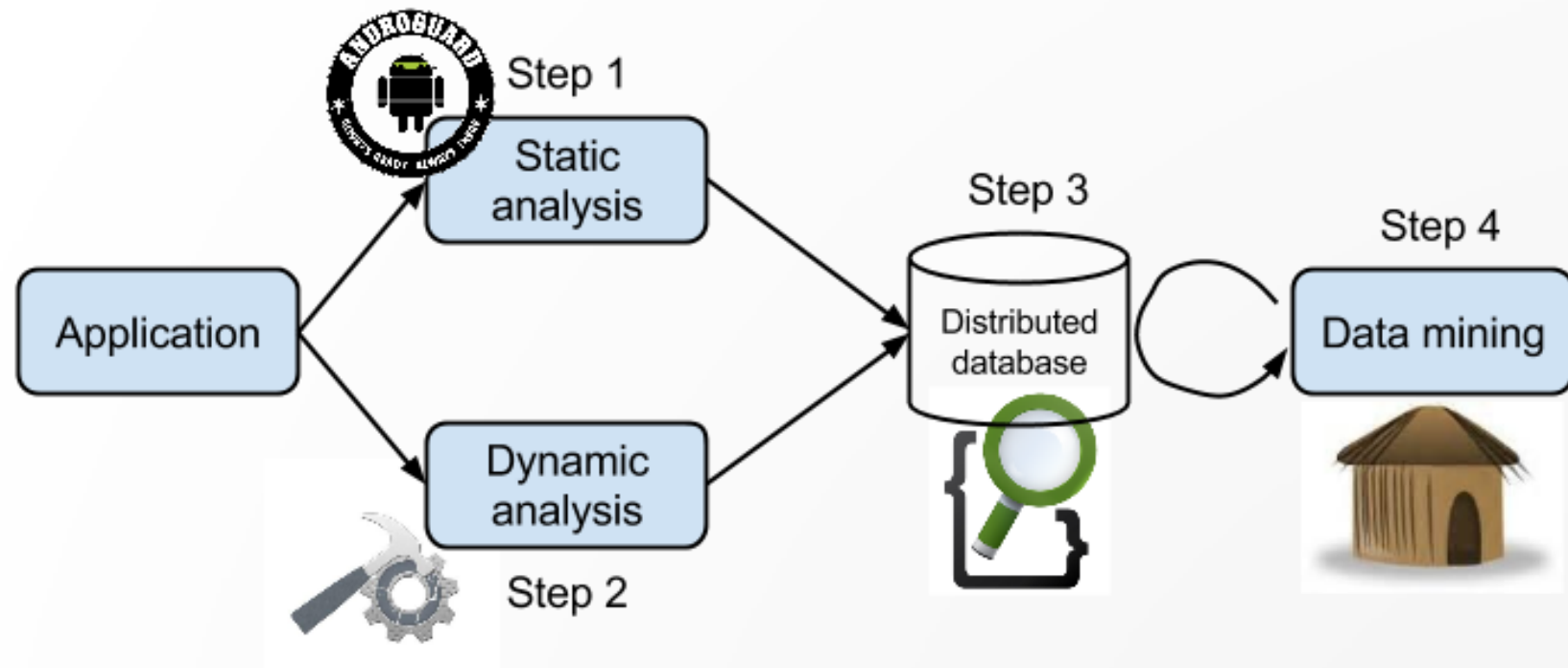
Microanalysis versus Macroanalysis



Analysis on one
specific application

Analysis of several
applications

Microanalysis overview



RULE N°1:

GATHER ALL POSSIBLE
INFORMATION ABOUT THE
APPLICATION BEHAVIOR



Step 1: Androguard

It just works great
Framework in python

Let us extract basic information about the application

Package name

Permissions

Services

Etc.



Step 2: Substrate

An API hooking framework

Changes behavior of one application, without patches, or specific ROM, or whatever

What you need is:

Root access

Compatible Android version

Substrate

Injects code into Zygote process (father of all processes)

Therefore, injected in all spawned processes

(Similar to Xposed)

Use Substrate to:

Hook access to personal information (read contacts, etc.)

Hook access to specific API (open socket)

Modify return of specific methods (anti-anti-emulation)

Hook PowerManager methods

```
159- /**
160   * Attach on PowerManager class.
161   */
162- private void attachOnPowerManagerClass() {
163     final String className = "android.os.PowerManager";
164
165     Map<String, Integer> methodsToHook = new HashMap<String, Integer>();
166
167     methodsToHook.put("goToSleep", 0);
168     methodsToHook.put("isScreenOn", 0);
169     methodsToHook.put("newWakeLock", 1);
170     methodsToHook.put("reboot", 1);
171     methodsToHook.put("userActivity", 1);
172     methodsToHook.put("wakeUp", 1);
173
174     try {
175         hookMethods(null, className, methodsToHook);
176         SubstrateMain.Log(new StringBuilder("hooking ").append(className)
177             .append(" methods sucessful").toString());
178
179     } catch (HookerInitializationException e) {
180         SubstrateMain.Log(new StringBuilder("hooking ").append(className)
181             .append(" methods has failed").toString(), e);
182     }
183 }
```

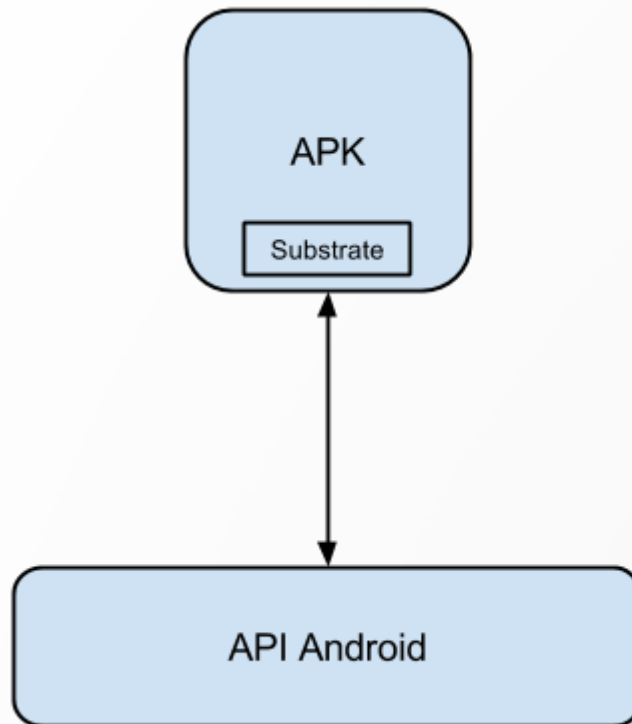
Hook PowerManager methods

```
159- /**
160   * Attach on PowerManager class.
161   */
162- private void attachOnPowerManagerClass() {
163     final String className = "android.os.PowerManager";
164
165     Map<String, Integer> methodsToHook = new HashMap<String, Integer>();
166
167     methodsToHook.put("goToSleep", 0);
168     methodsToHook.put("isScreenOn", 0);
169     methodsToHook.put("newWakeLock", 1);
170     methodsToHook.put("reboot", 1);
171     methodsToHook.put("userActivity", 1);
172     methodsToHook.put("wakeUp", 1);
173
174     try {
175         hookMethods(null, className, methodsToHook);
176         SubstrateMain.Log(new StringBuilder("hooking ").append(className)
177             .append(" methods sucessful").toString());
178
179     } catch (HookerInitializationException e) {
180         SubstrateMain.Log(new StringBuilder("hooking ").append(className)
181             .append(" methods has failed").toString(), e);
182     }
183 }
```

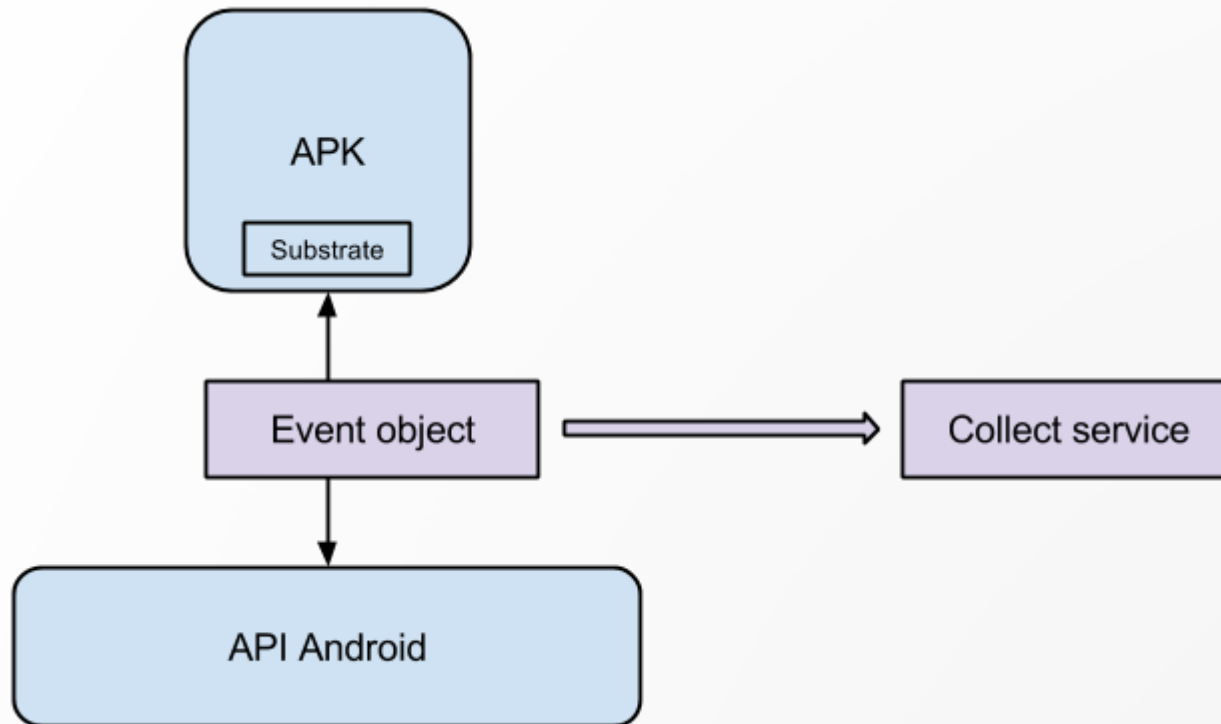
Class name

Methods name

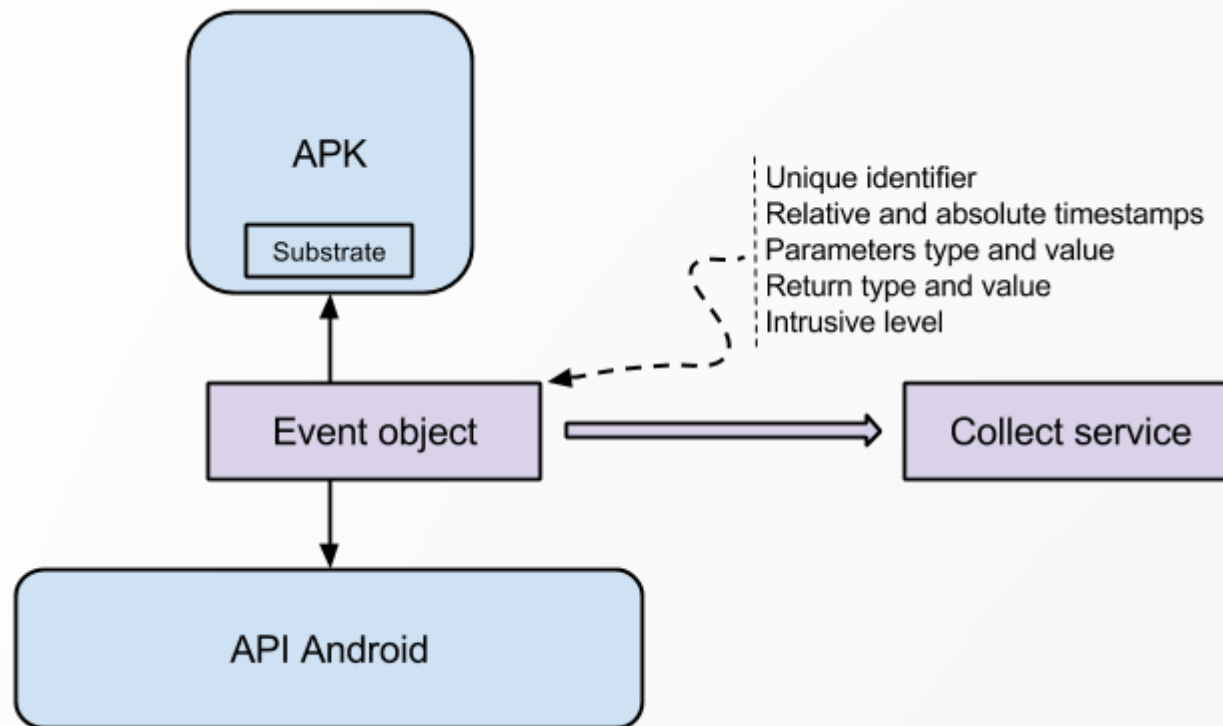
Build events in real time



Build events in real time



Build events in real time



Intrusive level indicator

Differentiates critical event from normal event

Writing is considered more intrusive than reading

Application doing lots of intrusive events is highlighted

RelativeTimestamp ^ ▸	◀ HookerName ▶	◀ PackageName ▶	◀ ClassName ▶	◀ MethodName ▶	◀ Return.ReturnValue
53274809	Crypto	com.melodis.midomiMusicIdentifier	javax.crypto.Cipher	init	

View: [Table](#) / [JSON](#) / [Raw](#)

Field	Action	Value
ClassName	🔍 🗑 📋	javax.crypto.Cipher
Data	🔍 🗑 📋	
HookerName	🔍 🗑 📋	Crypto
InstanceID	🔍 🗑 📋	1094434584
IntrusiveLevel	🔍 🗑 📋	0
MethodName	🔍 🗑 📋	init
PackageName	🔍 🗑 📋	com.melodis.midomiMusicIdentifier
Parameters	🔍 🗑 📋	{"ParameterType":"java.lang.Integer","ParameterValue":"1"}, {"ParameterType":"javax.crypto.spec.SecretKeySpec","ParameterValue":"SecretKeySpec[algorithm=AES/ECB/PKCS7Padding,key={101,53,98,49,102,56,52,102,101,49,49,48,52,100,99,52,97,97,102,99,52,99,102,54,99,102,100,102,48,99,50,56]}"]}
RelativeTimestamp	🔍 🗑 📋	53274809
Timestamp	🔍 🗑 📋	1397378485227
_id	🔍 🗑 📋	wttEjbPRTsKJKUmaHc4JSQ
_index	🔍 🗑 📋	hooker_test
_type	🔍 🗑 📋	event

Class name



RelativeTimestamp ^	HookerName	PackageName	ClassName	MethodName	Return.ReturnValue
53274809	Crypto	com.melodis.midomiMusicIdentifier	javax.crypto.Cipher	init	

View: [Table](#) / [JSON](#) / [Raw](#)

Field	Action	Value
ClassName		javax.crypto.Cipher
Data		
HookerName		Crypto
InstanceID		1094434584
IntrusiveLevel		0
MethodName		init
PackageName		com.melodis.midomiMusicIdentifier
Parameters		{"ParameterType":"java.lang.Integer","ParameterValue":"1"}, {"ParameterType":"javax.crypto.spec.SecretKeySpec","ParameterValue":"SecretKeySpec[algorithm=AES/ECB/PKCS7Padding,key={101,53,98,49,102,56,52,102,101,49,49,48,52,100,99,52,97,97,102,99,52,99,102,54,99,102,100,102,48,99,50,56]}"]}
RelativeTimestamp		53274809
Timestamp		1397378485227
_id		wttEjbPRTsKJKUmaHc4JSQ
_index		hooker_test
_type		event

Class name

Method name

Method name

RelativeTimestamp ^	HookerName	PackageName	ClassName	MethodName	Return.ReturnValue
53274809	Crypto	com.melodis.midomiMusicIdentifier	javax.crypto.Cipher	init	
View: Table / JSON / Raw					
Field	Action	Value			
ClassName	Q 0	javax.crypto.Cipher			
Data	Q 0				
HookerName	Q 0	Crypto			
InstanceID	Q 0	1094434584			
IntrusiveLevel	Q 0	0			
MethodName	Q 0	init			
PackageName	Q 0	com.melodis.midomiMusicIdentifier			
Parameters	Q 0	{"ParameterType":"java.lang.Integer","ParameterValue":"1"}, {"ParameterType":"javax.crypto.spec.SecretKeySpec","ParameterValue":"SecretKeySpec[algorithm=AES/ECB/PKCS7Padding,key={101,53,98,49,102,56,52,102,101,49,49,48,52,100,99,52,97,97,102,99,52,99,102,54,99,102,100,102,48,99,50,56]}"]}			
RelativeTimestamp	Q 0	53274809			
Timestamp	Q 0	1397378485227			
_id	Q 0	wtEjbPRTsKJKUmaHc4JSQ			
_index	Q 0	hooker_test			
_type	Q 0	event			

Class name

Method name

Method name

Parameters types
and values

RelativeTimestamp ^	HookerName	PackageName	ClassName	MethodName	Return.ReturnValue
53274809	Crypto	com.melodis.midomiMusicIdentifier	javax.crypto.Cipher	init	
View: Table / JSON / Raw					
Field	Action	Value			
ClassName	Q 0	javax.crypto.Cipher			
Data	Q 0				
HookerName	Q 0	Crypto			
InstanceID	Q 0	1094434584			
IntrusiveLevel	Q 0	0			
MethodName	Q 0	init			
PackageName	Q 0	com.melodis.midomiMusicIdentifier			
Parameters	Q 0	{"ParameterType":"java.lang.Integer","ParameterValue":"1"}, {"ParameterType":"javax.crypto.spec.SecretKeySpec","ParameterValue":"SecretKeySpec[algorithm=AES/ECB/PKCS7Padding,key={101,53,98,49,102,56,52,102,101,49,49,48,52,100,99,52,97,97,102,99,52,99,102,54,99,102,100,102,48,99,50,56]}"]}			
RelativeTimestamp	Q 0	53274809			
Timestamp	Q 0	1397378485227			
_id	Q 0	wttEjbPRTsKJKUmaHc4JSQ			
_index	Q 0	hooker_test			
_type	Q 0	event			

Main limitation

White list enumeration

We don't intercept what we don't declare

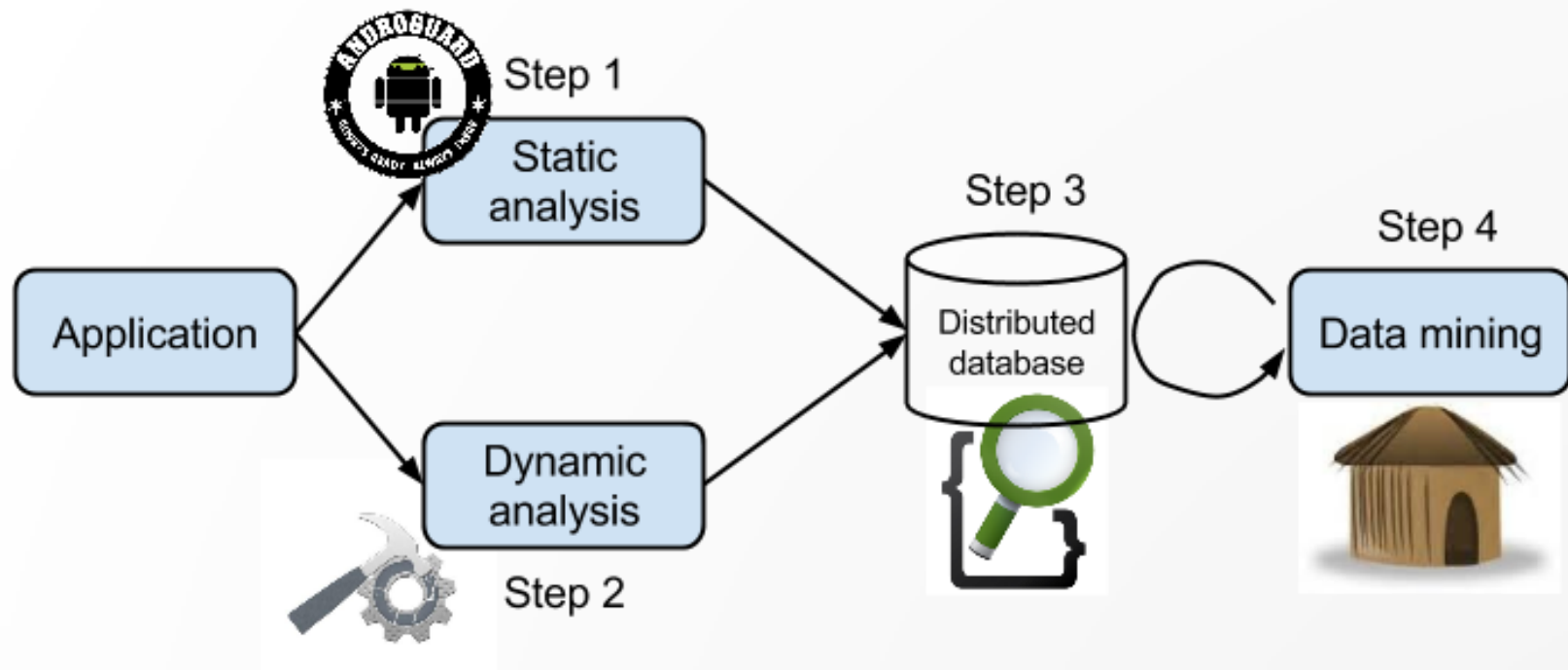
Main limitation

White list enumeration

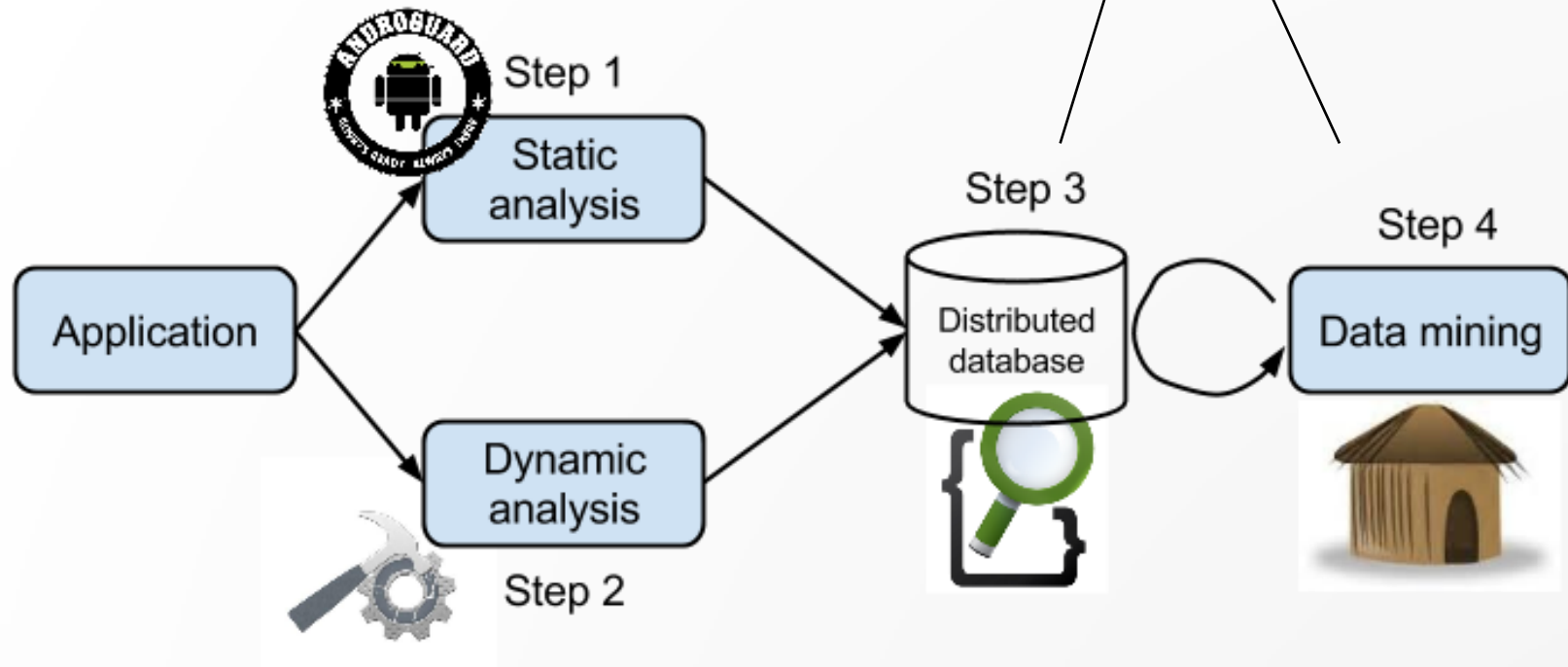
We don't intercept what we don't declare



Test on malware if we intercept
everything we want



Step 3 and 4: Collects and centralizes results for data mining



Store events in a distributed database

Elastic search

Interact with database

Kibana (front-end)

Example of Kibana web interface



You have to build your own Kibana interface

Basic malware generates 2000 events in 60 seconds

Macroanalysis

Macroanalysis

Automation and **parallelization** of microanalysis

Macroanalysis

Automation and **parallelization** of microanalysis

Look for specific patterns in thousands of applications

Macroanalysis

Automation and **parallelization** of microanalysis

Look for specific patterns in thousands of applications

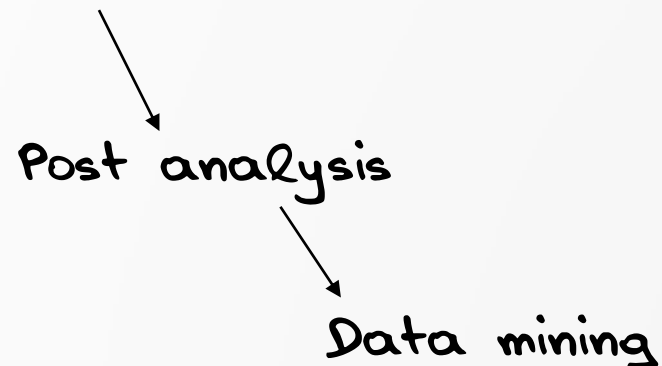


Post analysis

Macroanalysis

Automation and **parallelization** of microanalysis

Look for specific patterns in thousands of applications



Automation

Step 1: Prepare an Android emulator

Step 2: Configure a scenario

Install

Execute

Stimulate

External stimulation

Reboot

Automation

Step 1: Prepare an Android emulator

Step 2: Configure a scenario

Install

Execute

Stimulate

External stimulation

Reboot

Be closest as possible as the user

Automation

Step 1: Prepare an Android emulator

Step 2: Configure a scenario

Install

Execute

Stimulate

Monkey

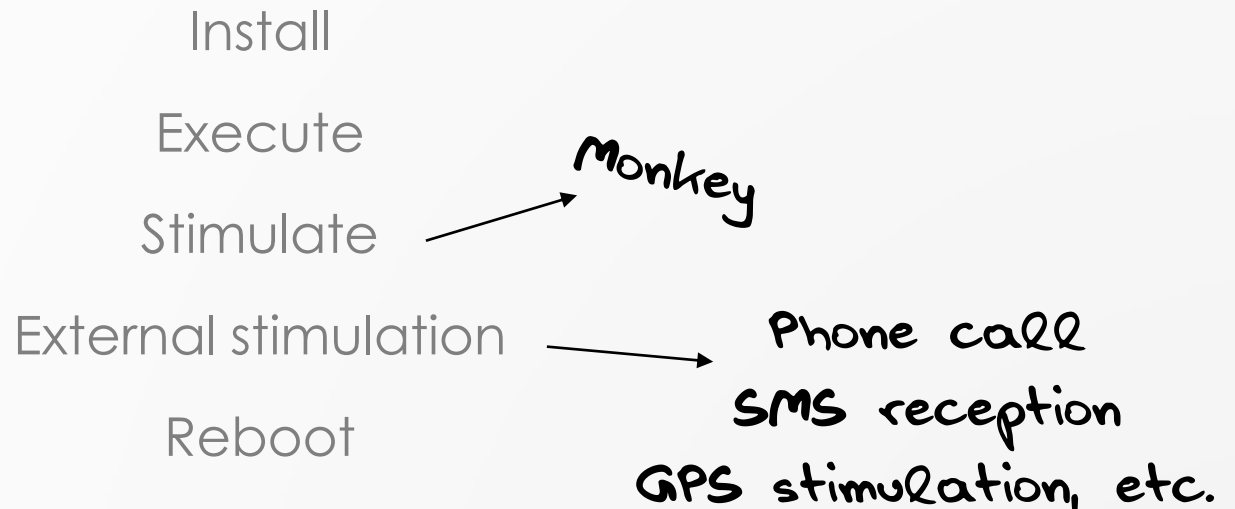
External stimulation

Reboot

Automation

Step 1: Prepare an Android emulator

Step 2: Configure a scenario



Step 3: Run the experiment

```
$ python hooker_xp.py -c automaticAnalysis.conf
```

Wait and see

Post-analysis

Python script to query Elasticsearch database

Query what you want to make:

- Statistics
- Highlights



Get thousands of APKs

Google store
Unofficial markets
APK in archives



Get thousands of APKs

Google store

Unofficial markets

APK in archives

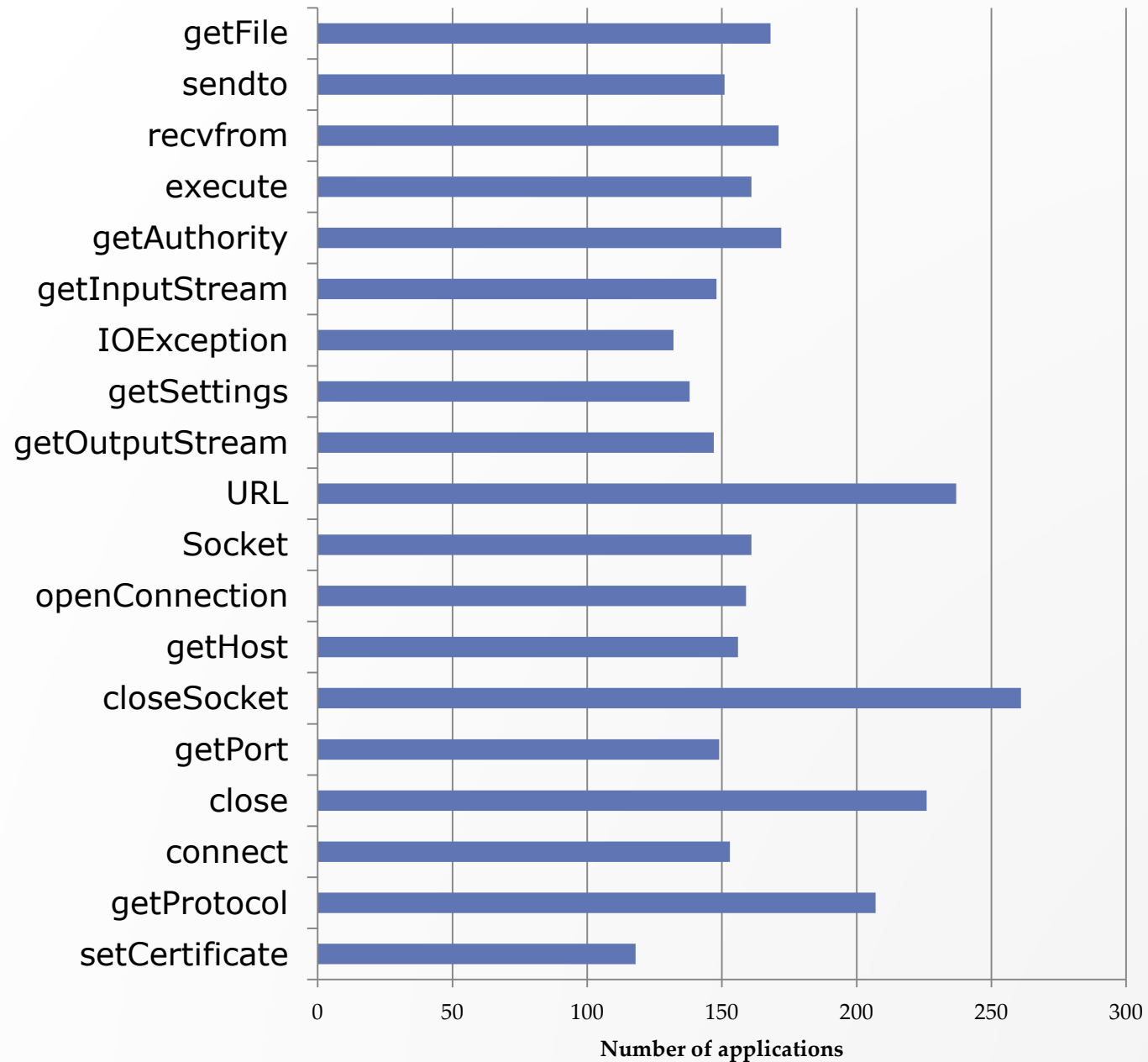
What we have tried until now:

1000 apps from SlideMe market in the paper

1000 apps from Google store

Network statistics

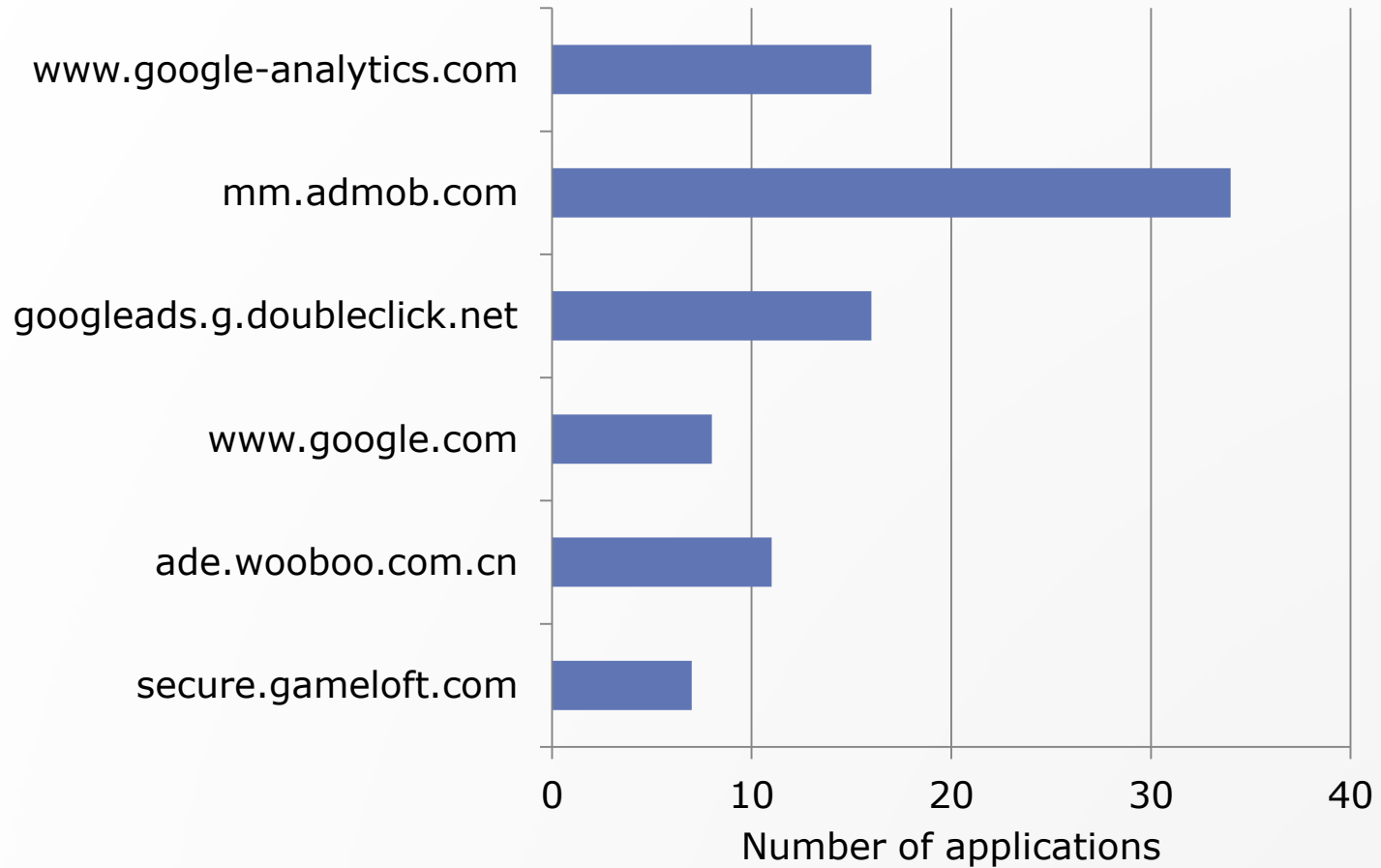
Most used Network methods



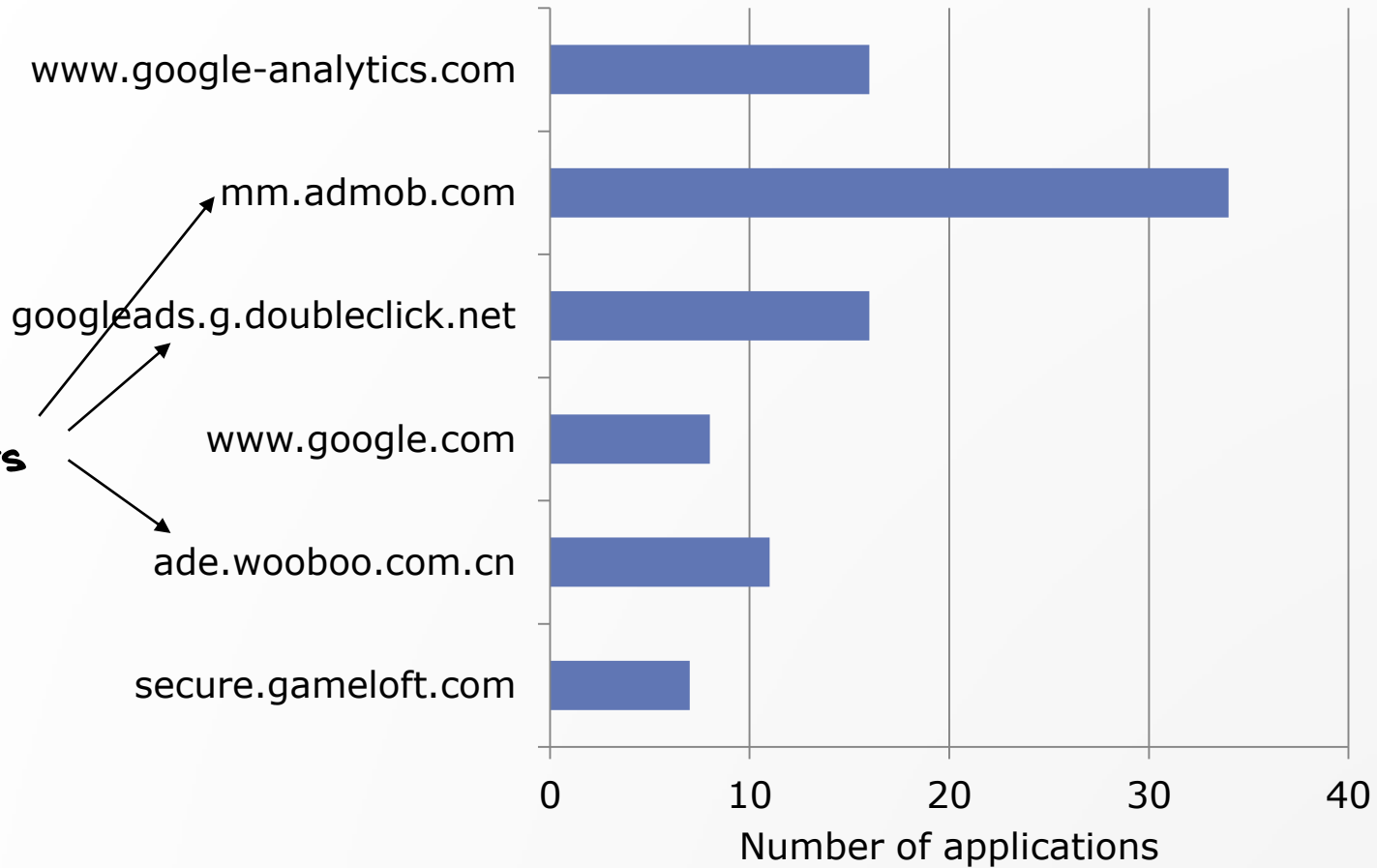
Internet permissions

477 apps asking for internet permissions
404 have been found using it

Domains most accessed

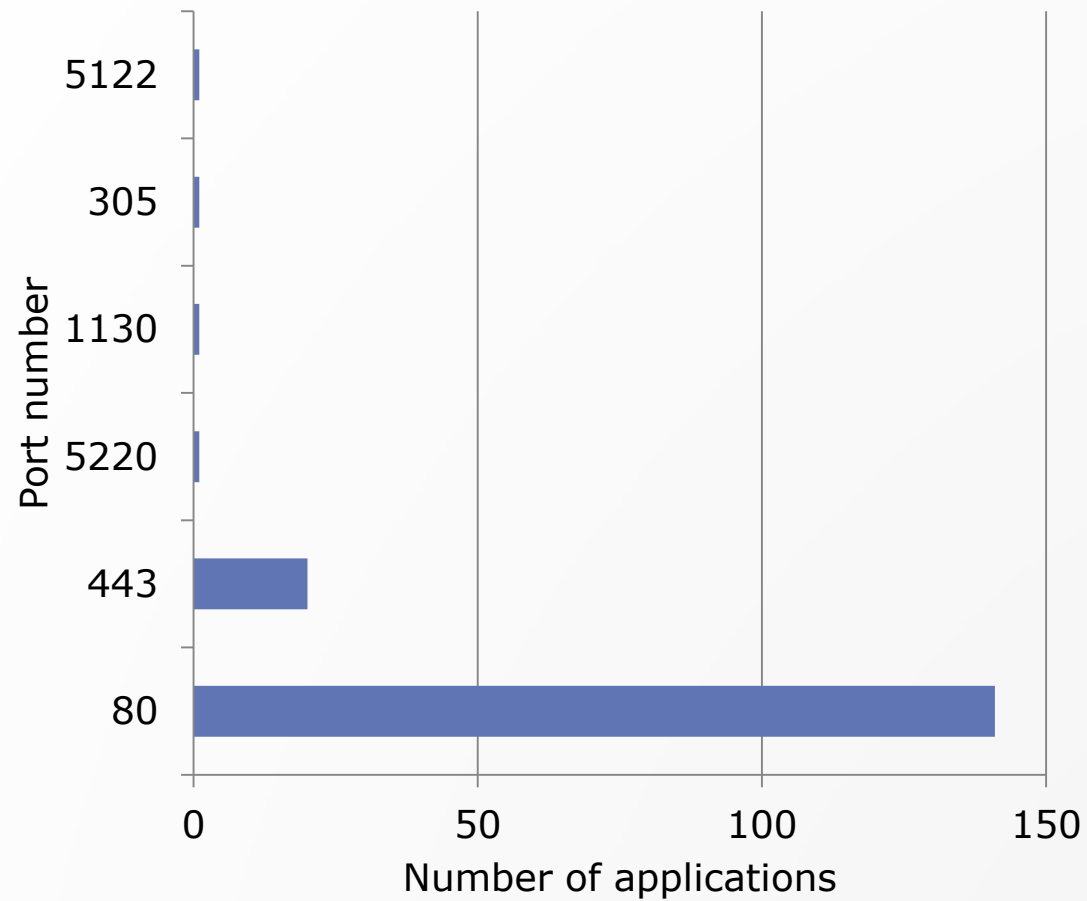


Domains most accessed



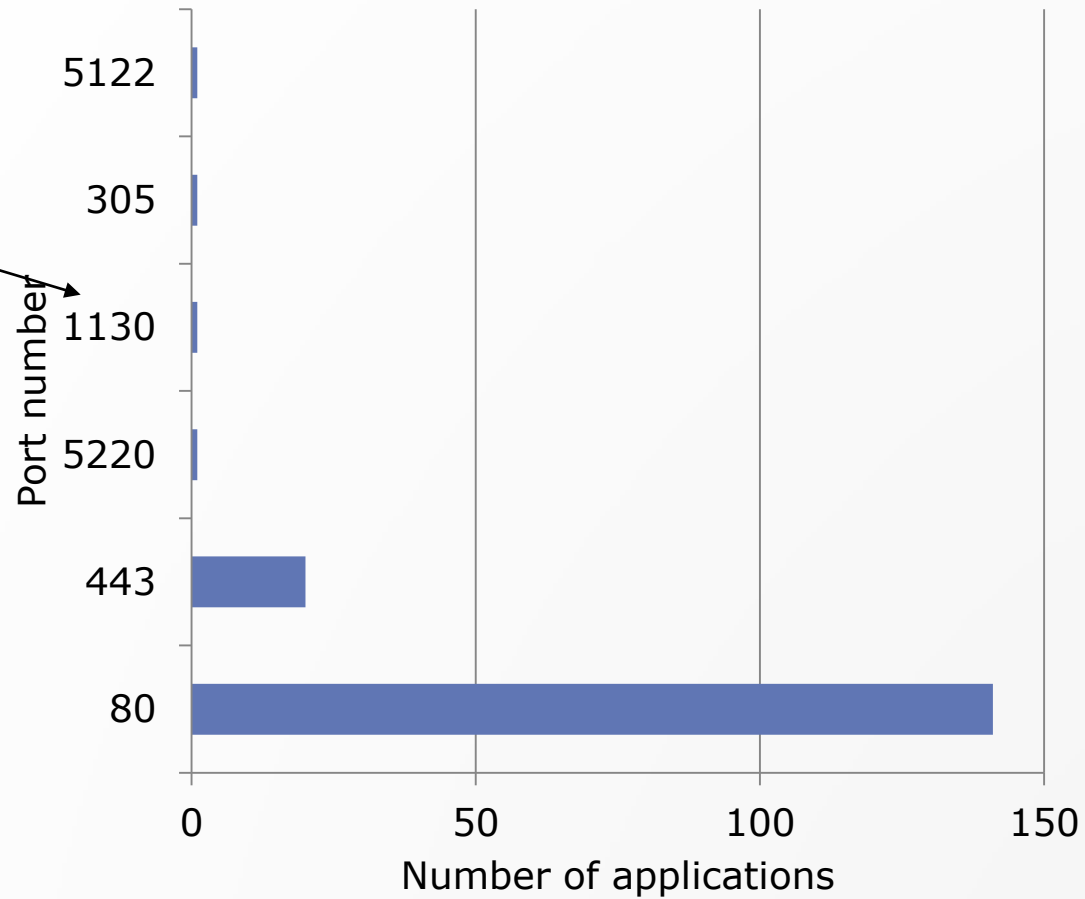
Advertisements

Port number accessed by applications



Port number accessed by applications

Noknok trojan?

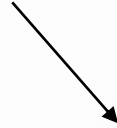




Wanna find some vulnerable apps?

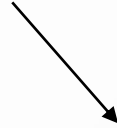
WebView and addJavaScriptInterface

WebView and
addJavaScriptInterface



Interface to call Java
from javascript

WebView and
addJavaScriptInterface



Interface to call Java
from javascript

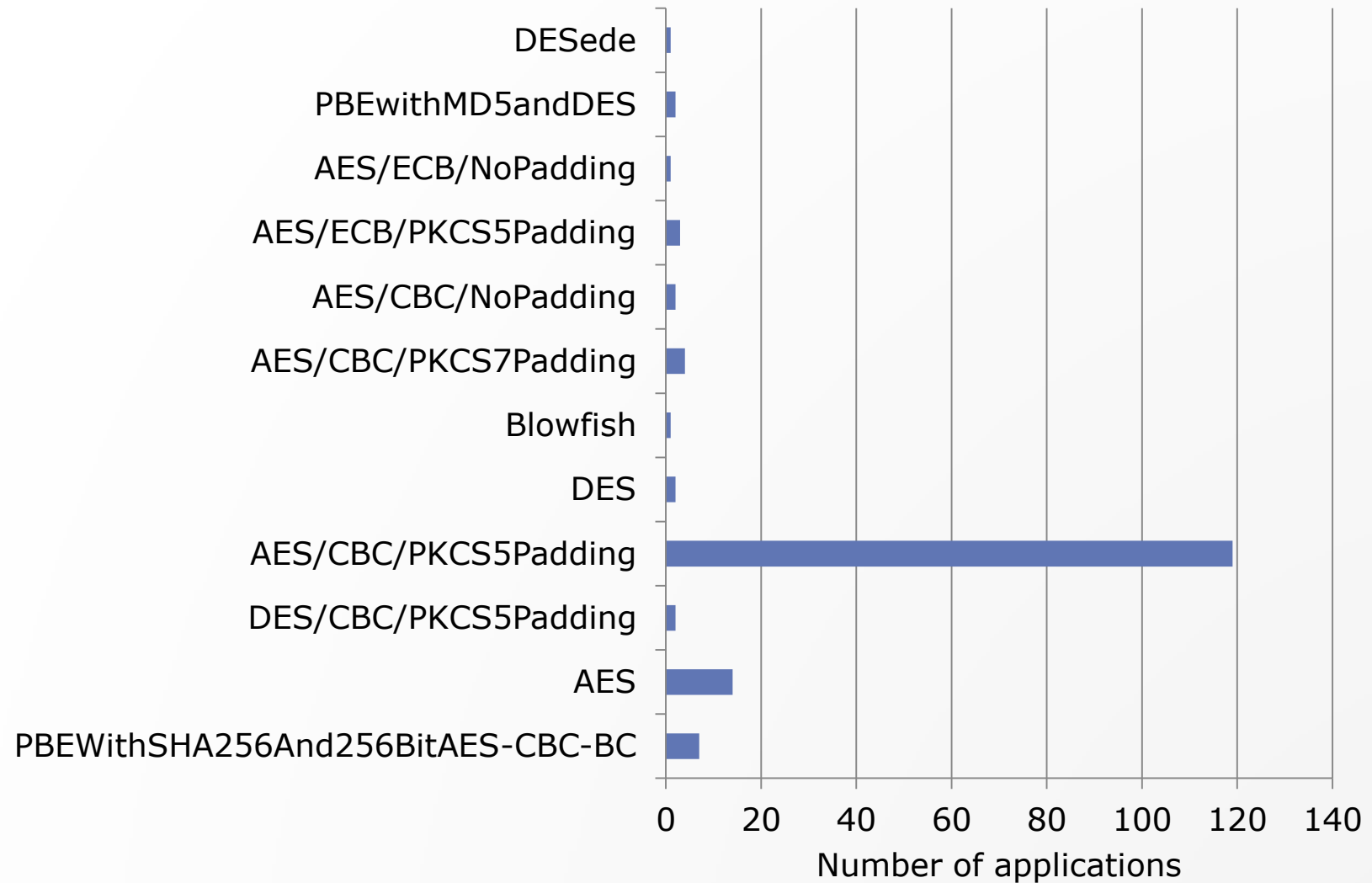


Remote code execution

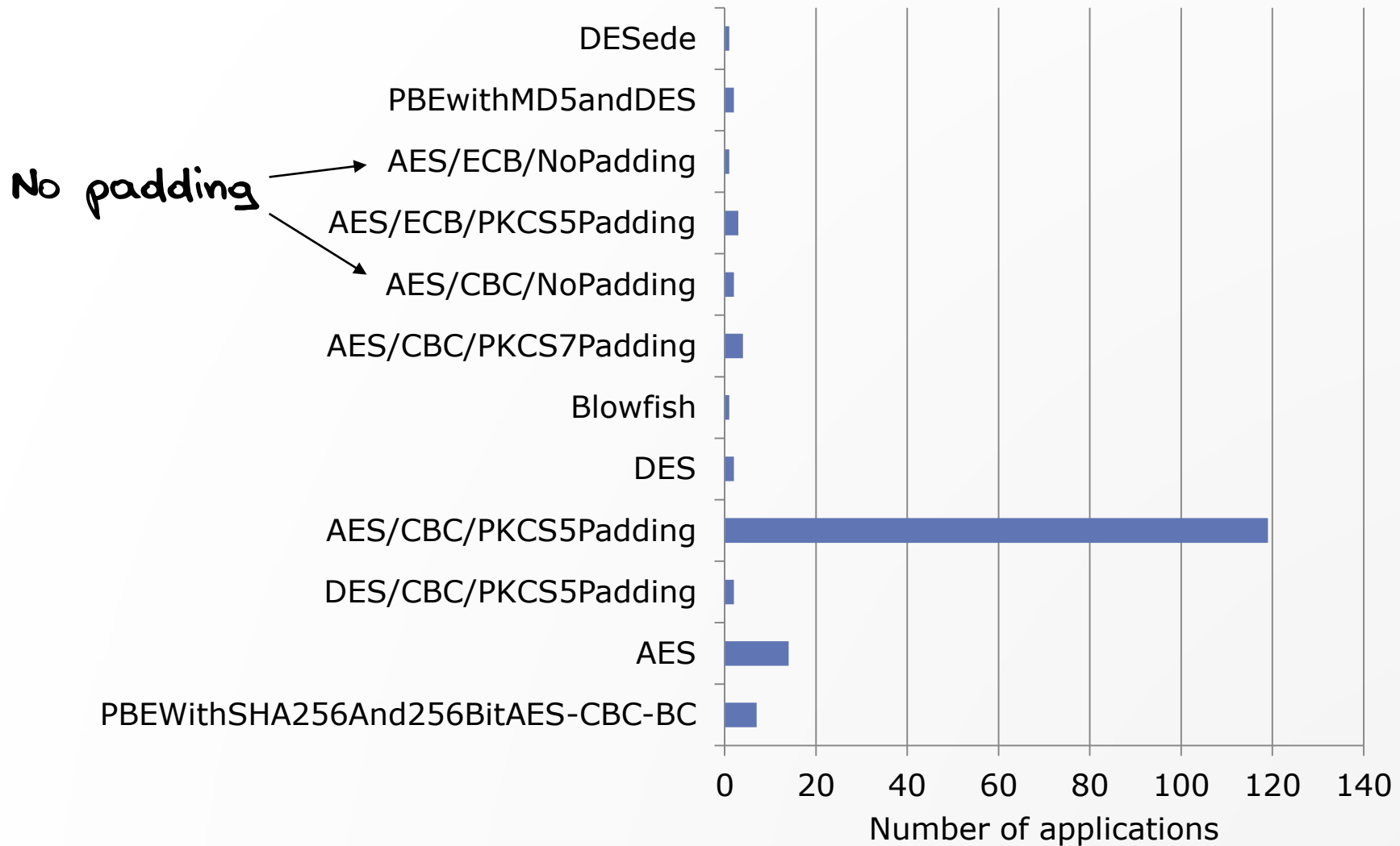
On 1000 applications from Google store
23 apps using `addJavaScriptInterface` method

Crypto statistics

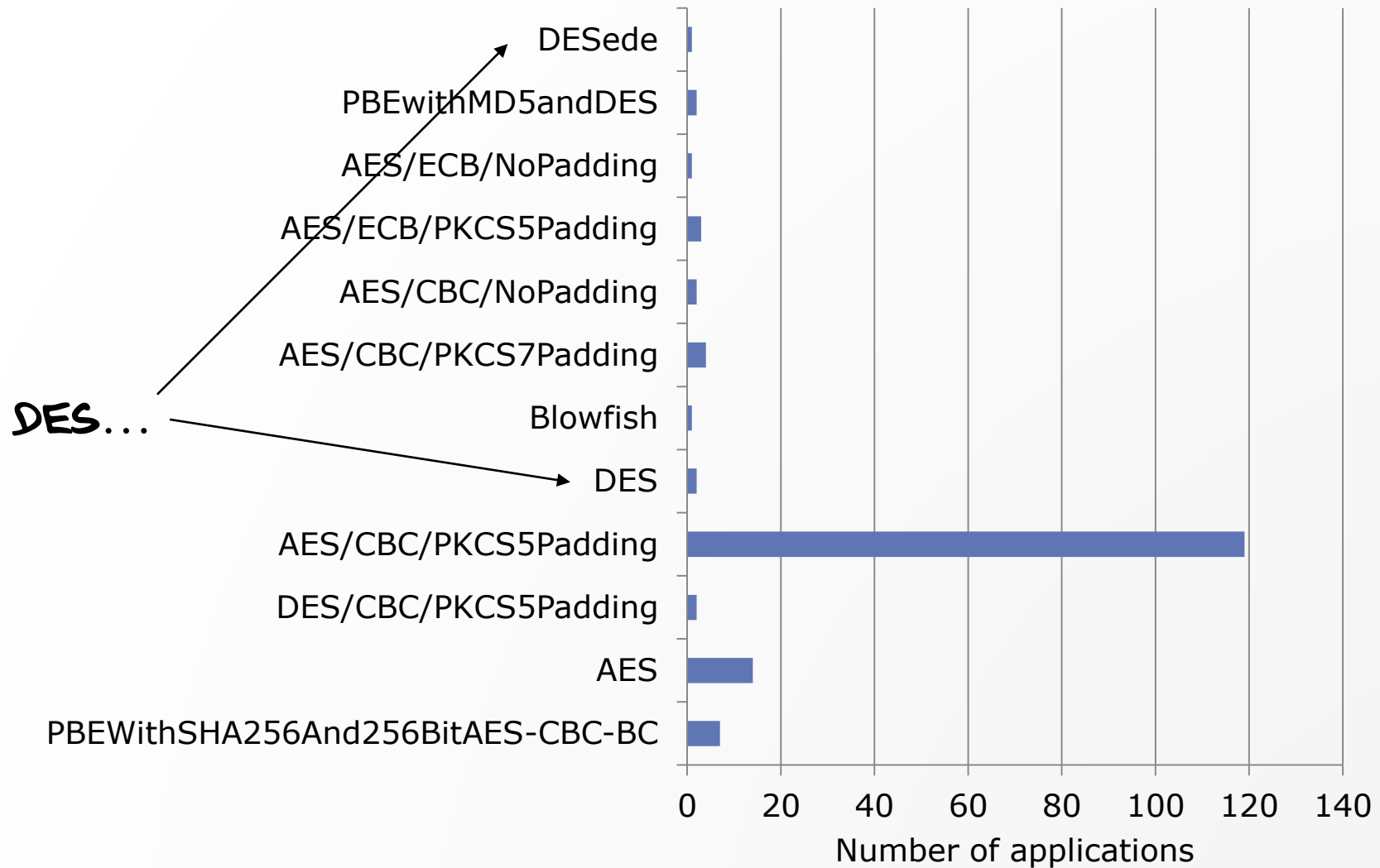
Use of cypher functions



Use of cypher functions



Use of cypher functions

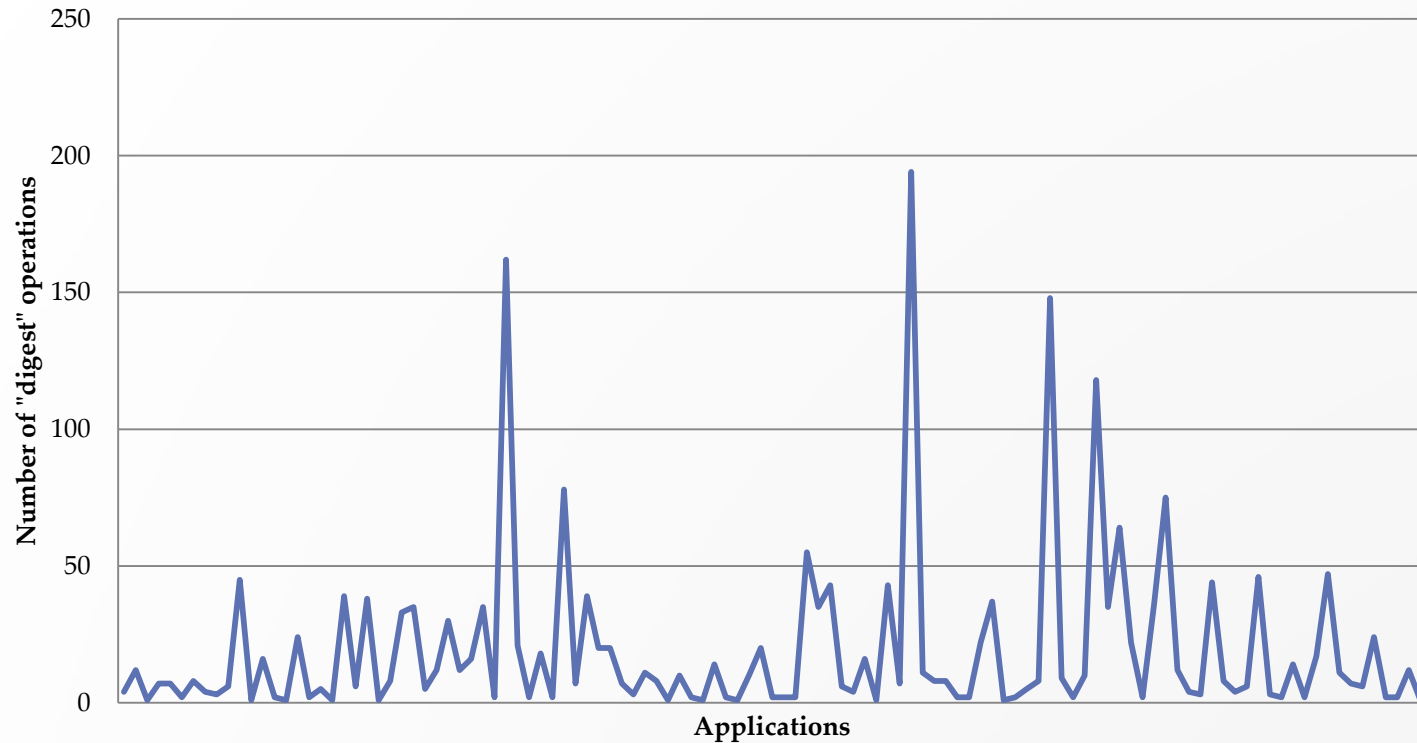


Bitcoin miners

Bitcoin miners → « Several apps from the GPlay are infected by crypto miners »

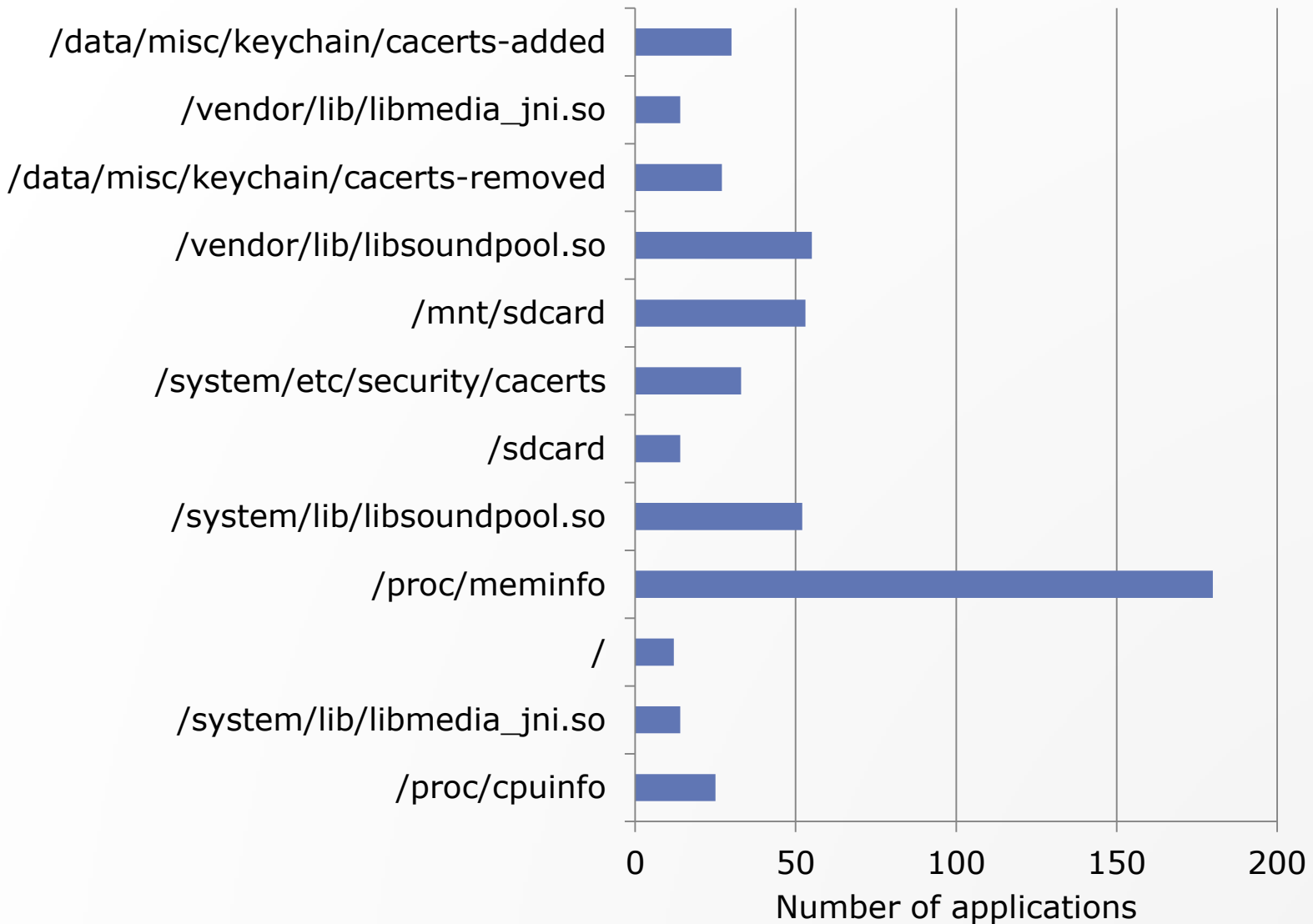
« Several apps from the GPlay are
Bitcoin miners → infected by crypto miners »

↓
Crypto hashing abuses

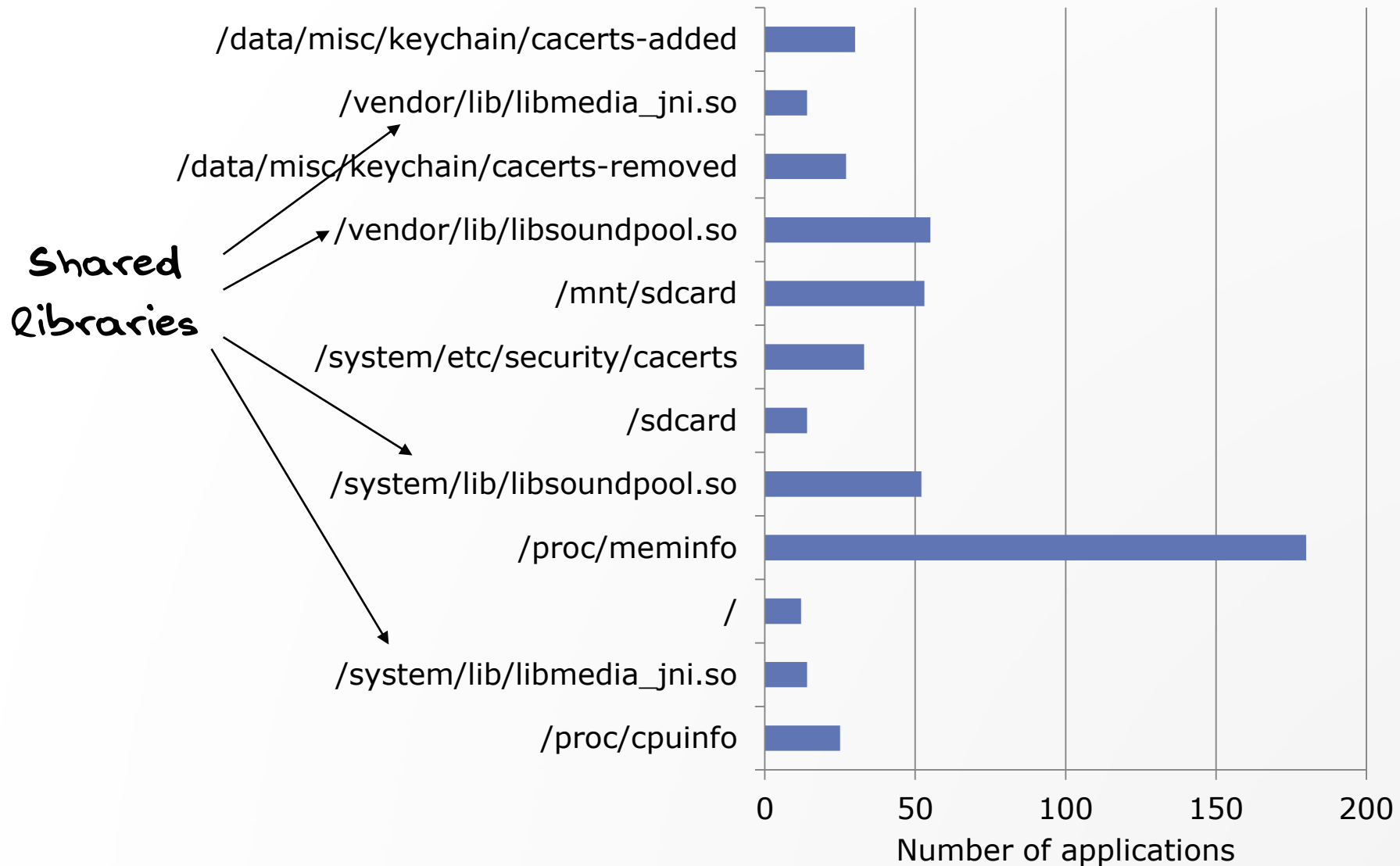


File statistics

Files accessed by application other than their /data/

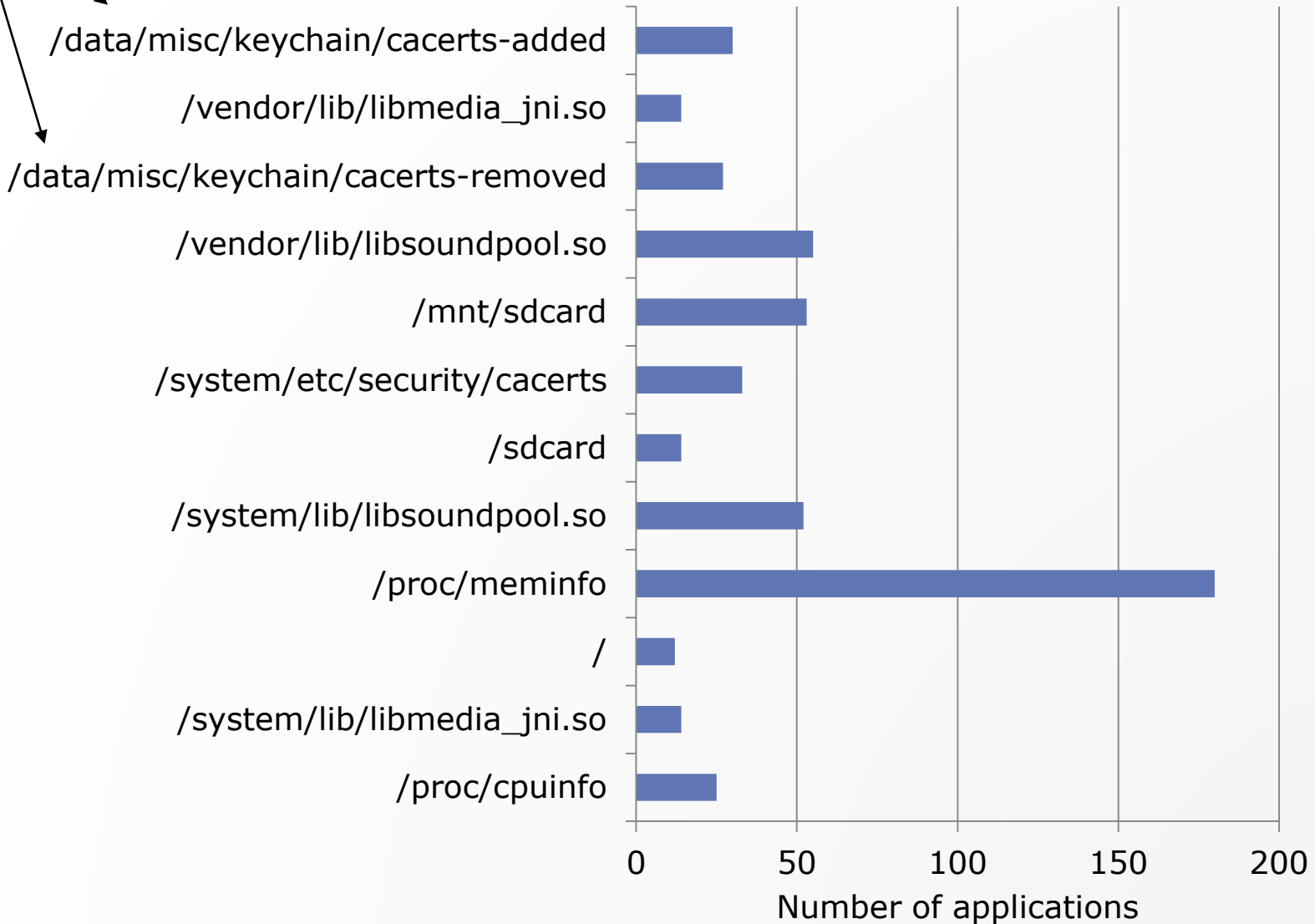


Files accessed by application other than their /data/



Certificates

Files accessed by application other than their /data/



File accesses are illustrating application behavior...

```
543 [MainProcess/MainThread/INFO] 31798: 7c358ec3a35f2eb3034f12ba3e76b613 accessing files:
544 [MainProcess/MainThread/INFO] 31799: /data/app/WidgetPreview.apk
545 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@WidgetPreview.apk@classes.dex
546 [MainProcess/MainThread/INFO] 31799: /data/app/SmokeTestApp.apk
547 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@SmokeTestApp.apk@classes.dex
548 [MainProcess/MainThread/INFO] 31799: /data/app/SmokeTest.apk
549 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@SmokeTest.apk@classes.dex
550 [MainProcess/MainThread/INFO] 31799: /data/app/com.amosys.hooker-2.apk
551 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.amosys.hooker-2.apk@classes.dex
552 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.amosys.hooker.generatecontacts-1.apk@classes.dex
553 [MainProcess/MainThread/INFO] 31799: /data/app/GestureBuilder.apk
554 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@GestureBuilder.apk@classes.dex
555 [MainProcess/MainThread/INFO] 31799: /data/app/com.amosys.hooker.generatecontacts-1.apk
556 [MainProcess/MainThread/INFO] 31799: /data/app/SoftKeyboard.apk
557 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@SoftKeyboard.apk@classes.dex
558 [MainProcess/MainThread/INFO] 31799: /data/app/ApiDemos.apk
559 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@ApiDemos.apk@classes.dex
560 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.saurik.substrate-1.apk@classes.dex
561 [MainProcess/MainThread/INFO] 31799: /data/app/CubeLiveWallpapers.apk
562 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@CubeLiveWallpapers.apk@classes.dex
563 [MainProcess/MainThread/INFO] 31799: /data/app/com.noshufou.android.su-1.apk
564 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.noshufou.android.su-1.apk@classes.dex
565 [MainProcess/MainThread/INFO] 31799: /data/app/com.saurik.substrate-1.apk
566 [MainProcess/MainThread/INFO] 31799: /sdcard/backups/apps
```

Backup app

```
543 [MainProcess/MainThread/INFO] 31798: 7c358ec3a35f2eb3034f12ba3e76b613 accessing files:
544 [MainProcess/MainThread/INFO] 31799: /data/app/WidgetPreview.apk
545 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@WidgetPreview.apk@classes.dex
546 [MainProcess/MainThread/INFO] 31799: /data/app/SmokeTestApp.apk
547 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@SmokeTestApp.apk@classes.dex
548 [MainProcess/MainThread/INFO] 31799: /data/app/SmokeTest.apk
549 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@SmokeTest.apk@classes.dex
550 [MainProcess/MainThread/INFO] 31799: /data/app/com.amossys.hooker-2.apk
551 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.amossys.hooker-2.apk@classes.dex
552 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.amossys.hooker.generatecontacts-1.apk@classes.dex
553 [MainProcess/MainThread/INFO] 31799: /data/app/GestureBuilder.apk
554 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@GestureBuilder.apk@classes.dex
555 [MainProcess/MainThread/INFO] 31799: /data/app/com.amossys.hooker.generatecontacts-1.apk
556 [MainProcess/MainThread/INFO] 31799: /data/app/SoftKeyboard.apk
557 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@SoftKeyboard.apk@classes.dex
558 [MainProcess/MainThread/INFO] 31799: /data/app/ApiDemos.apk
559 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@ApiDemos.apk@classes.dex
560 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.saurik.substrate-1.apk@classes.dex
561 [MainProcess/MainThread/INFO] 31799: /data/app/CubeLiveWallpapers.apk
562 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@CubeLiveWallpapers.apk@classes.dex
563 [MainProcess/MainThread/INFO] 31799: /data/app/com.noshufou.android.su-1.apk
564 [MainProcess/MainThread/INFO] 31799: /data/dalvik-cache/data@app@com.noshufou.android.su-1.apk@classes.dex
565 [MainProcess/MainThread/INFO] 31799: /data/app/com.saurik.substrate-1.apk
566 [MainProcess/MainThread/INFO] 31799: /sdcard/backups/apps
```

How to be sure to find an sdcard...

```
569 [MainProcess/MainThread/INFO] 31781: 978f317d7af23a6af0bdc295aa8c8b67 accessing files:
570 [MainProcess/MainThread/INFO] 31781: /mnt/card
571 [MainProcess/MainThread/INFO] 31781: /card
572 [MainProcess/MainThread/INFO] 31781: /proc/meminfo
573 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard
574 [MainProcess/MainThread/INFO] 31781: /mnt/extSdCard
575 [MainProcess/MainThread/INFO] 31781: /etc/vold.fstab
576 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/
577 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/ext_sd
578 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/_external_sd
579 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/sd
580 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/emmc
581 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/Locus/
582 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/external_sd
583 [MainProcess/MainThread/INFO] 31781: /mnt/sdcard/external-sd
584 [MainProcess/MainThread/INFO] 31781: /mnt/emmc
585 [MainProcess/MainThread/INFO] 31781: /mnt/external1
586 [MainProcess/MainThread/INFO] 31781: /removable/microsd
587 [MainProcess/MainThread/INFO] 31781: /mnt/emms
```

you have to trust this app...

```
729 [MainProcess/MainThread/INFO] 31469: /system/xbin/iptables
730 [MainProcess/MainThread/INFO] 31469: /system/xbin/su
731 [MainProcess/MainThread/INFO] 31469: /system/bin/iptables
732
```

That's weird right?

```
734 [MainProcess/MainThread/INFO] 31227: 4a7c8f5fc0b82a3c72dee4c21e62a4df accessing files:
735 [MainProcess/MainThread/INFO] 31227: /data/local/tmp
736 [MainProcess/MainThread/INFO] 31227: /data/local/bin/su
737 [MainProcess/MainThread/INFO] 31227: /system/sd/xbinsu
738 [MainProcess/MainThread/INFO] 31227: /data/local/xbinsu
739 [MainProcess/MainThread/INFO] 31227: /system/xbinsu
740 [MainProcess/MainThread/INFO] 31227: /system/bin/su
741 [MainProcess/MainThread/INFO] 31227: /sbin/su
742 [MainProcess/MainThread/INFO] 31227: /data/local/su
743 [MainProcess/MainThread/INFO] 31227: /system/bin/failsafe/su
744
```

That's weird right?

```
734 [MainProcess/MainThread/INFO] 31227: 4a7c8f5fc0b82a3c72dee4c21e62a4df accessing files:
735 [MainProcess/MainThread/INFO] 31227: /data/local/tmp
736 [MainProcess/MainThread/INFO] 31227: /data/local/bin/su
737 [MainProcess/MainThread/INFO] 31227: /system/sd/xbinsu
738 [MainProcess/MainThread/INFO] 31227: /data/local/xbinsu
739 [MainProcess/MainThread/INFO] 31227: /system/xbinsu
740 [MainProcess/MainThread/INFO] 31227: /system/bin/su
741 [MainProcess/MainThread/INFO] 31227: /sbin/su
742 [MainProcess/MainThread/INFO] 31227: /data/local/su
743 [MainProcess/MainThread/INFO] 31227: /system/bin/failsafe/su
744
```

Is this app legitimate?

Hooker has a lot more capabilities
You chose to extract what you want



Hightlight **weaknesses** in application

Hightlight **malwares** within thousands of applications

Hightlight **WTF** behavior on your system



Hightlight **weaknesses** in application

Hightlight **malwares** within thousands of applications

Hightlight **WTF** behavior on your system

Give it a try, play hooker now:

<https://github.com/AndroidHooker>

Questions



Play hooker now:

<https://github.com/AndroidHooker>