Studying Android in the Age of Restricted Transparency

Ivan Baheux, Inria-Murena CIFRE PhD student

Vincent Roca (Inria)
Mathieu Cunche (Inria, Insa Lyon)
Gaël Duval (Murena)

Who Am I ?

« Me » - Source : My phone

Ivan Baheux

3rd year PhD at PRIVATICS
CIFRE collaboration with MURENA
Cyber security background

I like :

CTF (Cyber security challenges)
Role playing games
Marxism

My PhD

What affects regulators and researchers when trying to test Android Apps

More specifically, how can apps or systems limit large scale testing

My PhD

What affects regulators and researchers when trying to test Android Apps

More specifically, how can apps or systems limit large scale testing

Why Android ?

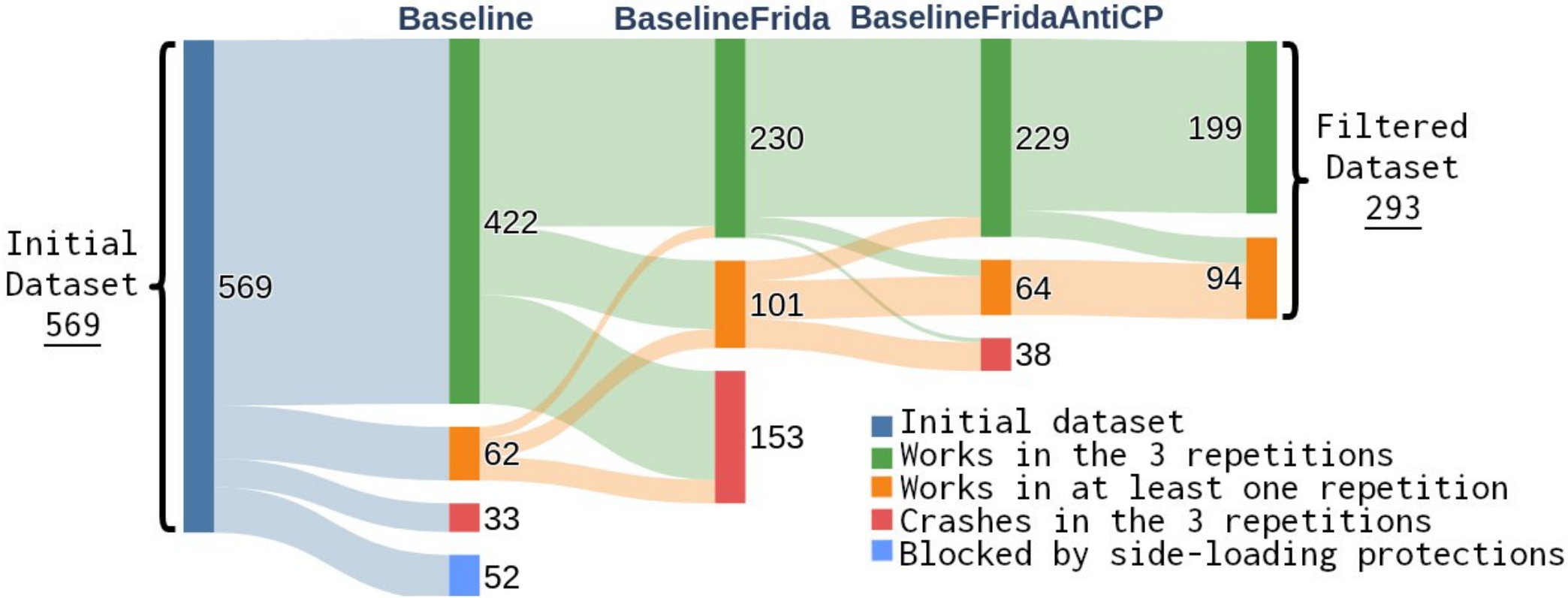
- *Murena !*
- *60% of global internet usage is on mobile devices*
- *Android has 72% market share*
- *Lacks proper privacy protection tools*

My PhD journey

From a large-scale study of Android Apps, we have encountered the reality of this ecosystem :

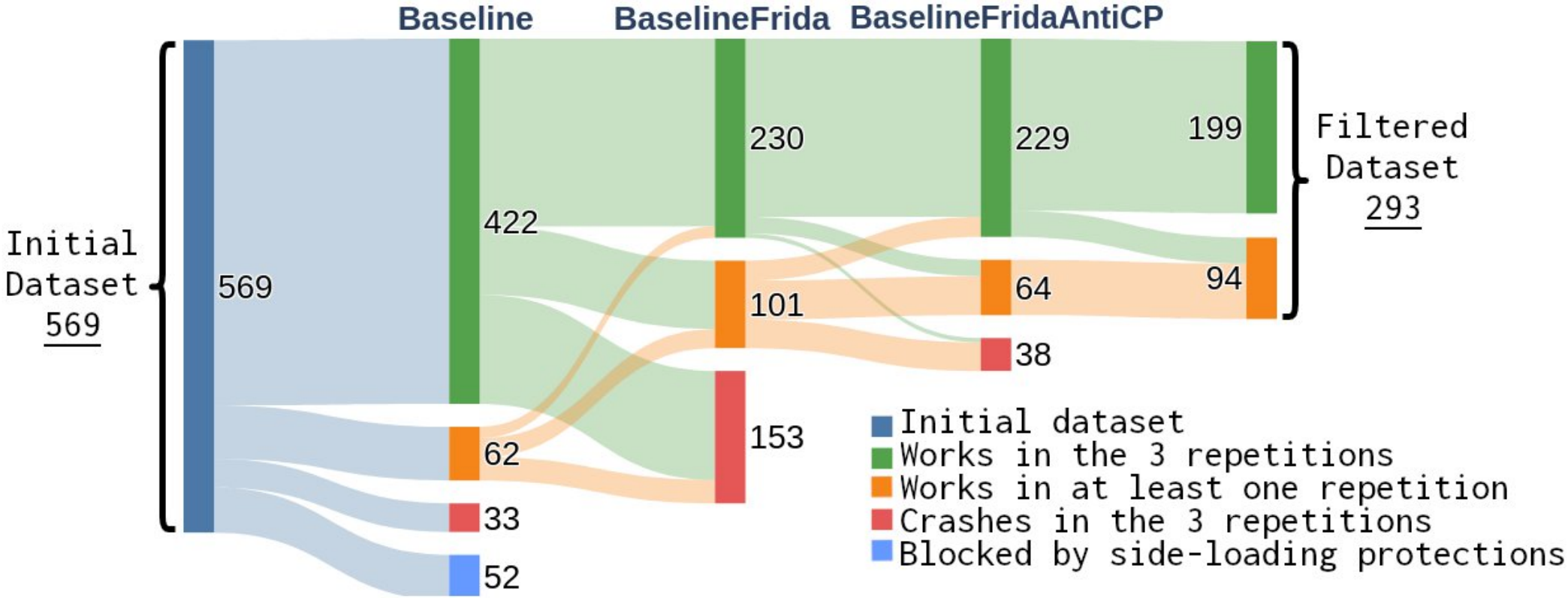
My PhD journey

From a large-scale study of Android Apps, we have encountered the reality of this ecosystem :



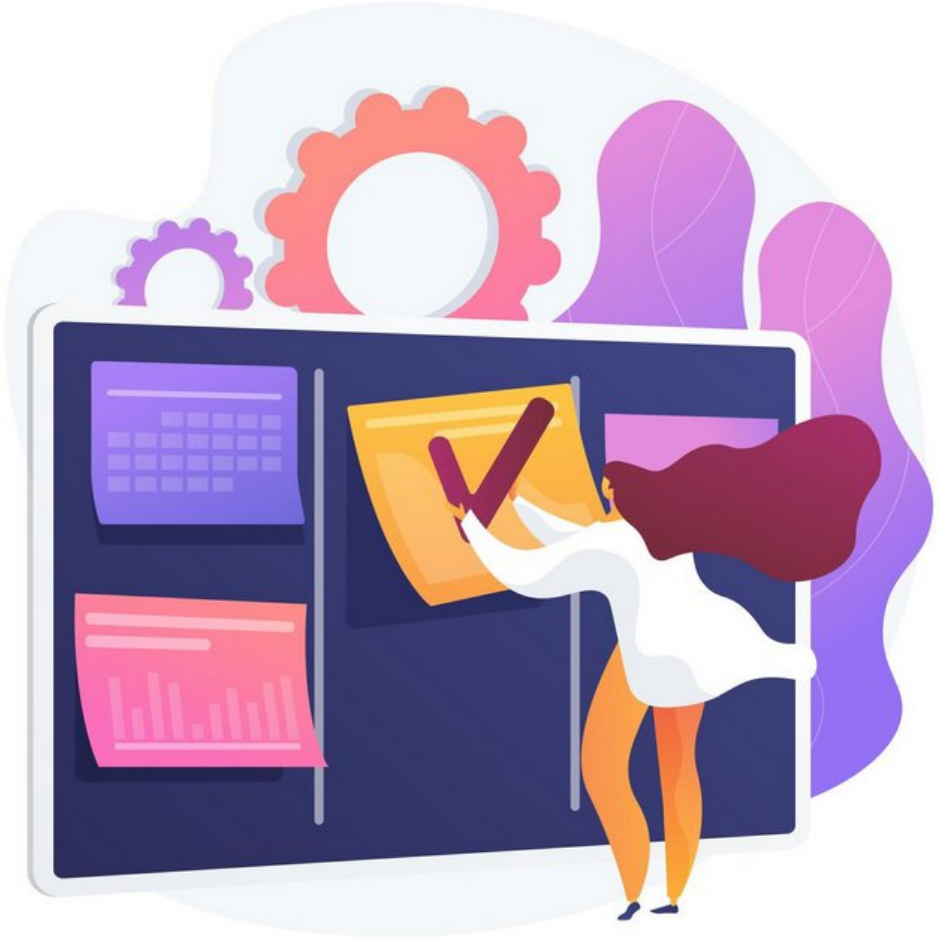
My PhD journey

From a large-scale study of Android Apps, we have encountered the reality of this ecosystem :



In recent years, these limitations have increased substantially...

Table of contents



Background

Objectives

As a researcher :

- Study apps to understand the whole ecosystem

Objectives

As a researcher :

- Study apps to understand the whole ecosystem

As a regulator :

- Verify the legality of the app’s personal data collection

Objectives

As a researcher :

- Study apps to understand the whole ecosystem

As a regulator :

- Verify the legality of the app’s personal data collection

As a user :

- Be assured of the security and privacy of each app

Monitoring data collection, on browser

Easy !

On most browsers (Chrome, firefox, brave..)
You can gather **un-encrypted** network traffic, with the developper settings.

Monitoring data collection, on browser

Easy !

On most browsers (Chrome, firefox, brave..)
You can gather **un-encrypted** network traffic, with the developper settings.

➡ By controlling the browser we control our data

We have :

- A lot of privacy focused browsers (Firefox, Brave..)
- Proper Privacy Enhancing Technologies:

AdBlockers (Adblock, UblockOrigin...)
Tracker protection (*Decentraleyes*...)

Monitoring data collection, on browser

Easy !

On most browsers (Chrome, firefox, brave..)
You can gather **un-encrypted** network traffic, with the developper settings.

➡ By controlling the browser we control our data

We have :
- Browsers for privacy (Firefox, Brave..)
- Privacy addons:

AdBlockers (Adblock, UblockOrigin...)
Tracker protection (*Decentraleyes*...)

Google fights against this freedom :
- Manifest v3
- The death.. of the death... of 3rd party cookies

Monitoring data collection, on mobile

A nightmare !

No easy method to gather un-encrypted network traffic

Monitoring data collection, on browser

A nightmare !

Traffic is mostly app-based

➡ By controlling the OS we control our data

Alternative OSs fight for privacy, but aren't very common :

– Lineage, Graphene, ***/e/OS***

Android ⇒ limited access to your data.

– Even worse if not rooted.

No easy method to gather un-encrypted network traffic

Collecting traffic in Android

Swimming in a vat of acid...

Static VS dynamic

2 complementary analysis methods :

Static

Without using the app

Advantages :
- Gives insight on the inner workings of the app
- Easy and quick to do

Drawbacks :
- Obfuscation limits scaling
- Nearly impossible to gather network traffic
- Parts of the app can be unused in reality

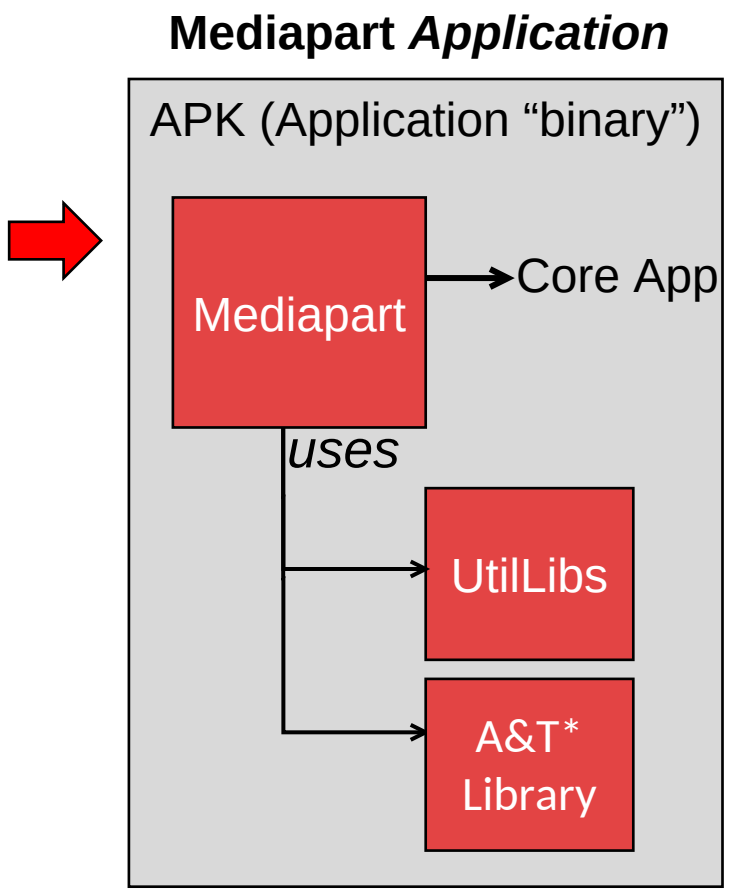
Dynamic

Actively using the app

Advantages :
- Allows to generate network traffic
- Shows how the app behaves

Drawbacks :
- Complex tooling
- Encryption

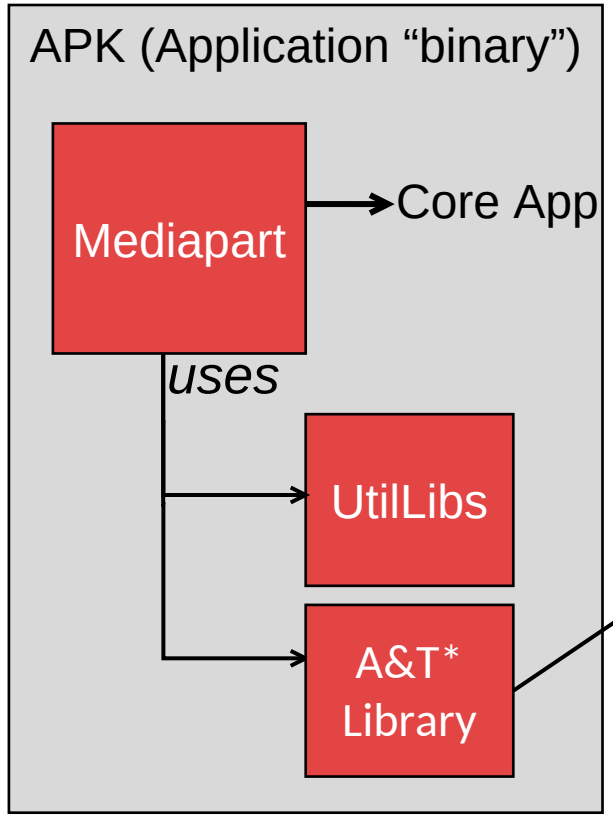
Statically test apps



*A&T = Advertising and tracking

Statically test apps

Mediapart *Application*

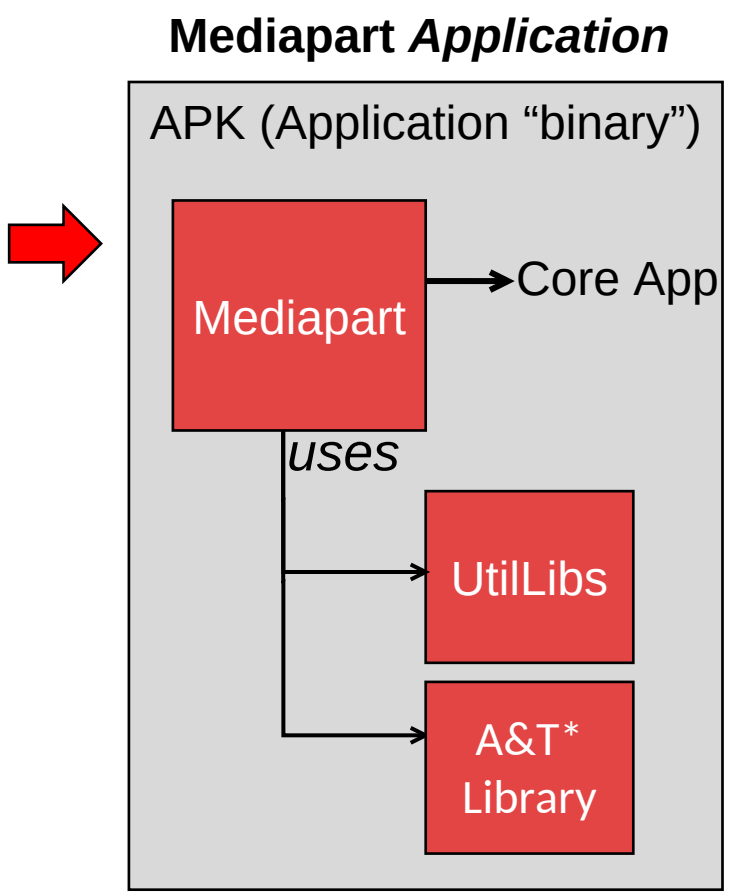


*A&T = Advertising and tracking

Android application are directly packaged with Ad/tracker libraries

... and many more

Statically test apps

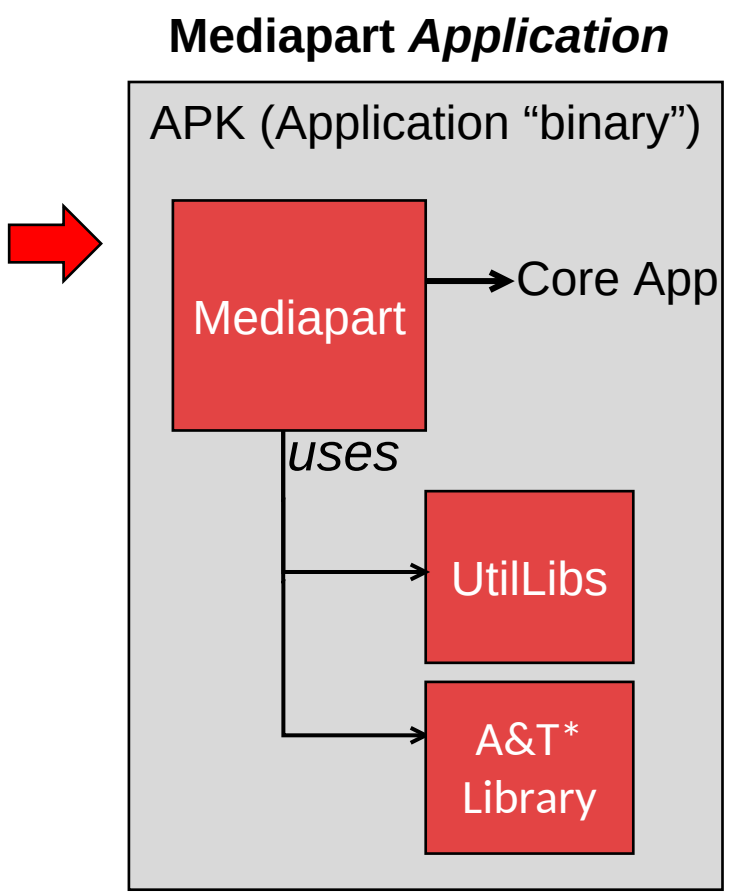


*A&T = Advertising and tracking

Easy to :

- Simply unzip the APK
- Decompile the app with APKlab/Jadx
- Search for specific security measures with APKiD

Statically test apps



*A&T = Advertising and tracking

Easy to :

- Simply unzip the APK
- Decompile the app with APKlab/Jadx
- Search for specific security measures with APKiD

But limited and hard to scale

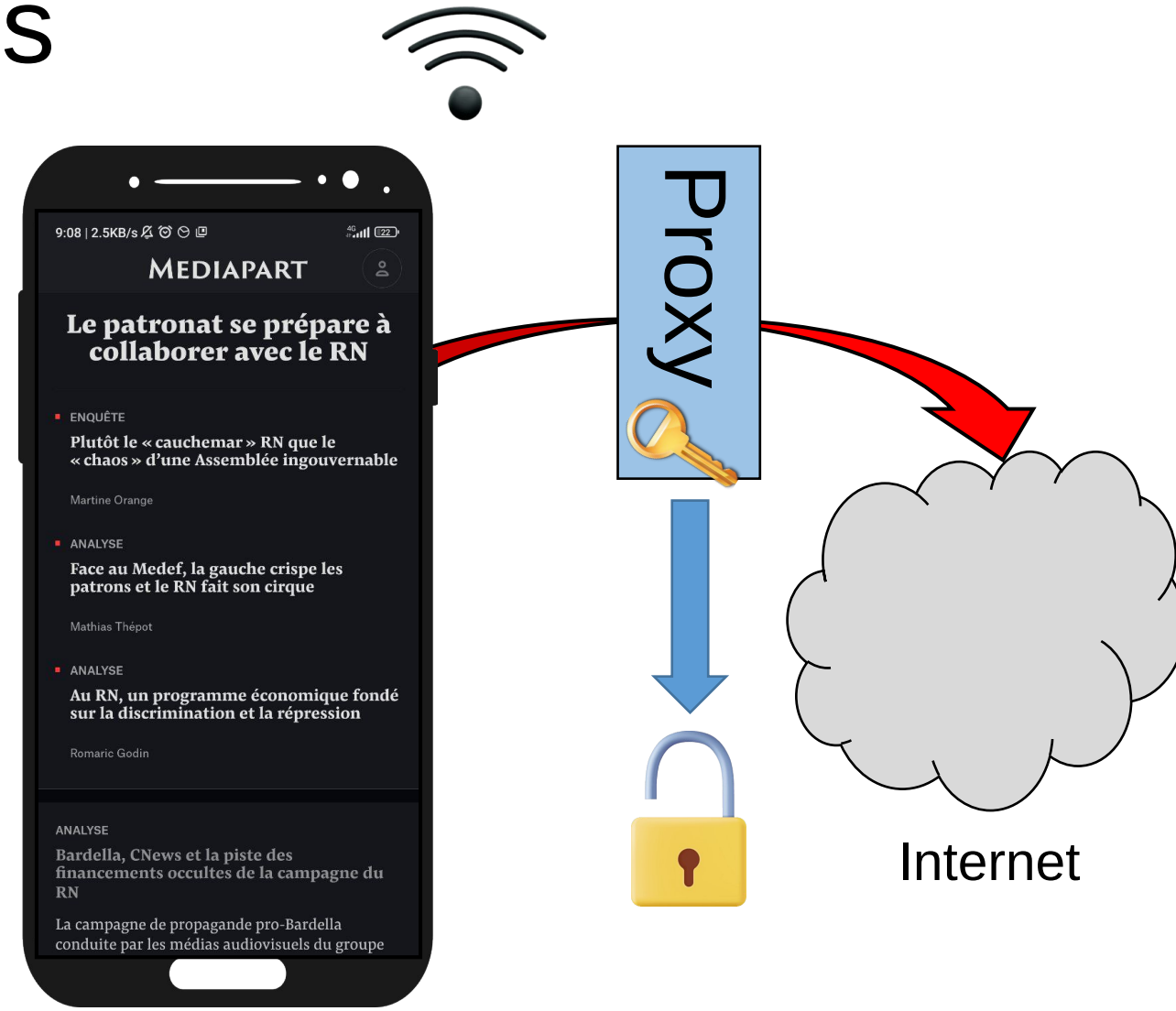
Dynamically test apps

Dynamically, it’s not as easy :

Dynamically test apps

Dynamically, it’s not as easy :

Sol1. Use a proxy



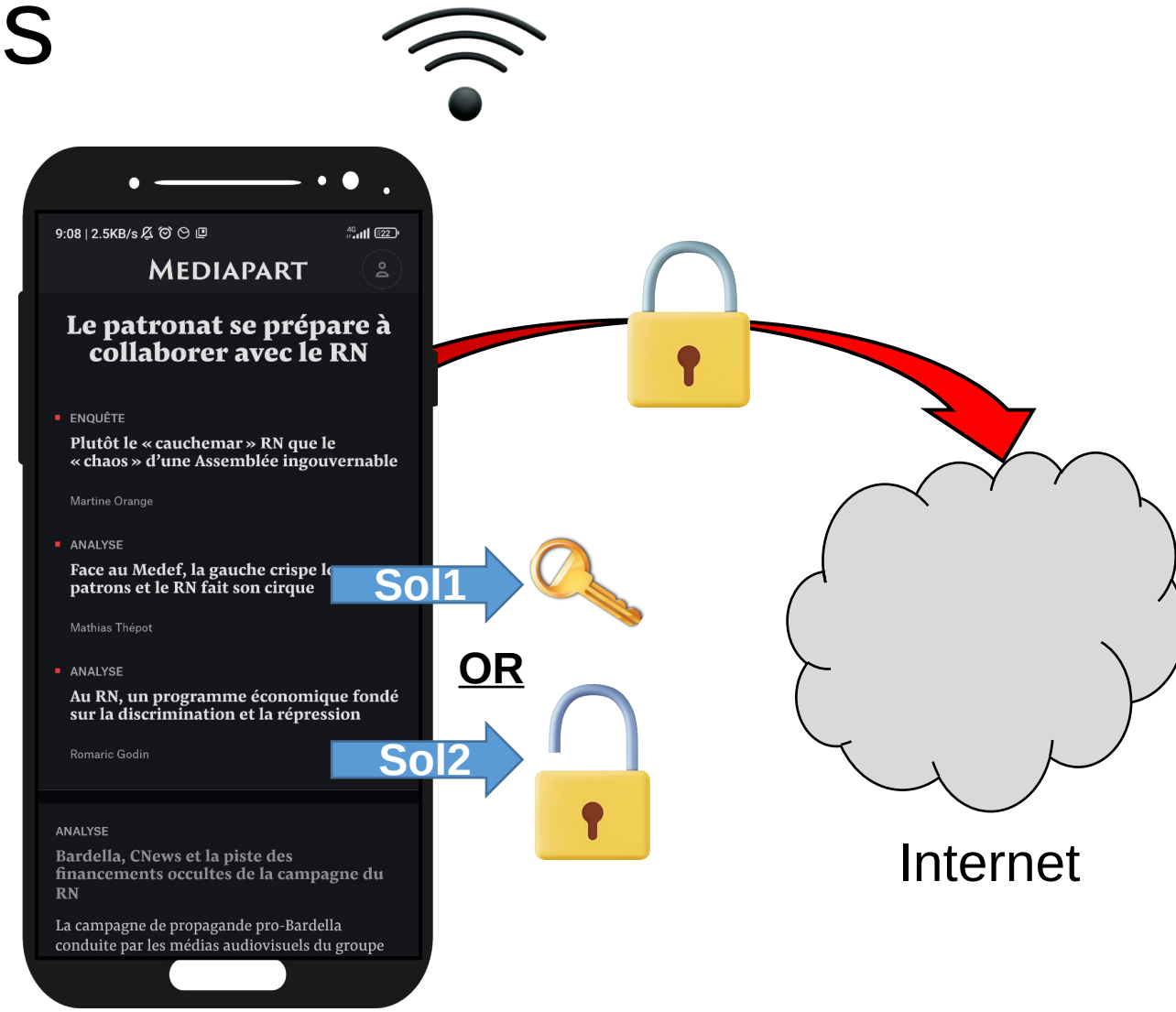
Dynamically test apps

Dynamically, it’s not as easy :

Sol1. Use a proxy

Sol2. Modify App or system either:

- A. To collect decryption key, or
- B. to collect unencrypted traffic



Dynamically test apps

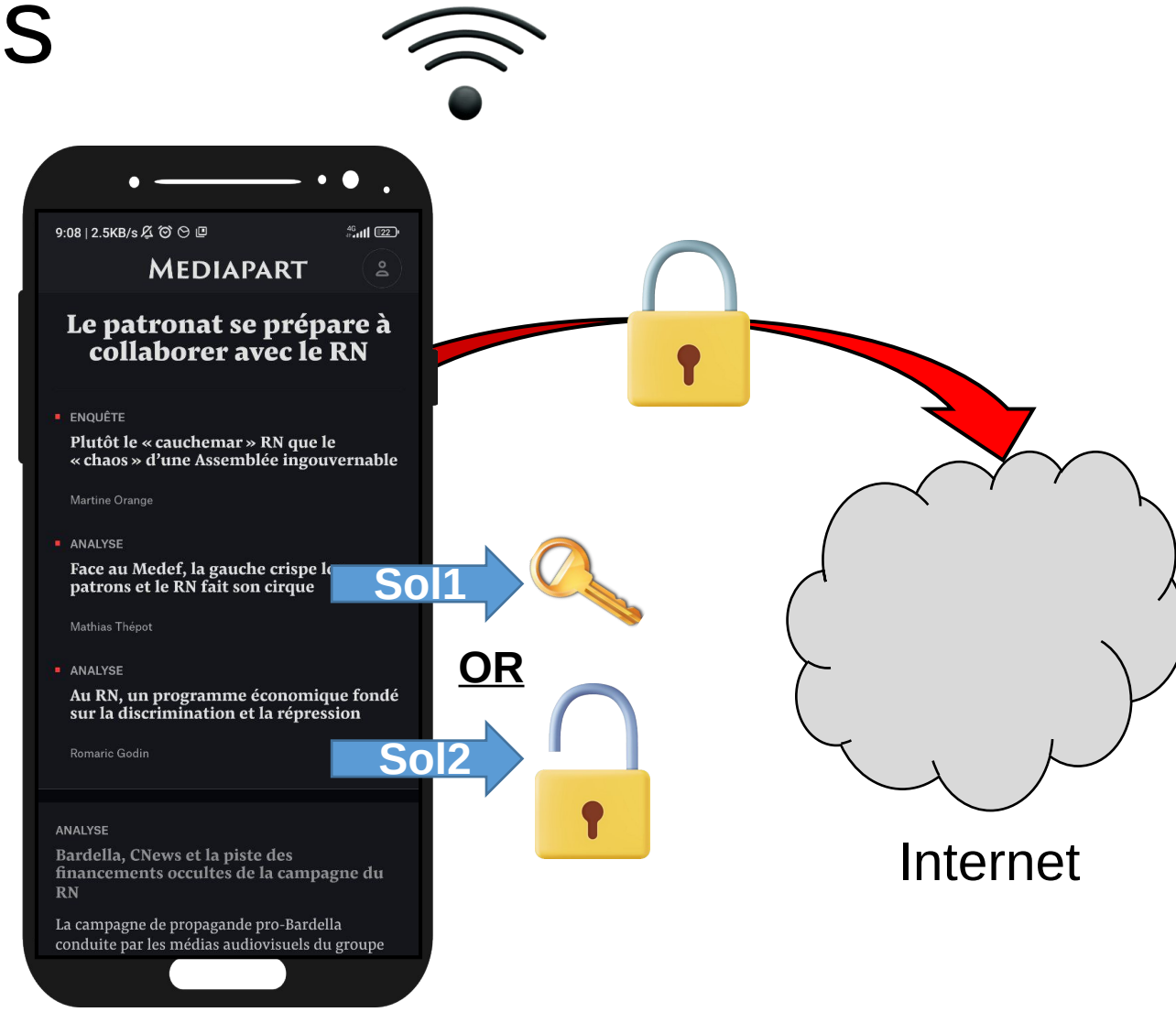
Dynamically, it’s not as easy :

Sol1. Use a proxy

Sol2. Modify App or system either:

- A. To collect decryption key, or
- B. to collect unencrypted traffic

Each solution has severe limitations



Dynamically test apps

Dynamically, it's not as easy :

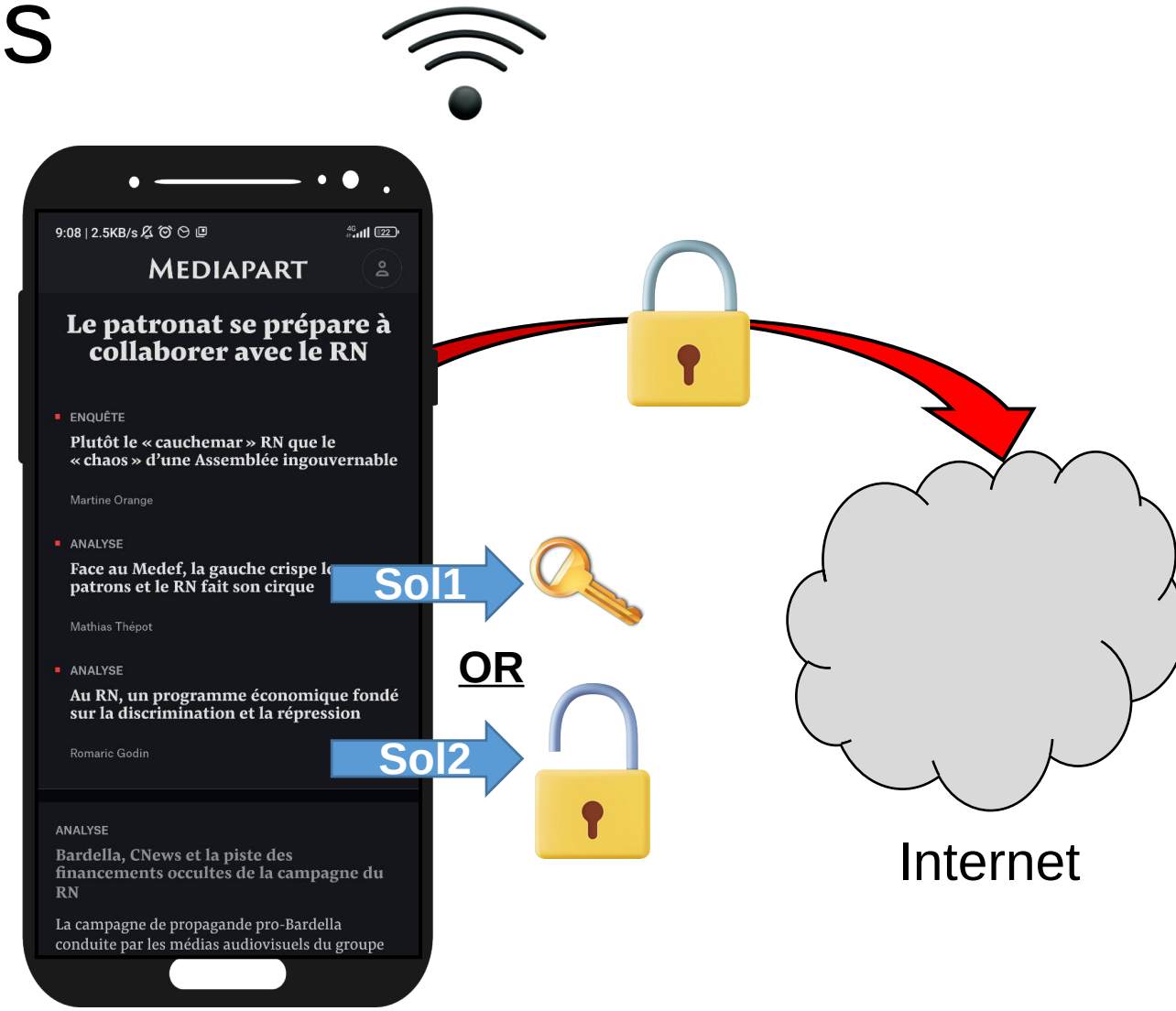
Sol1. Use a proxy

Sol2. Modify [illegible] or system either:

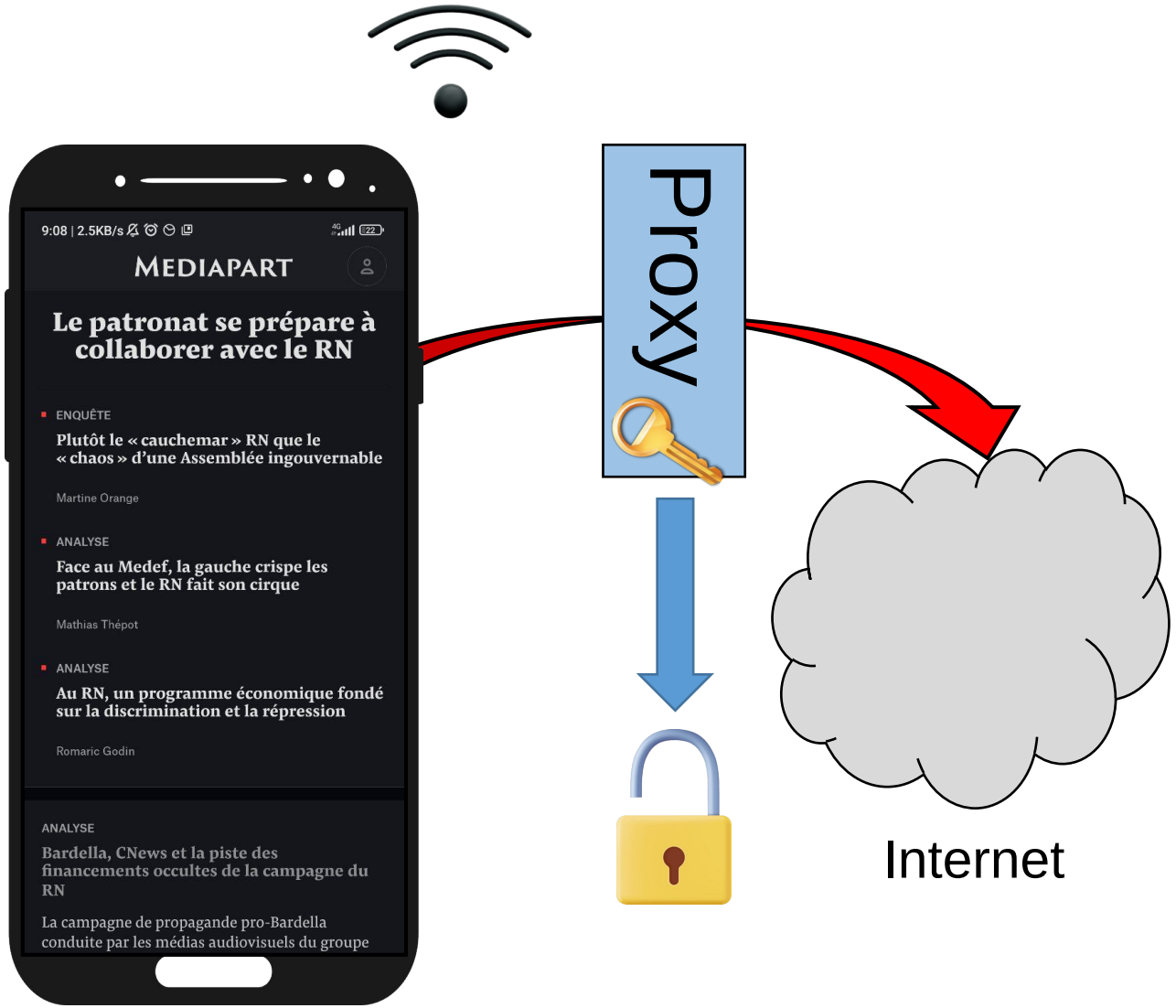
- A. To c[illegible]yption key, or
- B. to collect un[illegible]d traffic

Better solutions in theory but not in practice.. **yet**

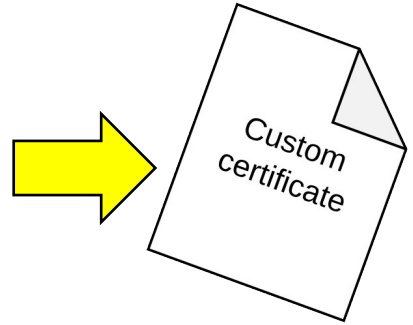
Each solution has severe limitations



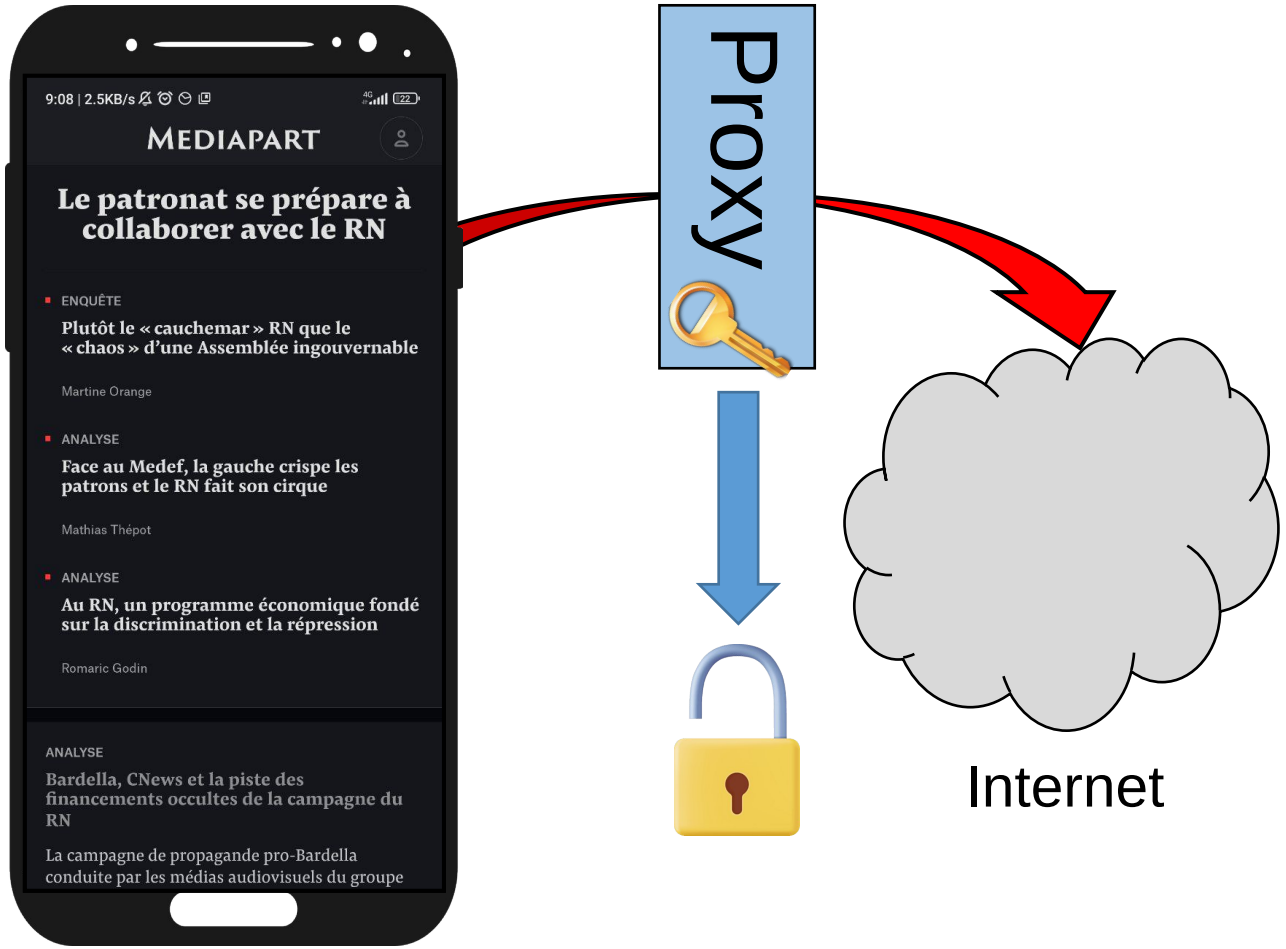
The best solution: Proxy



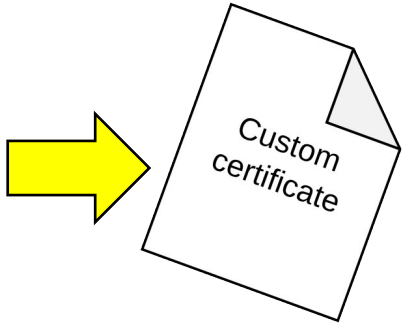
The best solution: Proxy



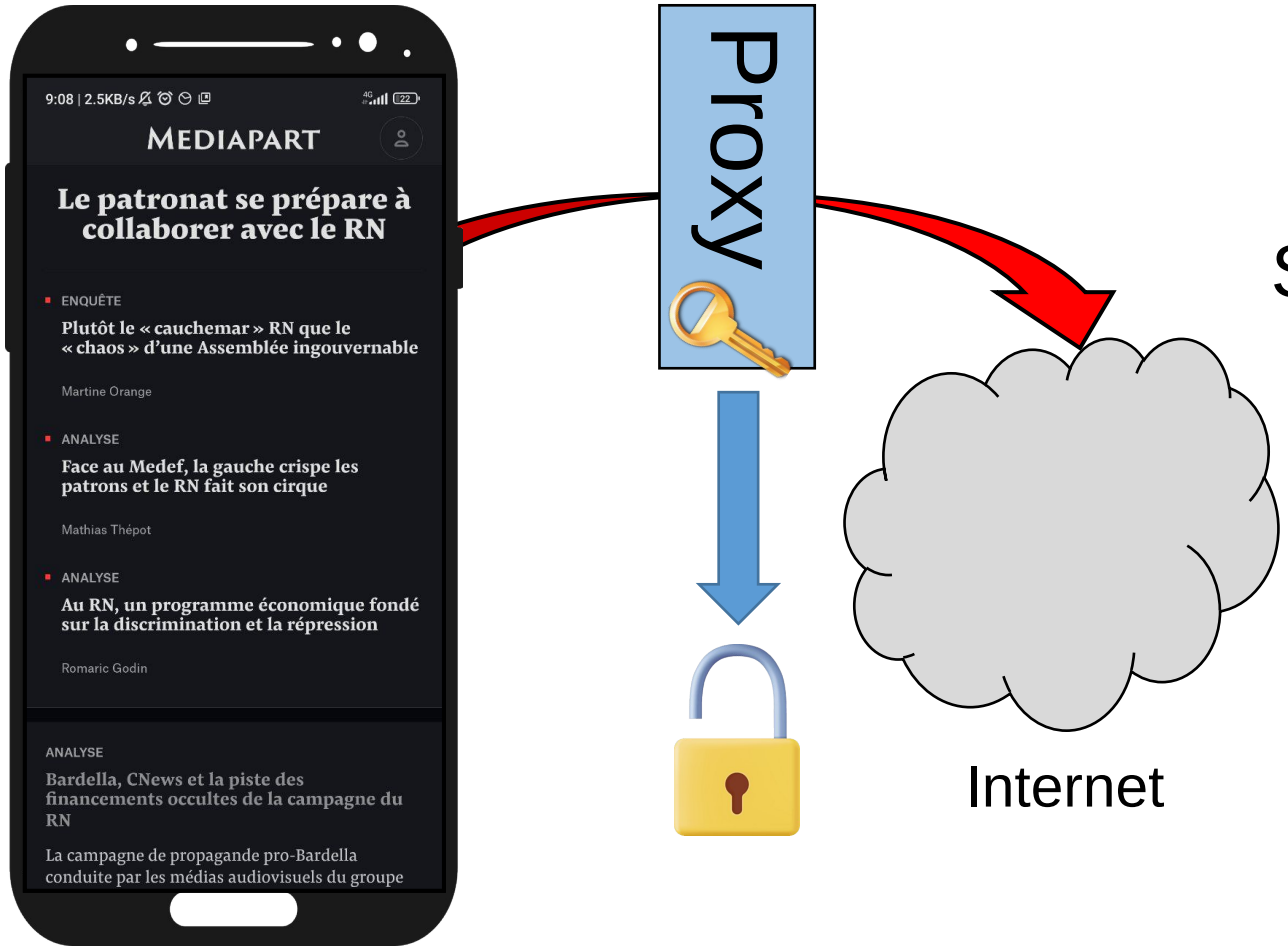
App must trust the proxy certificate



The best solution: Proxy

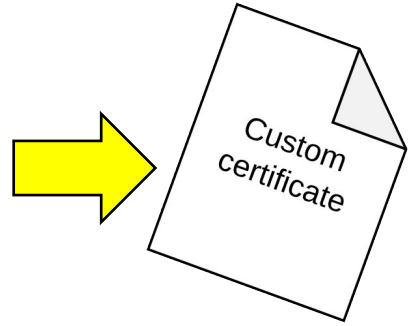


App must trust the proxy certificate

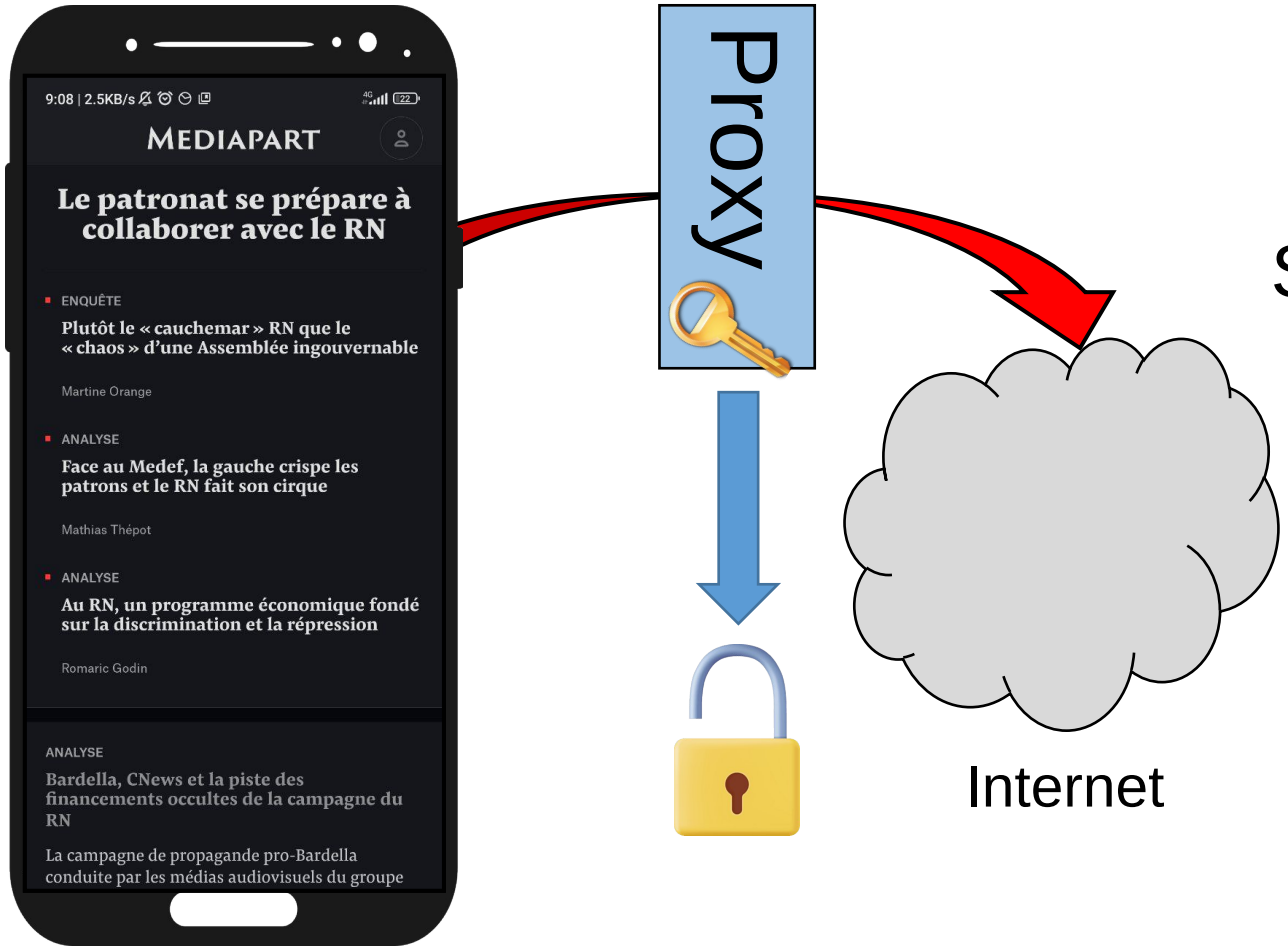


Since v7.0 (2016), only 2 solutions :
- Create a custom ROM

The best solution: Proxy



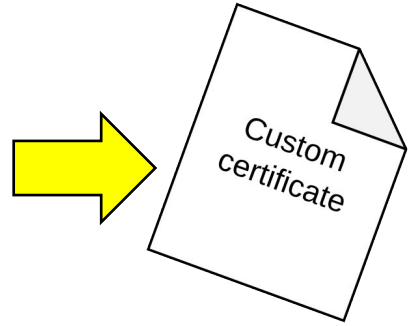
App must trust the proxy certificate



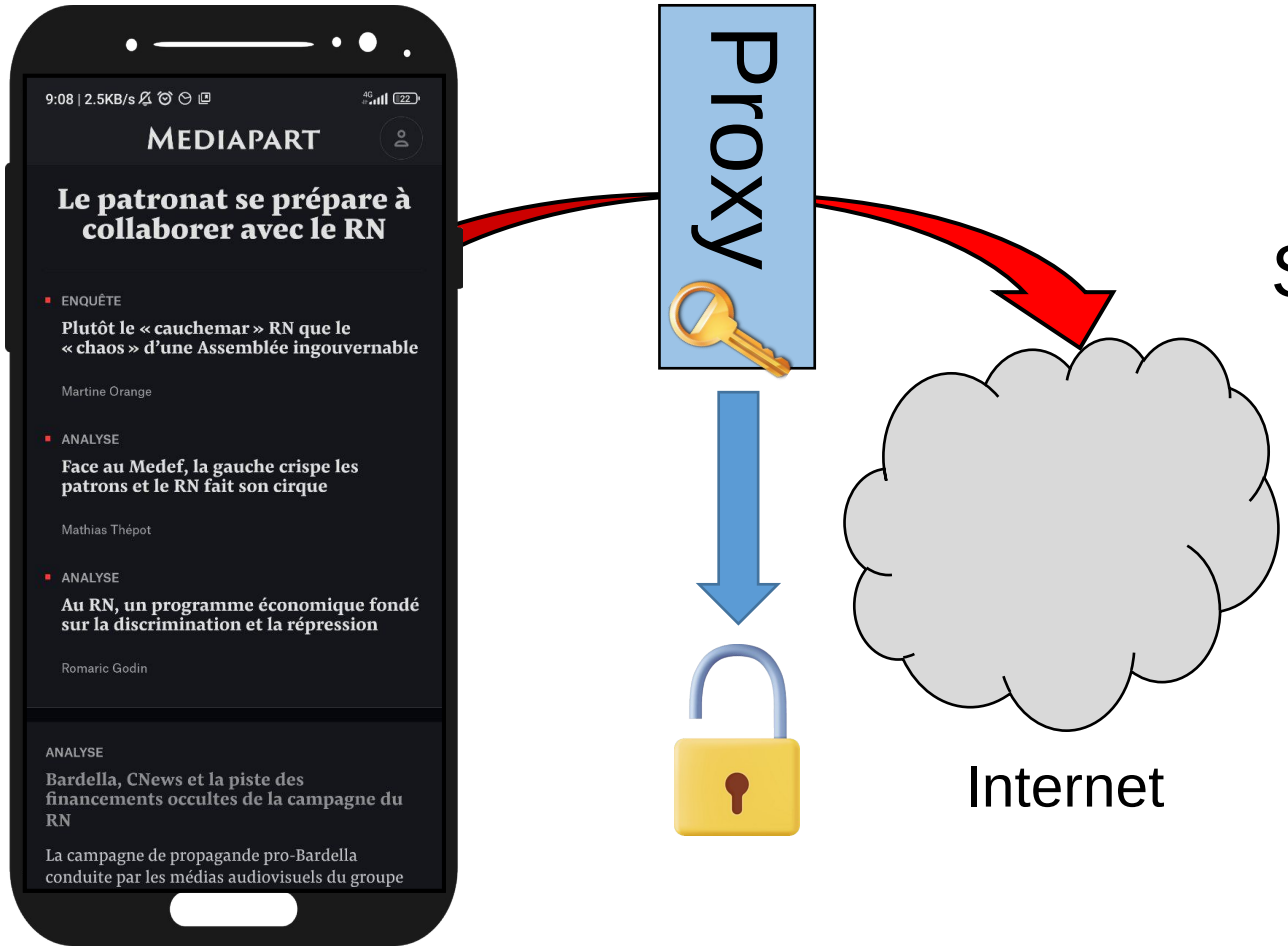
Since v7.0 (2016), only 2 solutions :

- ~~Create a custom ROM~~
  Complex and creates other problems

The best solution: Proxy



App must trust the proxy certificate



Since v7.0 (2016), only 2 solutions :

- ~~Create a custom ROM~~
  Complex and creates other problems

- Become root (superuser)

Android and Rooting

A MUST for dynamic analysis

This affects the device :

- Breaks bootloader security
- Breaks most hardware-based security
- Loss of warranties

Android and Rooting

A MUST for dynamic analysis

This affects the device :

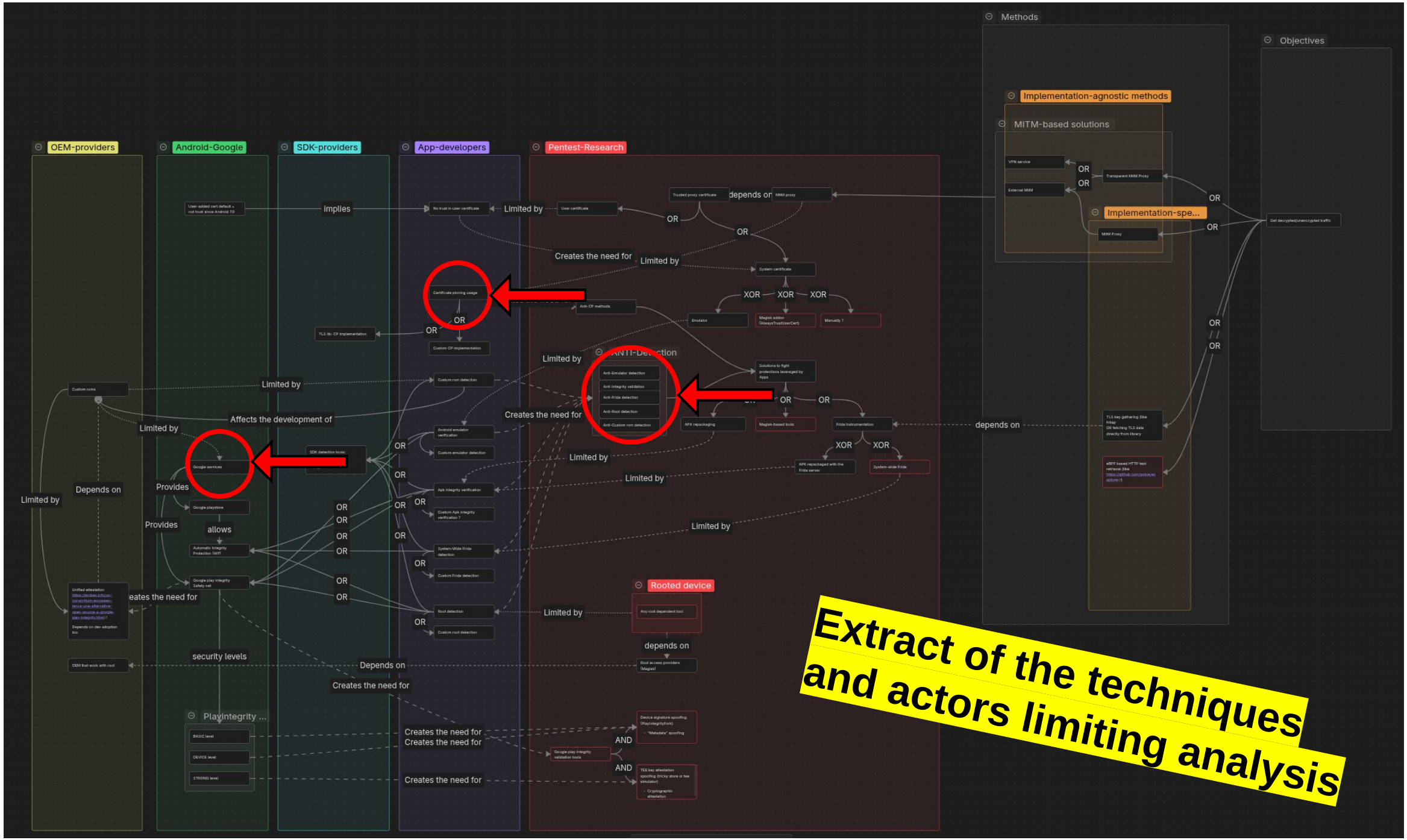
- Breaks bootloader security
- Breaks most hardware-based security
- Loss of warranties

This is not very accessible, dependent on OEM, and prone to errors and changes in Android.

Analysis countermeasures

As if the Acid Vat wasn’t enough, it was full of... snakes ?

Let’s take a look at what apps can do to avoid being analysed


Methods
Objectives
Implementation-agnostic methods
MITM-based solutions
Implementation-spe...
OEM-providers
Android-Google
SDK-providers
App-developers
Pentest-Research
implies
Limited by
depends on
Creates the need for
Affects the development of
Provides
allows
Depends on
security levels
Rooted device
Playintegrity ...
OR
XOR
AND
Extract of the techniques and actors limiting analysis

1 - Apps can detect root/fake devices

1 - Apps can detect root/fake devices

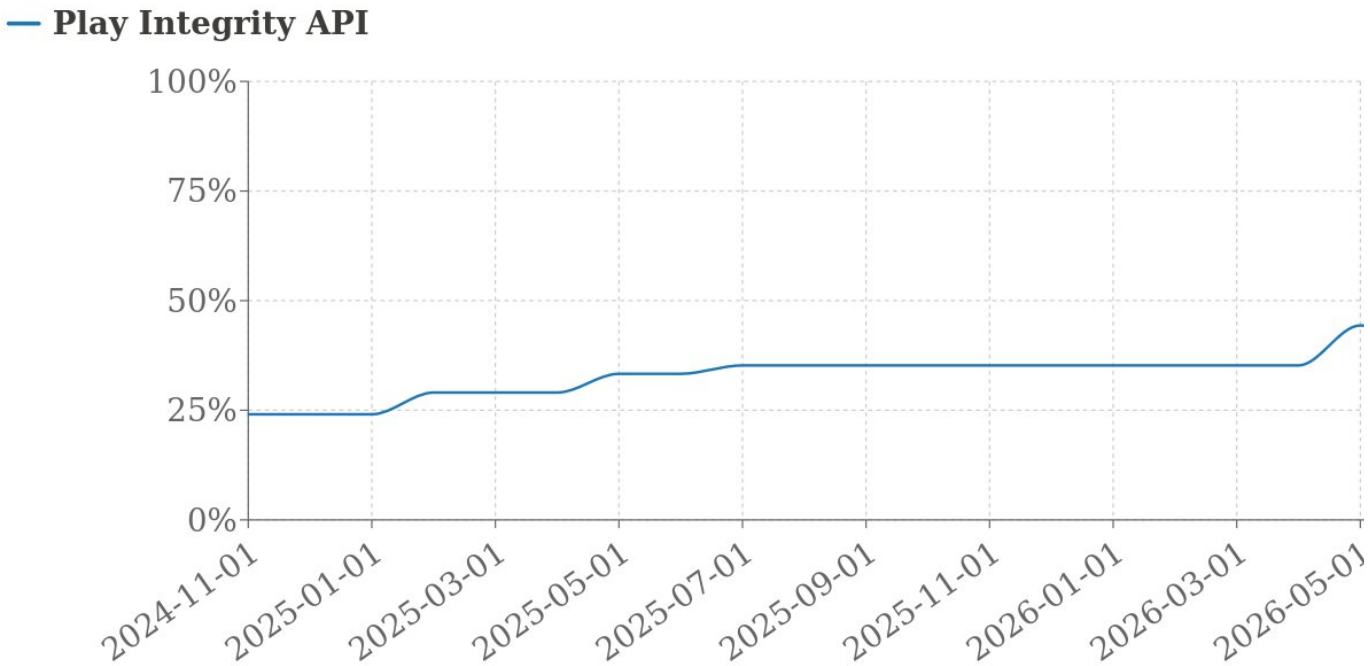
- Custom detection scripts
  - widely used by 3rd party libraries (Ads, tracking..)
- Security tools

1 - Apps can detect root/fake devices

- Custom detection scripts
  - widely used by 3rd party libraries (Ads, tracking..)
- Security tools

The most common :

- PlayIntegrity

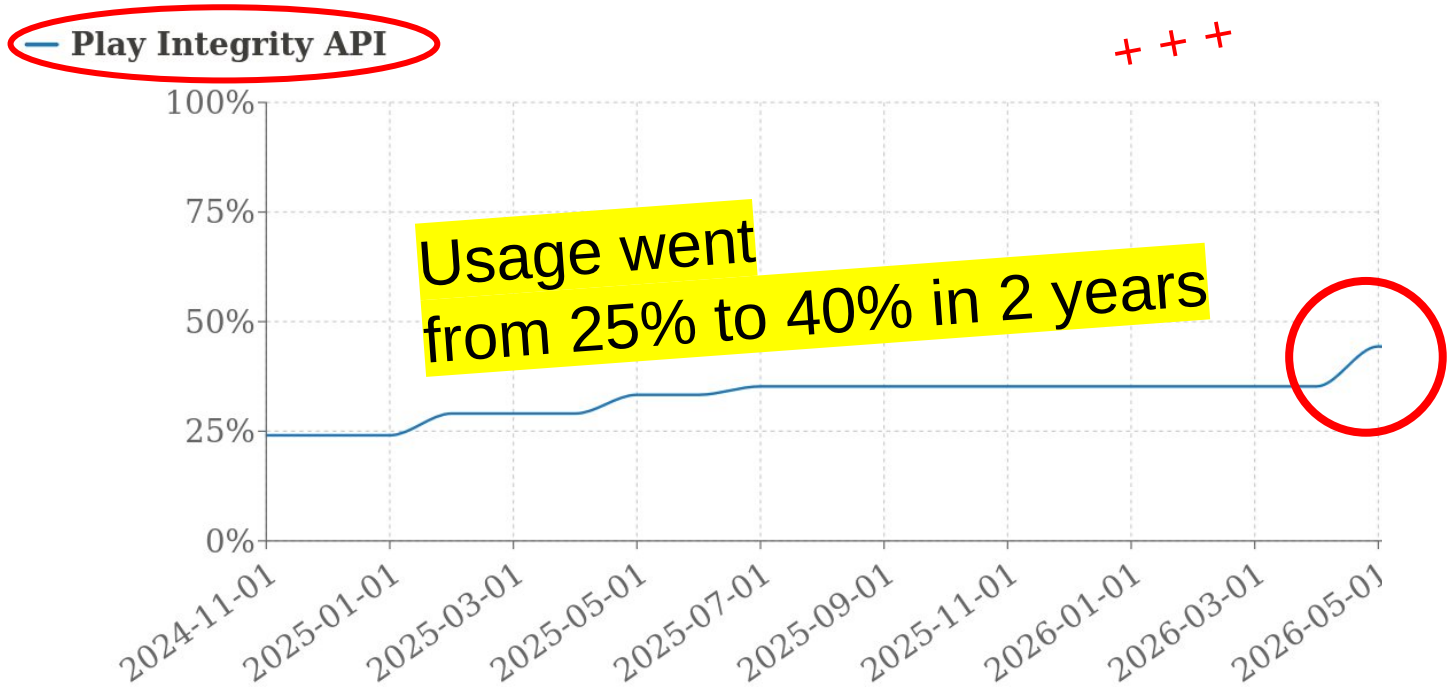


1 - Apps can detect root/fake devices

- Custom detection scripts
  - widely used by 3rd party libraries (Ads, tracking..)
- Security tools

The most common :

- PlayIntegrity



Play integrity api usage (% of 570 top apps)

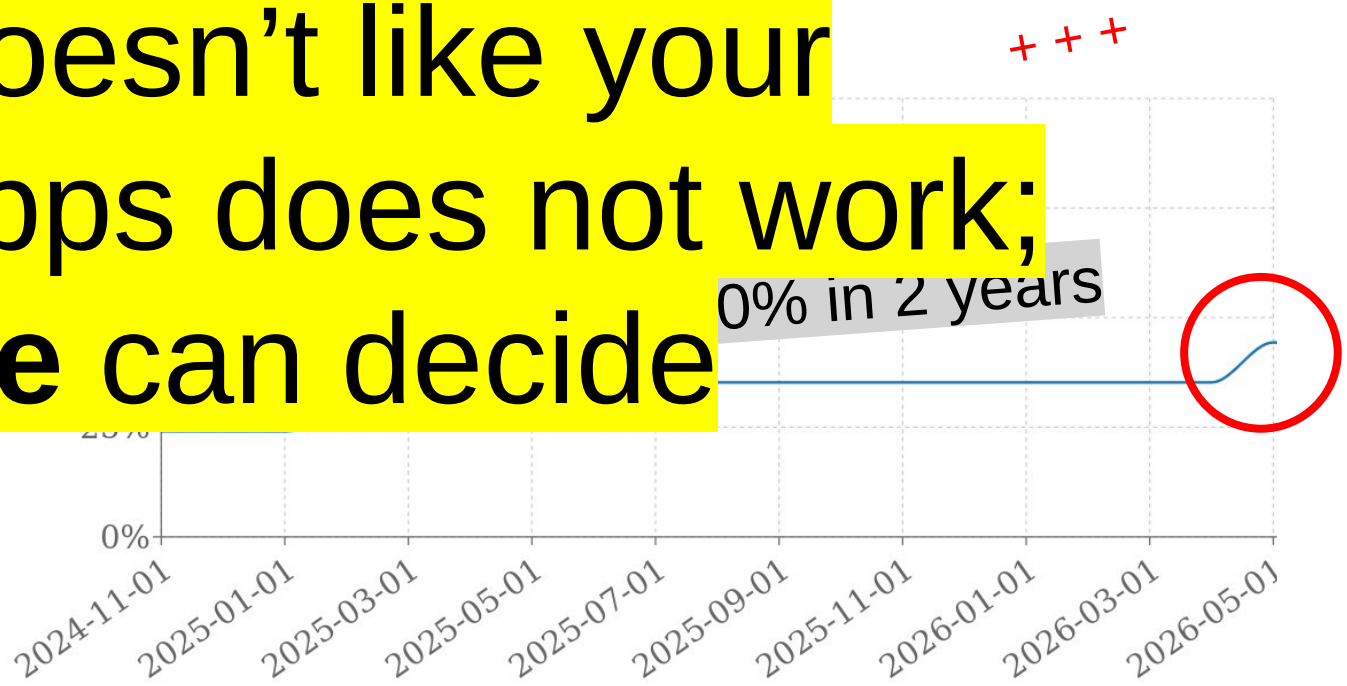
1 - Apps can detect root/fake devices

- Custom detection scripts
  - wi racking..)
- Secu
- The m
- Playl

What this means :

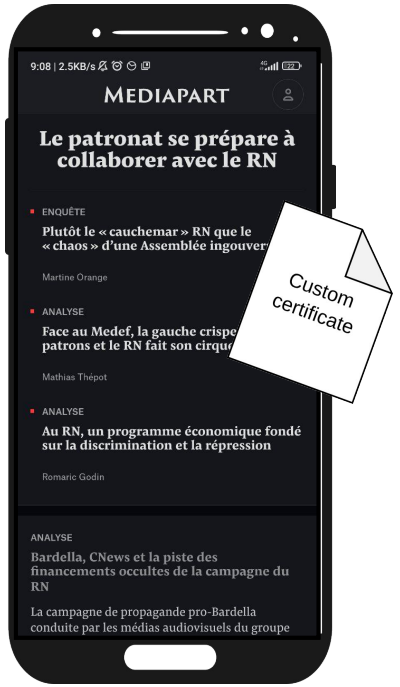
If **Google** doesn't like your setup, the apps does not work;

Only **Google** can decide



Play integrity api usage (% of 570 top apps)

2 - Apps can... detect proxies



2 - Apps can... detect proxies

App decides **not to trust** system certificates

App uses it’s own certificates :

this is called **Certificate pinning**

2 - Apps can... detect proxies

Instead of trusting the system certificates

– Which contains our own proxy certificate

The app brings it's own certificate and only trusts them :

this is called **Certificate pinning**

Dataset	Total
Full dataset: 217 *apps*	**37 (17.0%)**
Apps with CP only:	**62 (16.0%)**
37 apps, 104 domains,	**31 (29.8%)**
386 network flows	**27.0%**

more than 1/6 app uses CP

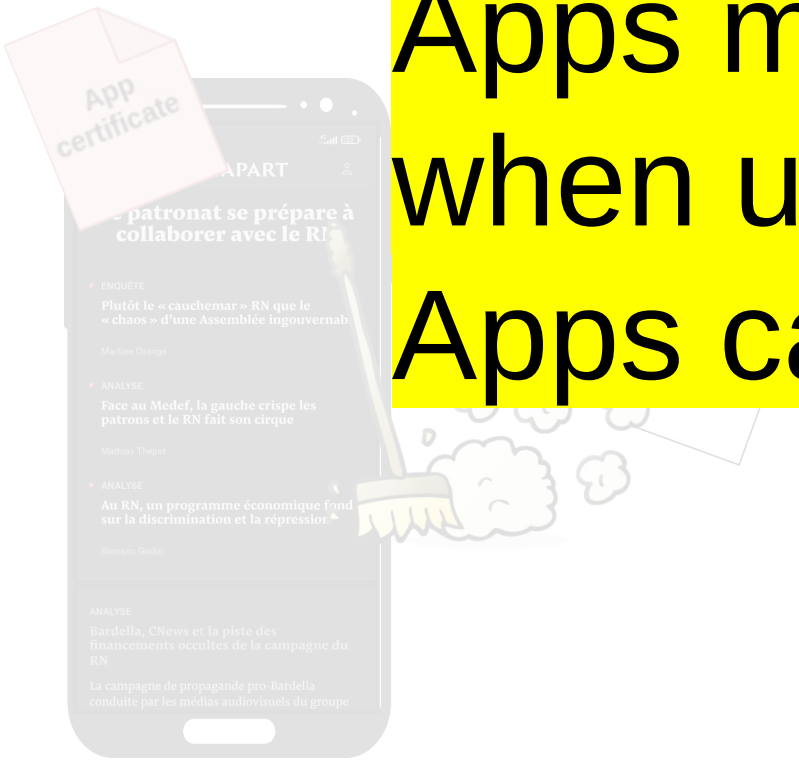
Amount and percentage of apps using CP

2 - Apps can... detect proxies

What this means :

Apps may stop sending data when under scrutiny.

Apps can avoid being analyzed



ates

tificate

d only trusts them :

more than 1/6 app uses CP

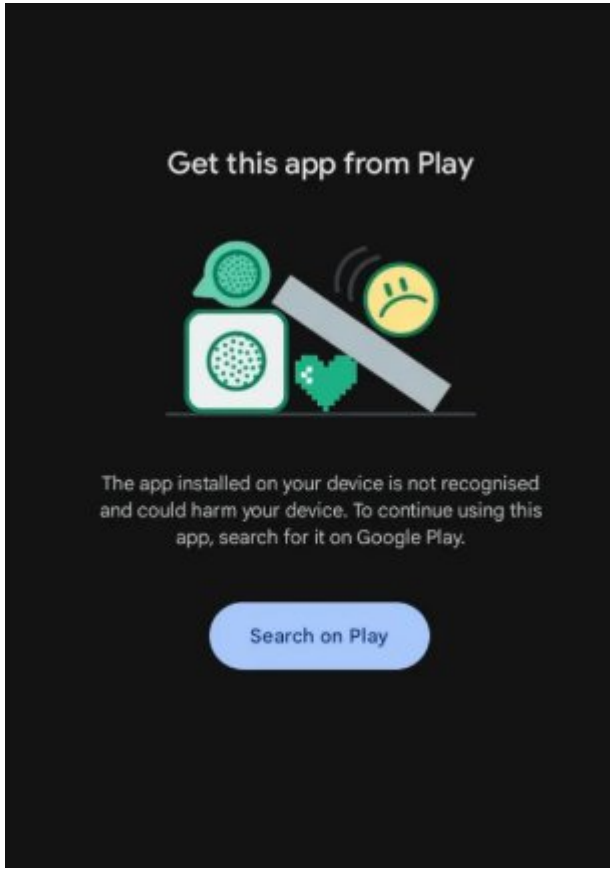
Full dataset: 217 *apps*	37 (17.0%)
Apps with CP only:	62 (16.0%)
37 apps, 104 domains,	31 (29.8%)
386 network flows	27.0%

Amount and percentage of apps using CP

3 - Apps can detect if they are not downloaded from the Playstore

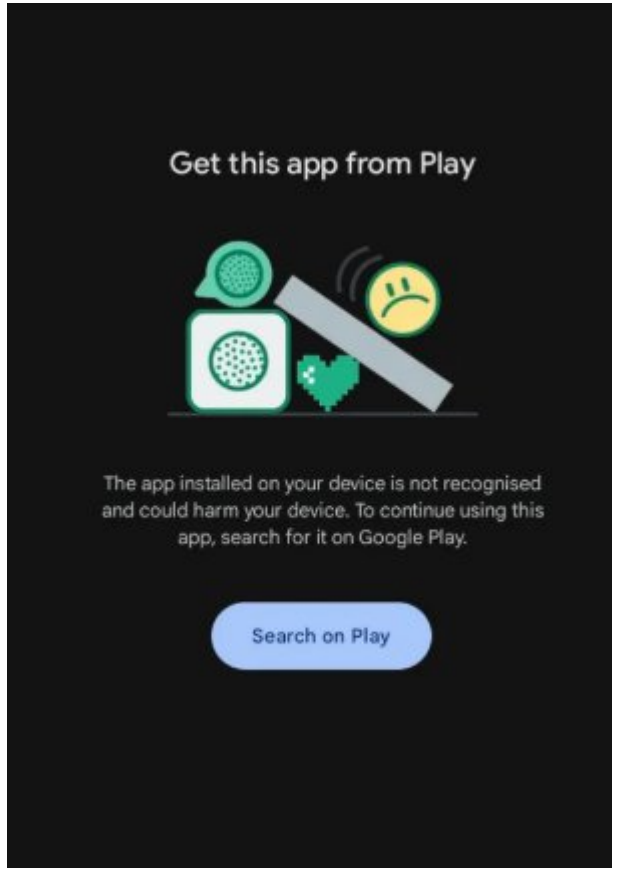
3 - Apps can detect if they are not downloaded from the Playstore

Since 2024, apps may show this message when started.



3 - Apps can detect if they are not downloaded from the Playstore

Since 2024, apps may show this message when started.

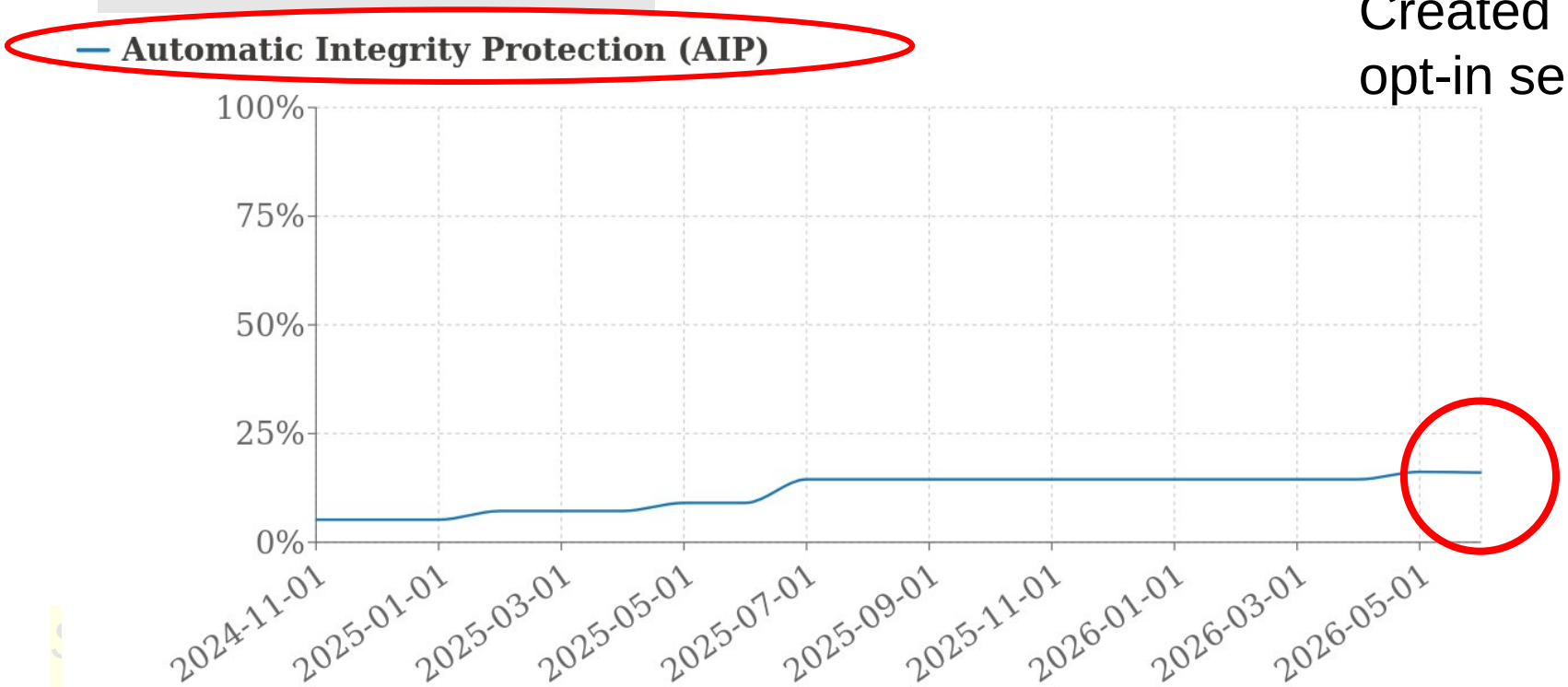


For regulators,
we are dependent on the Playstore :

- They decide if we are allowed to test
- Region/availability restrictions
- Playstore only provides last app version

3 - Apps can detect if they are not downloaded from the Playstore

Since 2024, apps may show this message when started.

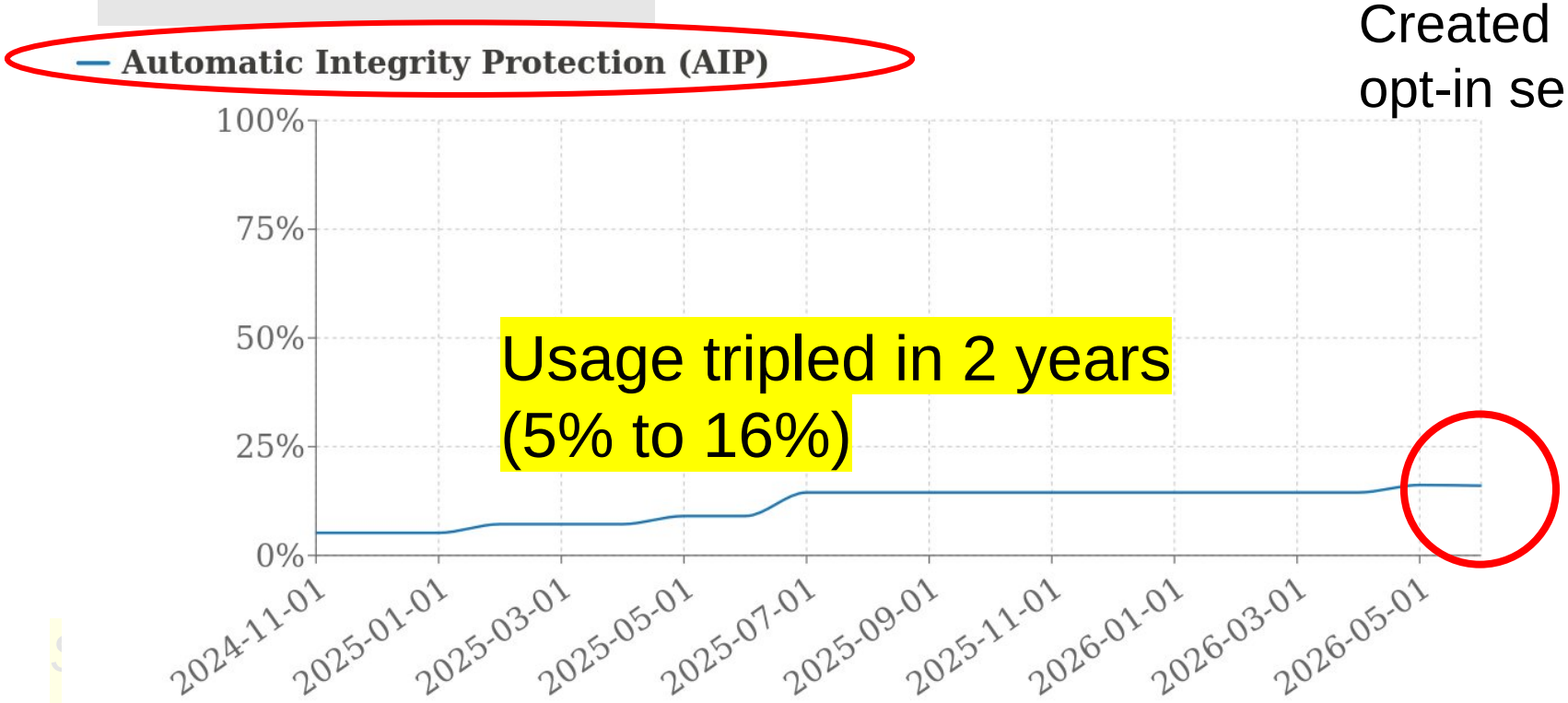


This protection is called Automatic Integrity Protection.
Created by Google as a simple opt-in service.

3 - Apps can detect if they are not downloaded from the Playstore

Since 2024, apps may show this message when started.

This protection is called Automatic Integrity Protection.
Created by Google as a simple opt-in service.



Usage tripled in 2 years (5% to 16%)

3 - Apps can detect if they are not downloaded from the Playstore

What this means : Apps cannot be shared and redistributed.
Reproducibility is impossible.
Existing App databases can’t be tested

Many more adversarial techniques

Many more adversarial techniques

- **Dynamic code loading**
  - Apps will update themselves, rendering static studies useless

This is the case with **TikTok** :

- After a few minutes of scrolling, the APK will silently update (without the Playstore)

Many more adversarial techniques

- **Dynamic code loading**
  - Apps will update themselves, rendering static studies useless
- **Delaying behaviors**
  - Can wait some time before collecting personal data (hours, days, weeks)

Many more adversarial techniques

- **Dynamic code loading**
  - Apps will update themselves, rendering static studies useless
- **Delaying behaviors**
  - Can wait some time before collecting personal data (hours, days, weeks)
- Apps can detect **static or dynamic code changes**

Many more adversarial techniques

- **Dynamic code loading**
  - Apps will update themselves, rendering static studies useless
- **Delaying behaviors**
  - Can wait some time before collecting personal data (hours, days, weeks)
- Apps can detect **static or dynamic code changes**
- **Device fingerprinting**
  - Heavily instrumented devices can be detected

A required tool : Frida

Frida allows to modify the app itself (dynamically):
– Remove certificate pinning
– Remove root/integrity detections

And again, limitations and side-effects exist
Frida can also be detected...

How to build a usable setup

Why existing setups fall short

A very ***realistic*** testing device

A very ***realistic*** testing device

- A physical device

Because an emulator is trivially detected

A very ***realistic*** testing device

Allows root

- A physical device
- Rooted with Magisk

A very ***realistic*** testing device

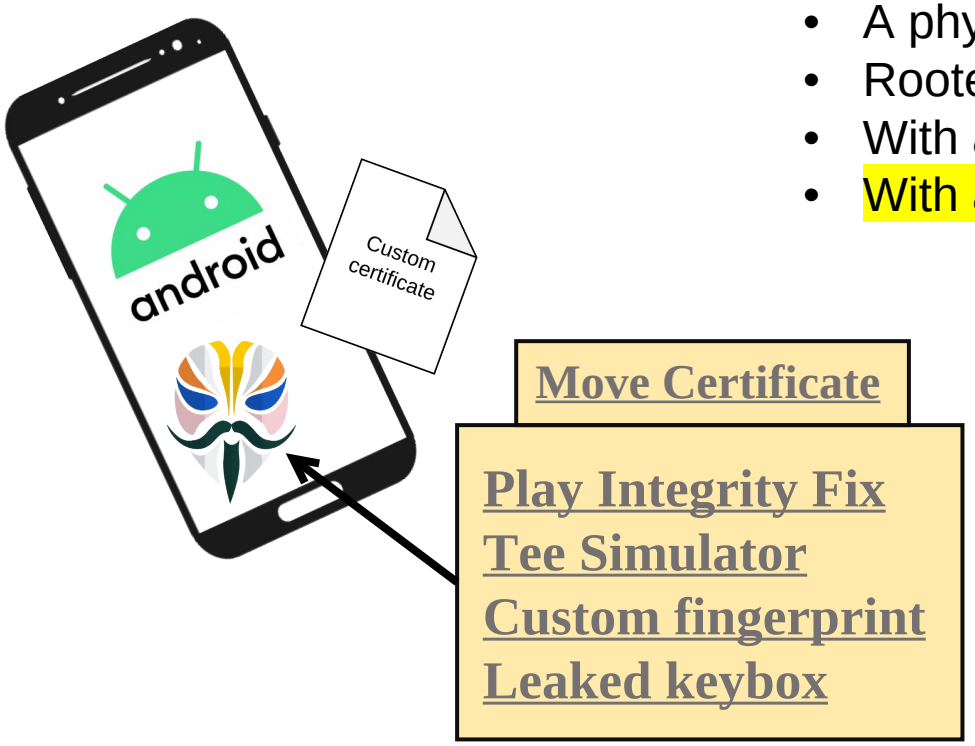
- A physical device
- Rooted with Magisk
- With addons to fight PlayIntegrity

Play Integrity Fix
Tee Simulator
Custom fingerprint
Leaked keybox

+ dependencies

Fixes PlayIntegrity limitations for a few days/weeks only, there is no stable solution

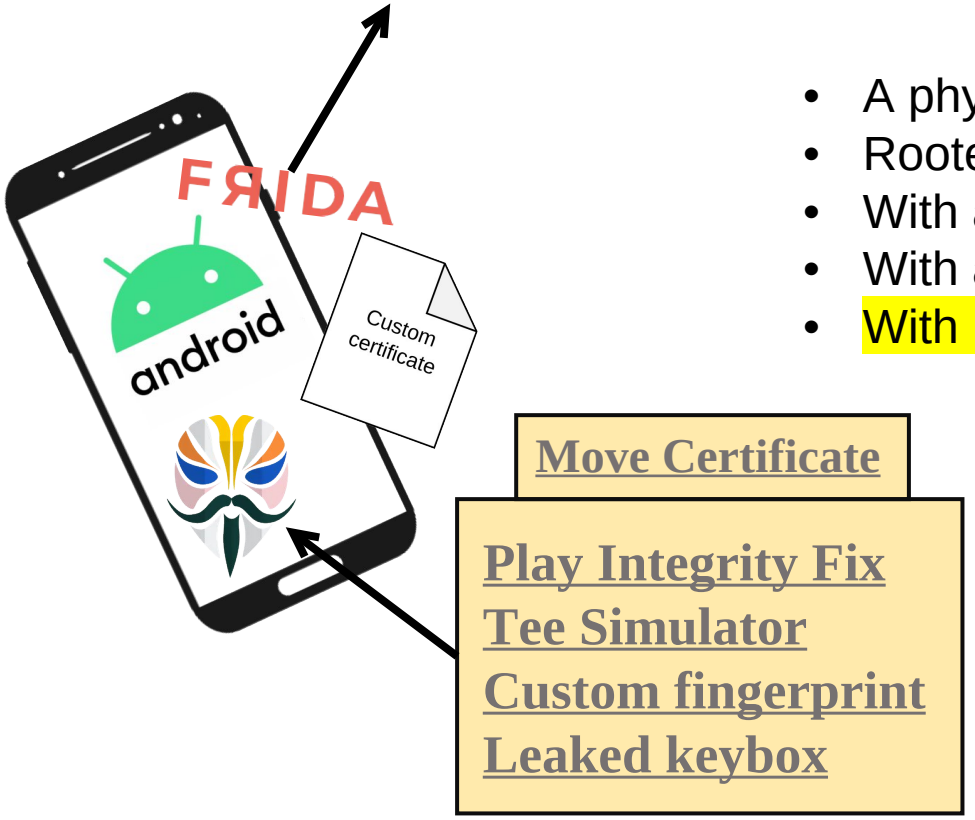
A very ***realistic*** testing device



- A physical device
- Rooted with Magisk
- With addons to fight PlayIntegrity
- With a proxy certificate

A very ***realistic*** testing device

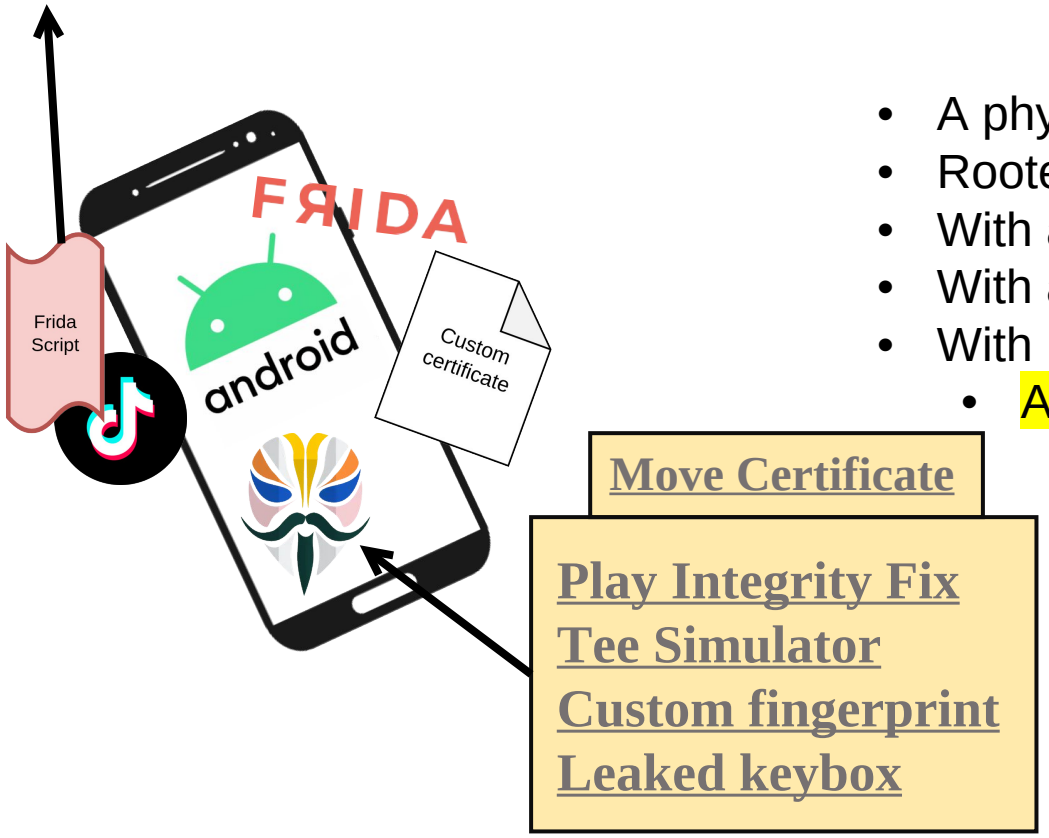
Allows to dynamically modify apps



- A physical device
- Rooted with Magisk
- With addons to fight PlayIntegrity
- With a proxy certificate
- With Frida running

A very ***realistic*** testing device

Allows to avoid root checks, and certificate pinning... sometimes



- A physical device
- Rooted with Magisk
- With addons to fight PlayIntegrity
- With a proxy certificate
- With Frida running
  - And the Frida script that modifies the app

A very ***realistic*** testing device.. ?

- A physical device
- Rooted with Magisk
- With addons to fight PlayIntegrity
- With a proxy certificate
- With Frida running
  - And the Frida script that modifies the app

Move Certificate

Play Integrity Fix
Tee Simulator
Custom fingerprint
Leaked keybox

+ dependencies

Great, you are ready to test apps, assuming :

- ***Overhead is not too high***
- ***Your tools work for the whole testing period***
- ***Your tools do not have side-effects***
- ***Apps do not find ways to detect XXXX***

...a pretty **unrealistic** testing device

- What this means for research
  - This ecosystem is very complex
  - Testing is very hard AND changes all the time
  - Our setup is prone to failures when anything updates
  - We need to study each and every **single variable**

- Google has a very important responsibility in this
  - They actively fight against control
  - They often decide to pull the switch and nothing works anymore

Takeaways

Between security and control

Android has been designed less controlable than a browser

- This affects users and regulators alike

Between security and control

Android has been designed less controlable than a browser

- This affects users and regulators alike

Google takes advantage of it’s complete control of Android :

- They impose what an accepted device is,
- They impose which actors should be trusted,
- They can decide to pull the switch.

Between security and control

Android has been designed less controlable than a browser

- This affects users and regulators alike

Google takes advantage of it’s complete control of Android :

- They impose what an accepted device is,
- They impose which actors should be trusted,
- They can decide to pull the switch.

Security is the main argument to limit control with each update :

- But security by obfuscation is not actual security,
- Complete opposition to the browser ecosystem;

Every solution brings new problems

This important cat-and-mouse game is highly limiting :

- Every solution brings new biases,
- Any difference with a normal device can be detected.

Every solution brings new problems

This important cat-and-mouse game is highly limiting :

- Every solution brings new biases,
- Any difference with a normal device can be detected.

Should applications have this much control over the user ?

- Users should be allowed to see what data is being collected,
- Users should be asked for better consent.

Every solution brings new problems

This important cat-and-mouse game is highly limiting :

- Every solution brings new biases,
- Any difference with a normal device can be detected.

Should applications have this much control over the user ?

- Users should be allowed to see what data is being collected,
- Users should be asked for better consent.

Technical solutions are temporary :

- Proper solutions will only be regulation,

Future of Android

A problem for OS alternatives (Lineage, */e/OS*, Graphene)

PlayIntegrity is a Sword of Damocles for OS alternatives :

- Google’s hardware-backed attestations can’t be valid on unofficial OSes

Future of Android

A problem for OS alternatives (Lineage, /e/OS, Graphene)

PlayIntegrity is a Sword of Damocles for OS alternatives :

- Google’s hardware-backed attestations can’t be valid on unofficial OSes

A problem for research

Devices cannot be tested without important biases

Lack of reproducibily of the environment

Future of Android

A problem for OS alternatives (Lineage, */e/OS*, Graphene)

PlayIntegrity is a Sword of Damocles for OS alternatives :

- Google’s hardware-backed attestations can’t be valid on unofficial OSes

A problem for research

Devices cannot be tested without important biases

Lack of reproducibily of the environment

A danger for regulation

We cannot check the privacy processes of apps :

- Apps have the control to hide unlawful practices

Thank you for listening !

Annex slides

Android, a *closed* environment

Android applications are difficult to analyze

<u>The system is an adversary</u>

- Needs a proxy to decrypt trafic
- Needs root to add certificates
- Root access is not always possible

<u>The apps have a lot of freedom</u>

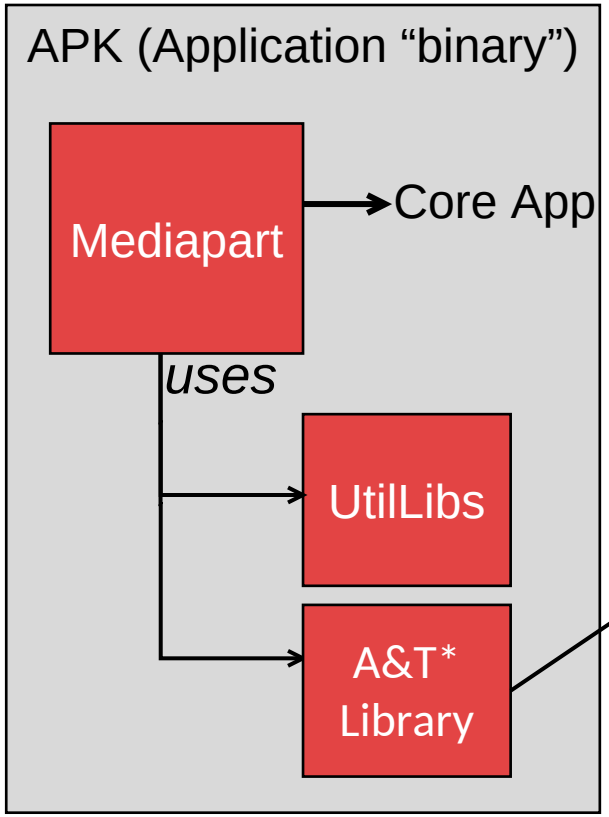
- Freely ignore user added certificates
- Can detect changes in the system
- Can circumvent the built-in trust system

<u>Apps implement heavy obfuscation :</u>

- DCL (Dynamic Code Loading)
- Certificate pinning
- Root detection

Android Apps

Mediapart *Application*



*A&T = Advertising and tracking

Android application are directly packaged with Ad/tracker libraries

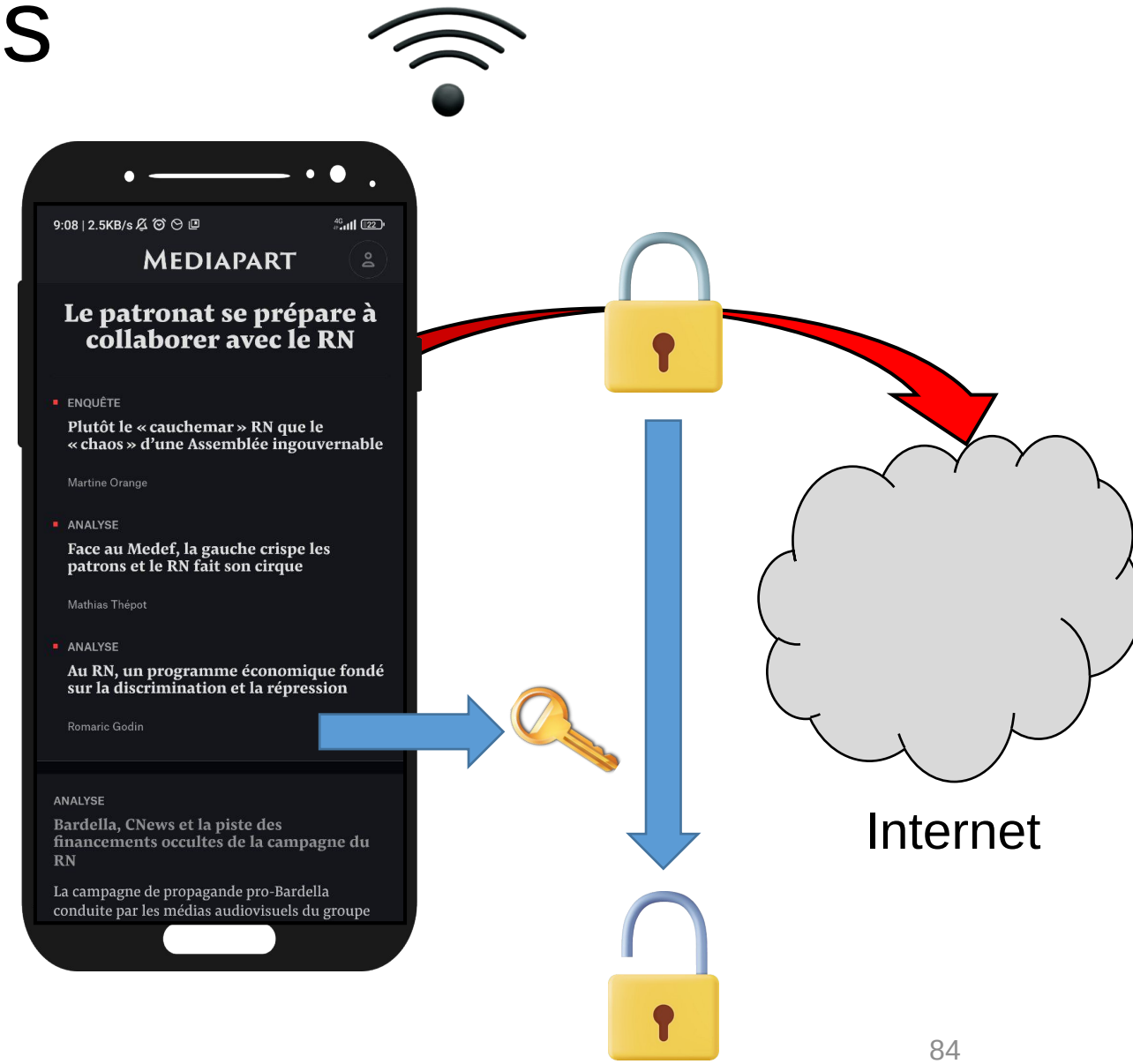
... and many more

Dynamically test apps

Dynamically, it’s not as easy :

Sol1. Use a proxy

Sol2. Modify App or system
to collect decryption keys



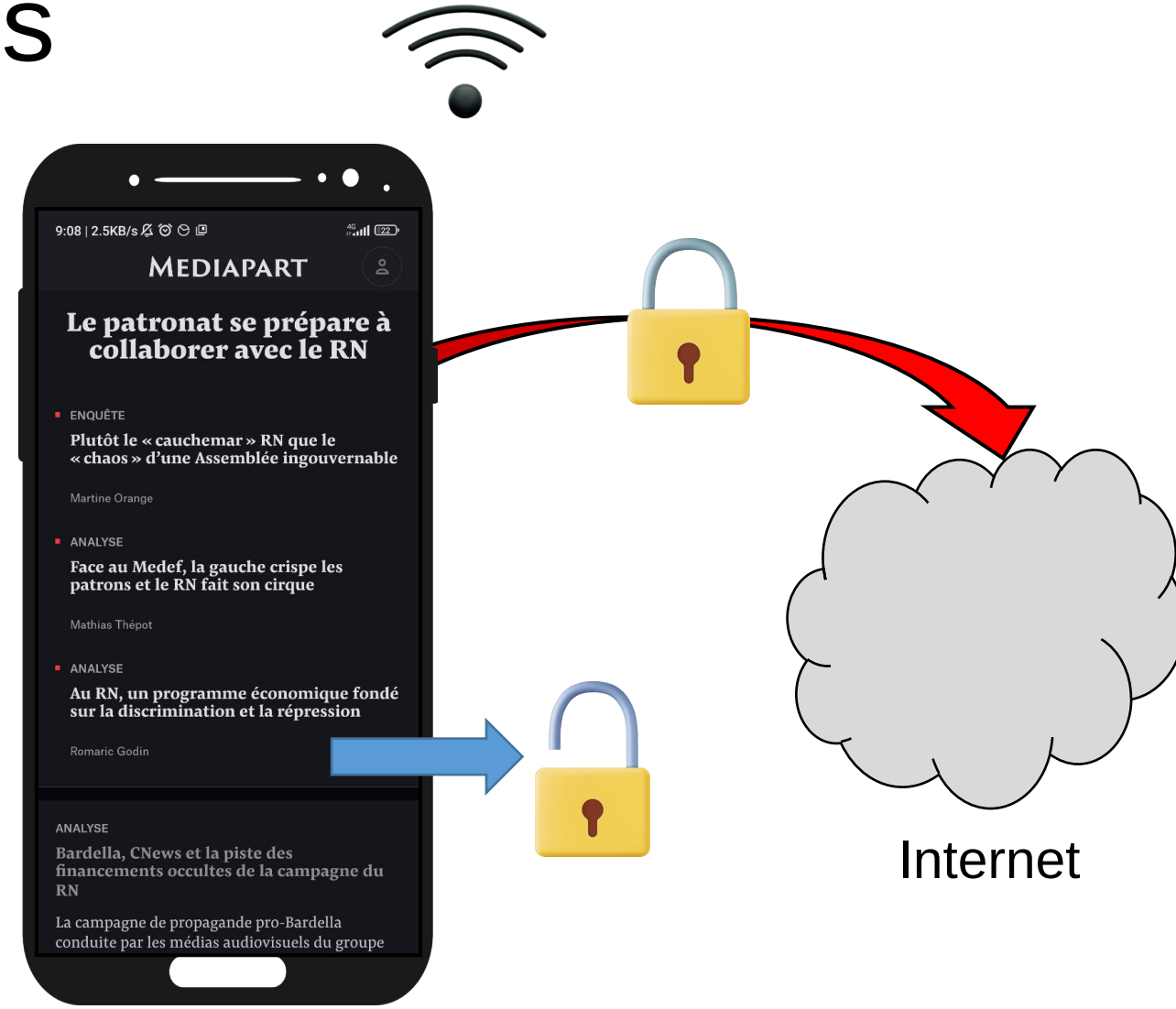
Dynamically test apps

Dynamically, it’s not as easy :

Sol1. Use a proxy

Sol2. Modify App or system
to collect decryption keys

Sol3. Modify App or system
to collect unencrypted traffic



Google choosing which devices to trust

Fingerprints

?

Hardware backed keys