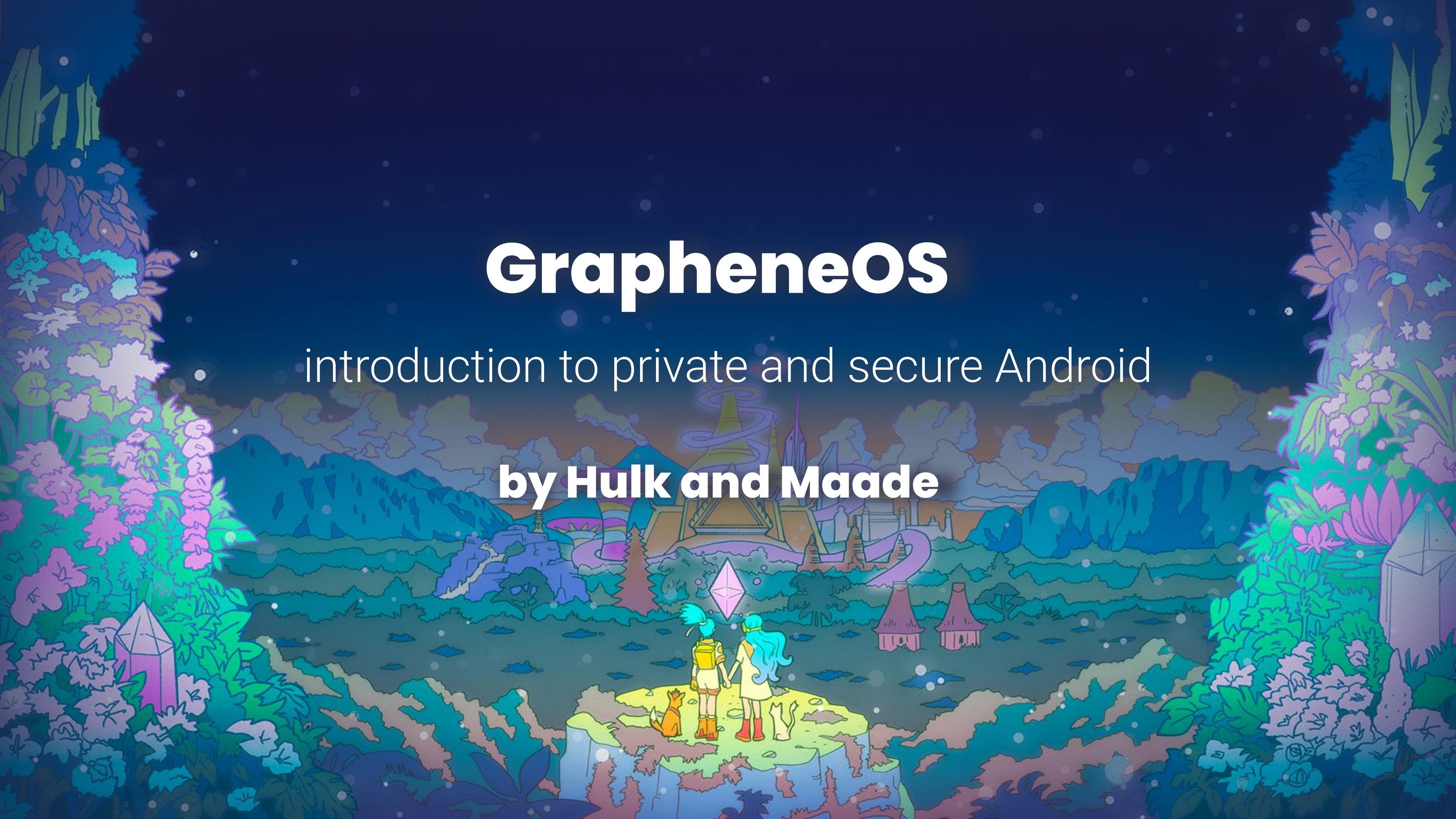


GrapheneOS

introduction to private and secure Android

by Hulk and Maade

GrapheneOS

Who are we?

- International **non-profit** organization working on **open-source** projects
- Our goal is to make the most **private** and **secure** mobile OS
- Lots of our hardening features successfully landed **upstream**
- Serving **more than 250k** users worldwide

Enhancing Android Security

Hardened Allocator

Mitigates some memory corruptions

USB-C port Control

Defending against physical attacks

Memory Tagging

Detecting various forms of memory corruption

Sandboxing Google Play

Reducing privileges

Enhancing Android Security

Hardened Allocator

Mitigates some memory corruptions

USB-C port Control

Defending against physical attacks

Memory Tagging

Detecting various forms of memory corruption

Sandboxing Google Play

Reducing privileges

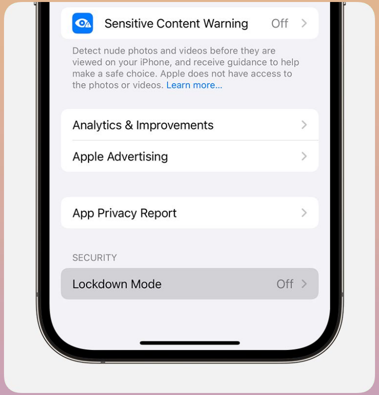
Focus of this presentation

Quick Security Tip for Apple Users

Turn On Lockdown Mode

1. Open the Settings app
2. Go to Privacy & Security
3. Scroll down to Lockdown Mode
4. Turn On Lockdown Mode

Source: support.apple.com/105120



Sensitive Content Warning Off >

Detect nude photos and videos before they are viewed on your iPhone, and receive guidance to help make a safe choice. Apple does not have access to the photos or videos. Learn more...

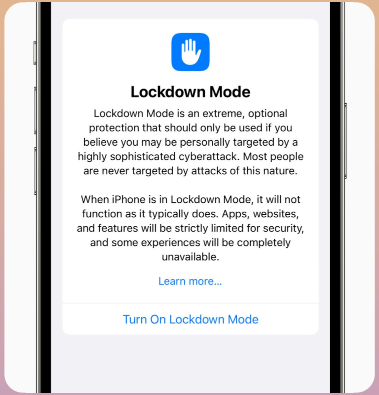
Analytics & Improvements >

Apple Advertising >

App Privacy Report >

SECURITY

Lockdown Mode Off >



Lockdown Mode

Lockdown Mode is an extreme, optional protection that should only be used if you believe you may be personally targeted by a highly sophisticated cyberattack. Most people are never targeted by attacks of this nature.

When iPhone is in Lockdown Mode, it will not function as it typically does. Apps, websites, and features will be strictly limited for security, and some experiences will be completely unavailable.

Learn more...

Turn On Lockdown Mode

Enhancing Android Security

Hardened Allocator

Mitigates some memory corruptions

USB-C port Control

Defending against physical attacks

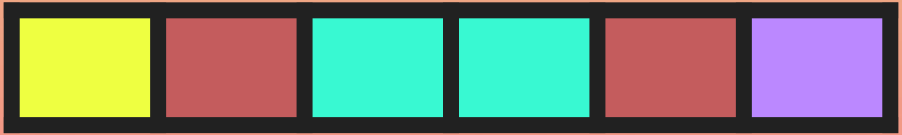
Memory Tagging

Detecting various forms of memory corruption

Sandboxing Google Play

Reducing privileges

Focus of this presentation



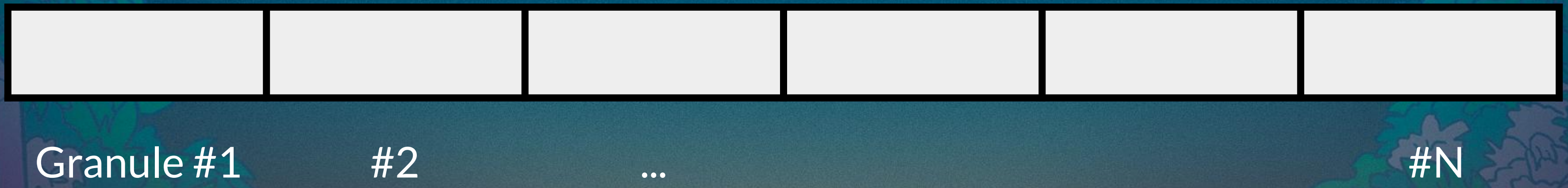
Memory Tagging

A technique to detect (and prevent) memory corruptions

DC7 SEA

Concept of Memory Tagging

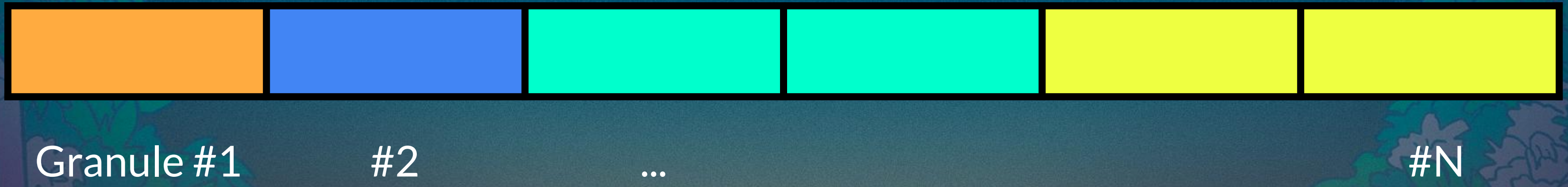
- Memory is divided into **granules** of a fixed size



Source: Andrey Konovalov (xairy.io)

Concept of Memory Tagging

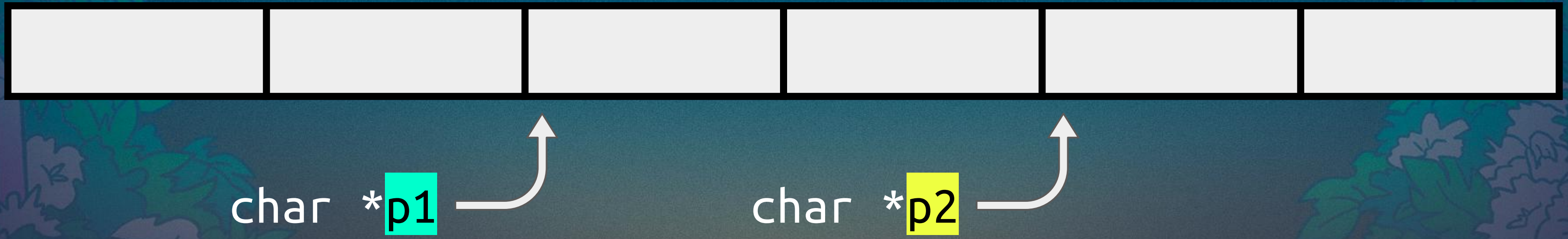
- Memory is divided into **granules** of a fixed size
- Each memory **granule** has a **tag** (aka **color**)



Source: Andrey Konovalov (xairy.io)

Concept of Memory Tagging

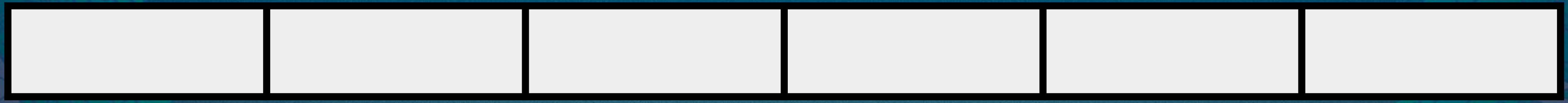
- Memory is divided into **granules** of a fixed size
- Each memory **granule** has a **tag** (aka **color**)
- Every **pointer** has a **tag**



Source: Andrey Konovalov (xairy.io)

Concept of Memory Tagging

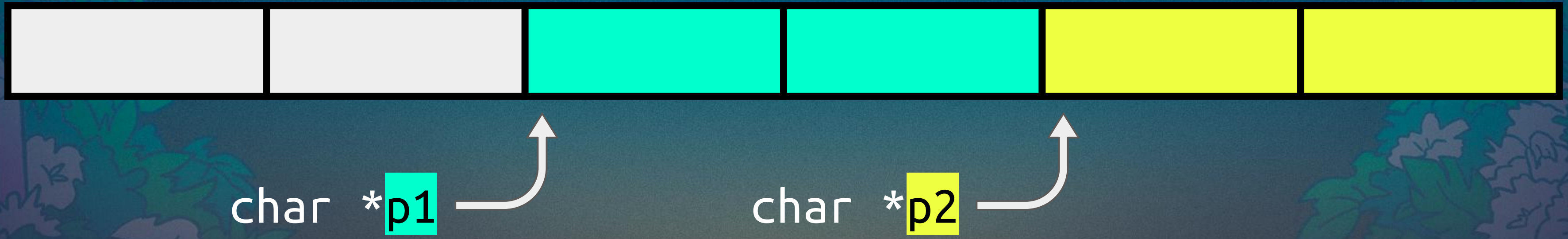
- Memory is divided into **granules** of a fixed size
- Each memory **granule** has a **tag** (aka **color**)
- Every **pointer** has a **tag**
- On allocation, both memory and pointer get a matching **random tag**



Source: Andrey Konovalov (xairy.io)

Concept of Memory Tagging

- Memory is divided into **granules** of a fixed size
- Each memory **granule** has a **tag** (aka **color**)
- Every **pointer** has a **tag**
- On allocation, both memory and pointer get a matching **random tag**



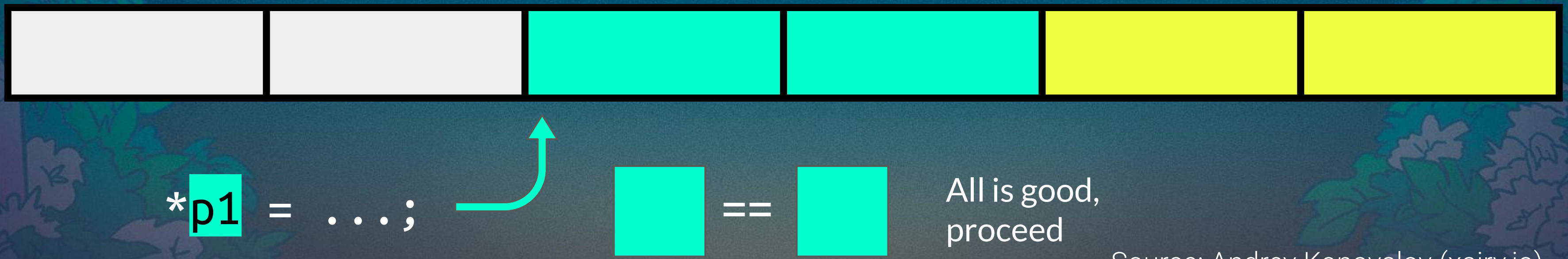
char *p1

char *p2

Source: Andrey Konovalov (xairy.io)

Concept of Memory Tagging

- Memory is divided into **granules** of a fixed size
- Each memory **granule** has a **tag** (aka **color**)
- Every **pointer** has a **tag**
- On allocation, both memory and pointer get the same **random tag**
- On pointer dereference, **pointer tag** must match **memory tag**



`*p1 = ...;`

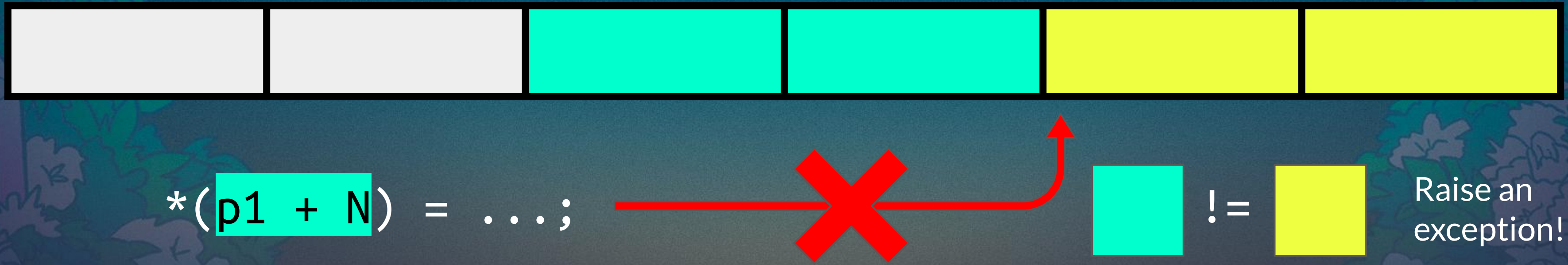
==

All is good,
proceed

Source: Andrey Konovalov (xairy.io)

Concept of Memory Tagging

- Memory is divided into **granules** of a fixed size
- Each memory **granule** has a **tag** (aka **color**)
- Every **pointer** has a **tag**
- On allocation, both memory and pointer get the same **random tag**
- On pointer dereference, **pointer tag** must match **memory tag**

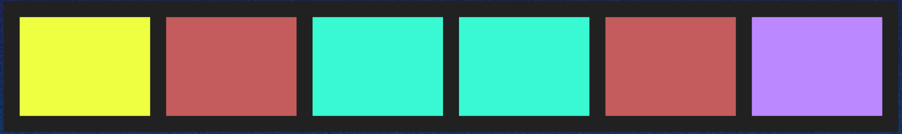


```
*(p1 + N) = ...;
```

🟩 != 🟨

Raise an exception!

Positive Impact of MTE



Linear overflows are deterministically detected

(100%)

Use-after-free are deterministically detected

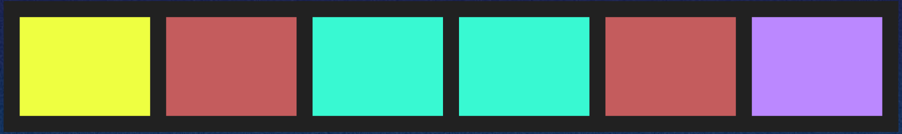
(100%)

Probabilistic protection against **various forms** of memory corruption

(~93% to 100%)

Source: github.com/GrapheneOS/hardened_malloc

(Not so) Positive Impact of MTE



Storing memory tags

+3% RAM usage

Performance impact

~2 to 5%

Only available on 8th and 9th gen Pixel devices

as of Nov 2024

Source: github.com/GrapheneOS/hardened_malloc

Enhancing Android Security

Hardened Allocator

Mitigates some memory corruptions

USB-C port Control

Defending against physical attacks

Memory Tagging

Detecting various forms of memory corruption

Sandboxing Google Play

Reducing privileges

Focus of this presentation

DC7 SEA

Google Play

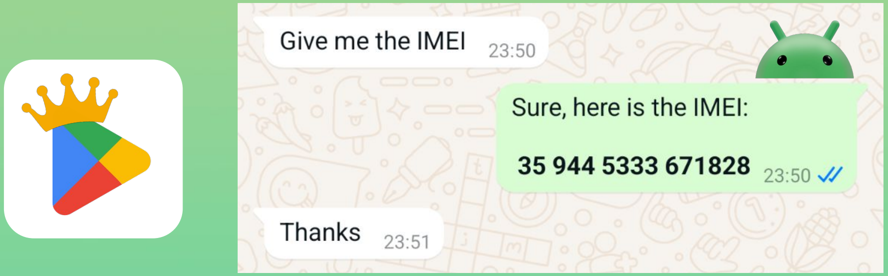
Sandboxed Google Play

Why restrict Google Play?

On Stock Android:

- Google Play can **silently install** any app without user's consent
- Google Play can access **private data**
- Users **cannot revoke** these high-privileges from Google Play

If an attacker gains **control** over Google Play it would be **GAME OVER**

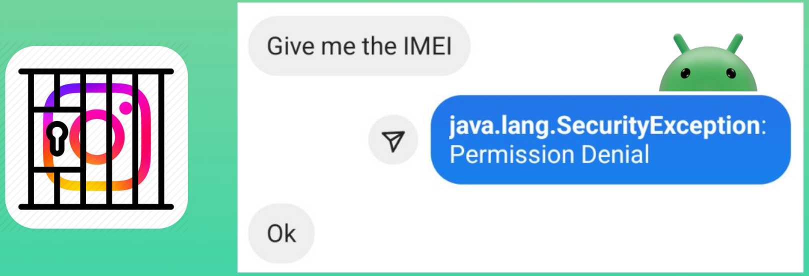


Give me the IMEI 23:50

Sure, here is the IMEI:

35 944 5333 671828 23:50

Thanks 23:51



Give me the IMEI

java.lang.SecurityException: Permission Denial

Ok

Can you live without Google Play?

App Compatibility

Lots of 3rd party apps need Google Play services to run properly

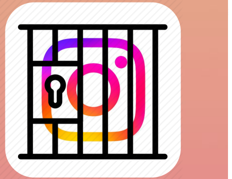
Delayed Notifications

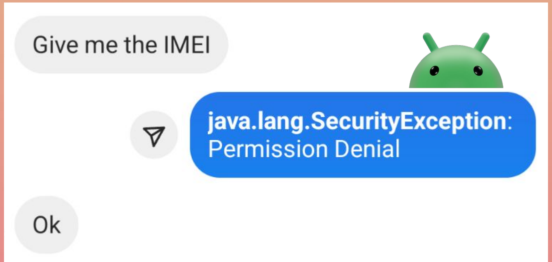
WhatsApp, Telegram, Instagram notifications will be delayed

Location Issues

Will not be able to use Google Location Accuracy nor Find My Device

Many people do NEED Google Play

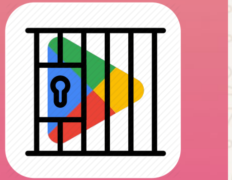


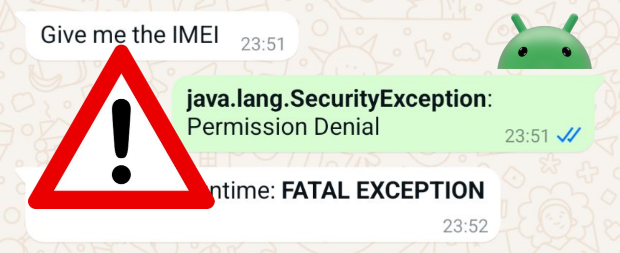


Give me the IMEI

java.lang.SecurityException: Permission Denial

Ok





Give me the IMEI 23:51

java.lang.SecurityException: Permission Denial 23:51

ntime: FATAL EXCEPTION 23:52

"Harsh" Privilege Reduction

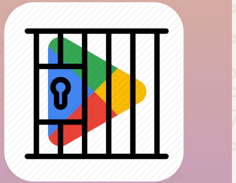
- Third-party apps *know* how to handle exceptions
- Google Play *does not know* how to handle exceptions

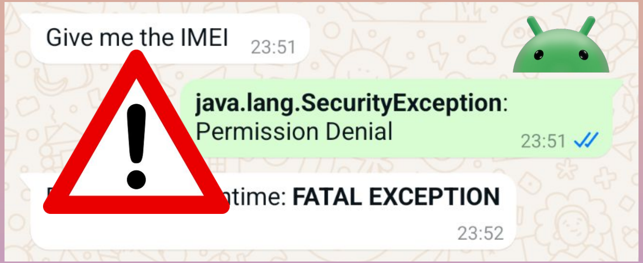
How to solve this? Be gentle

“Gentle” Privilege Reduction

- Google Play *does not know* how to handle exceptions

- We can *trick* Google Play into *thinking* that it has all of the privileges

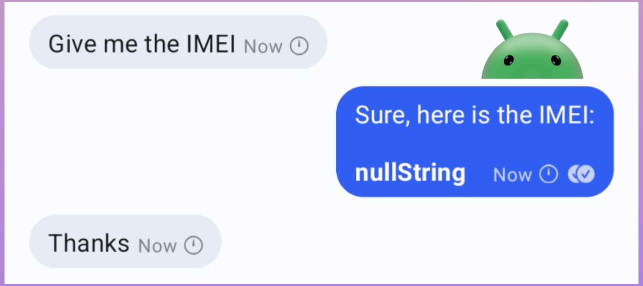




Give me the IMEI 23:51

java.lang.SecurityException: Permission Denial 23:51

ntime: **FATAL EXCEPTION** 23:52



Give me the IMEI Now

Sure, here is the IMEI:

nullString Now

Thanks Now

Sandboxed Google Play

App Compatibility

Perfect compatibility even with apps that heavily rely on Google Play services

Push Notifications

All notifications arrive on time

Location Accuracy

Users can opt-in to using Google Location Accuracy and Find My Device

Source: grapheneos.org/usage#sandboxed-google-play

CONTACT US

t.me/Hulk_GrapheneOS
t.me/Daydream9487

grapheneos.org/contact