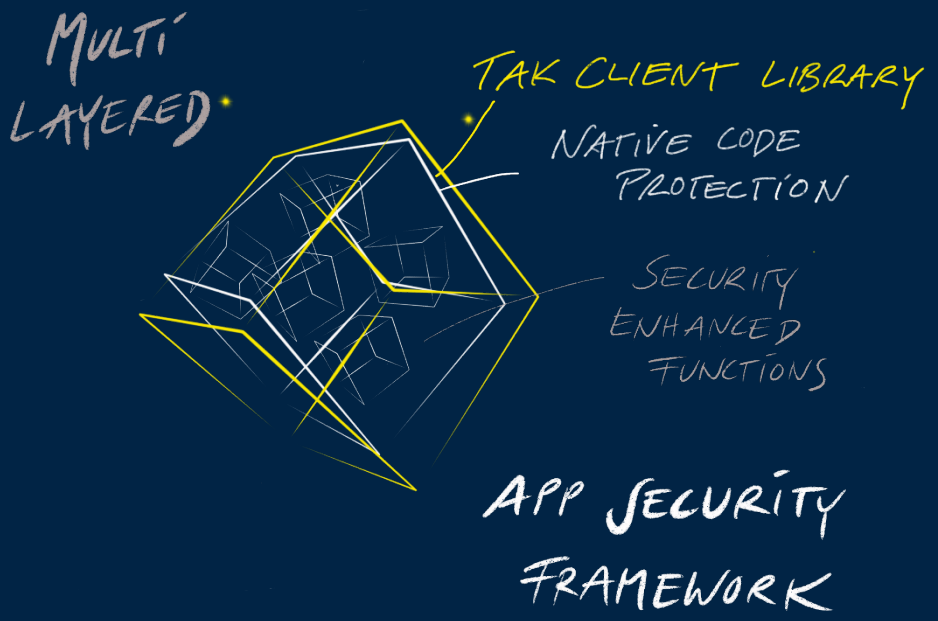
build
Mobile Security

# Mobile App Security

An introduction




**Marc Obrador**

# Who am I?


## Marc Obrador

Co-founder & Head of Product Architecture @ Build38

- Barcelona
- marc@build38.com
- @marcobrador
- /in/marc-obrador

# Agenda

1. Introduction

2. Some Common Threads
    1. Man-In-The-Middle
    2. App Tampering & Repackaging
    3. Root / Jailbreak

3. Recap

# Agenda

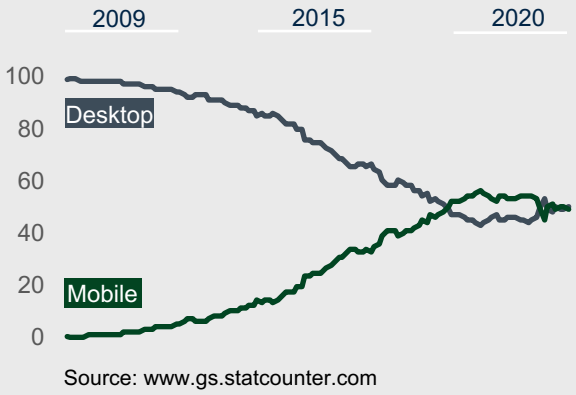
## 1. Introduction

## 2. Some Common Threads

1. Man-In-The-Middle
2. App Tampering & Repackaging
3. Root / Jailbreak

## 3. Recap

# Why Mobile App Security?

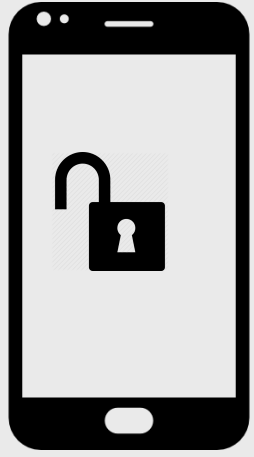



Source: www.gs.statcounter.com

Mobile-first world

Regulation

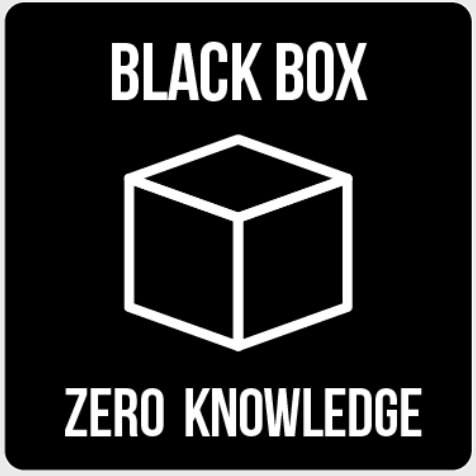
(depending on market)

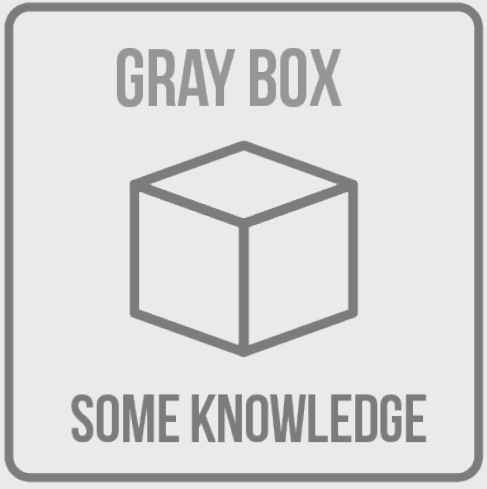


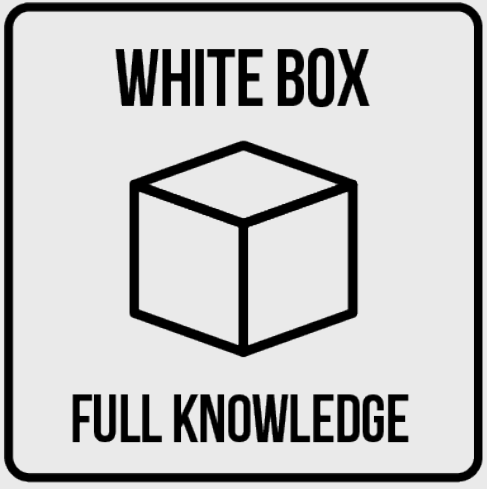
Smartphone =

*untrusted device*

# Mobile AppSec vs “traditional” Cyber Securtity



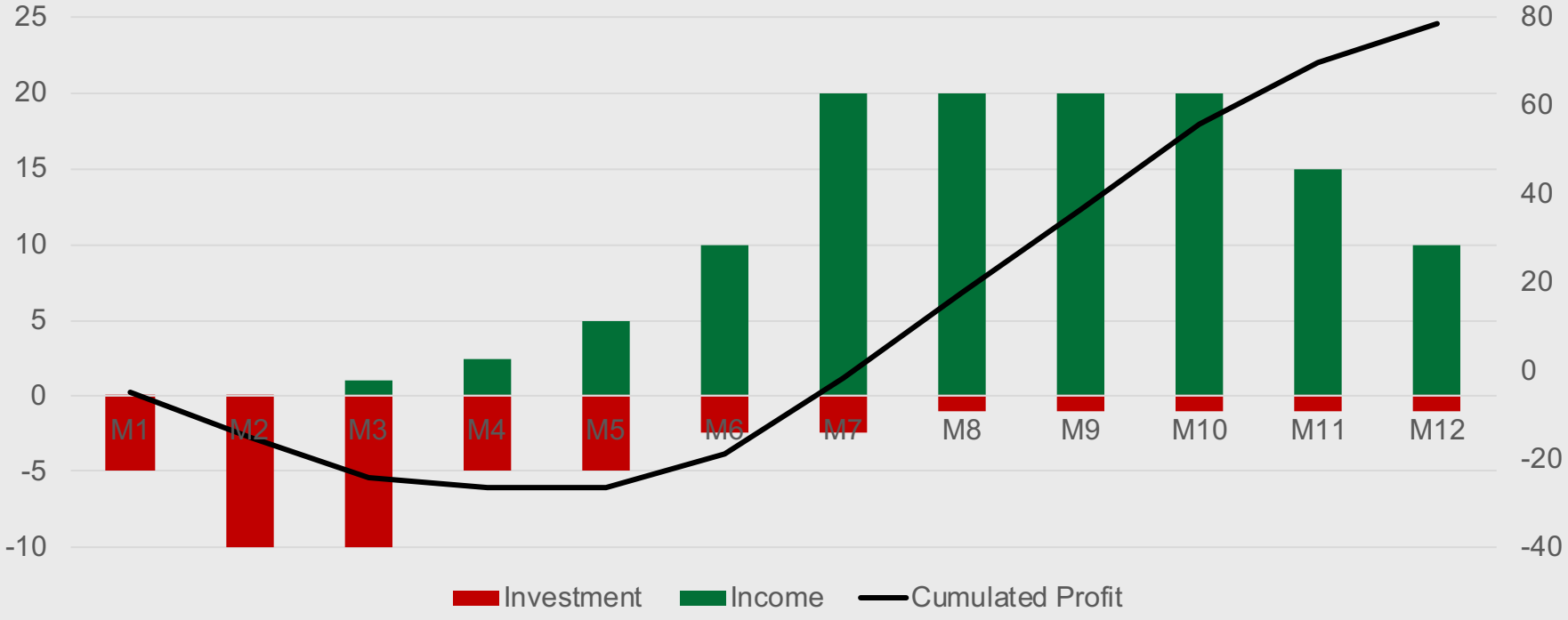



# Is there anything I can do?

Let’s first switch our perspective

# The hacker’s perspective

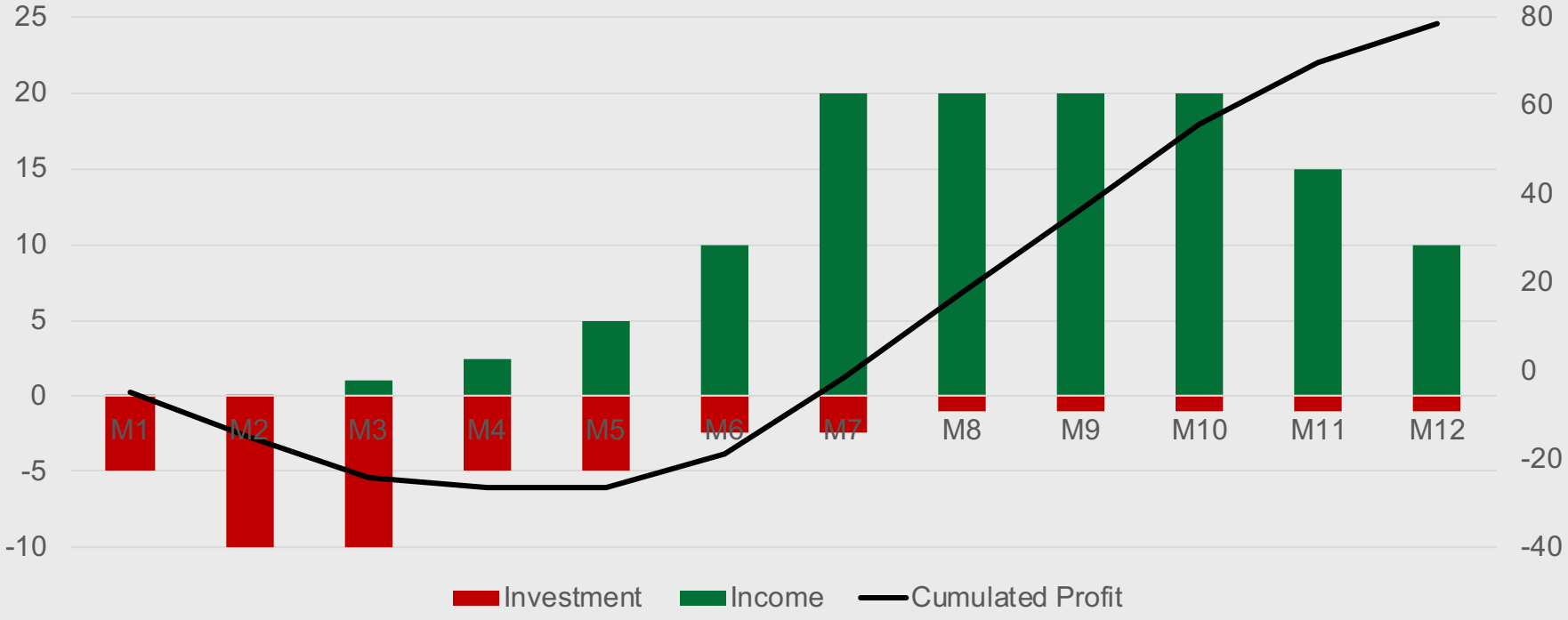


# Is there anything I can do?

## Is there anything I can do?

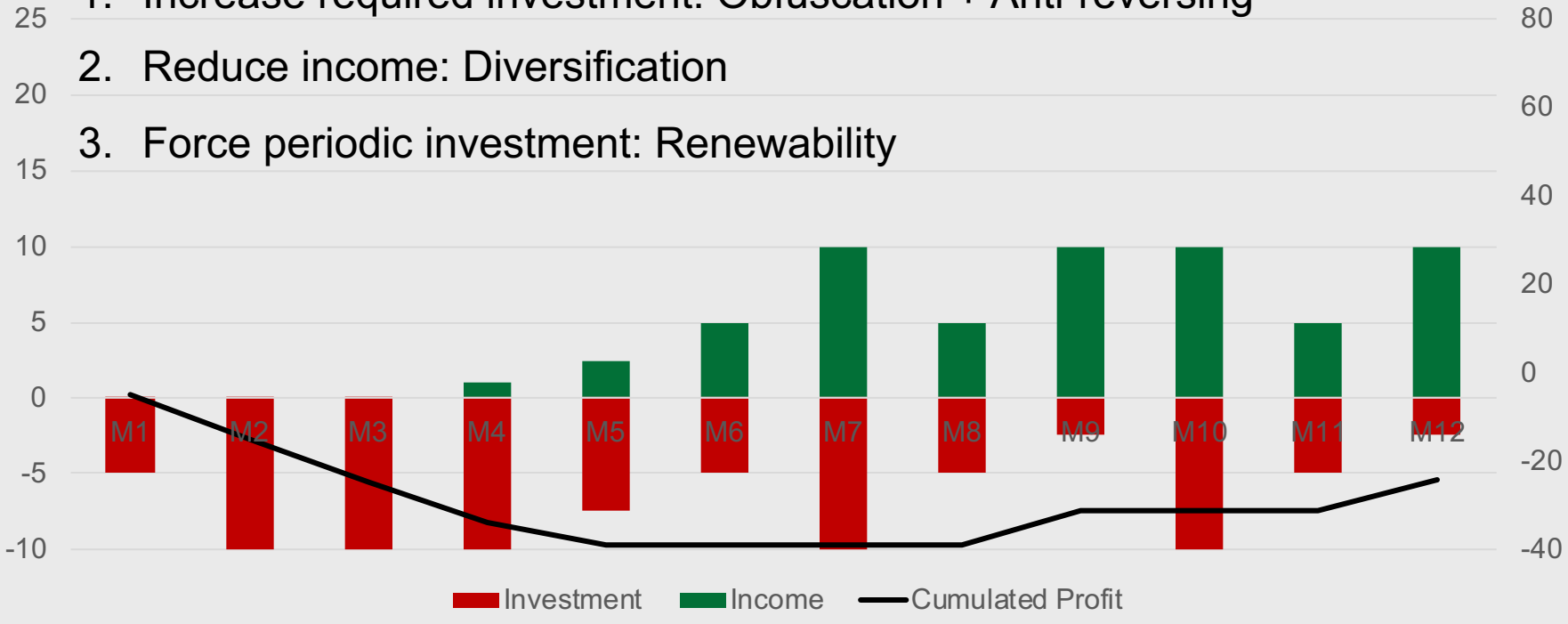
# Make it unattractive for the hacker

# Is there anything I can do?



# Is there anything I can do?

1. Increase required investment: Obfuscation + Anti-reversing
2. Reduce income: Diversification
3. Force periodic investment: Renewability



# Things to protect

User Data

Business Data / IP

DRM

# Agenda

1. Introduction

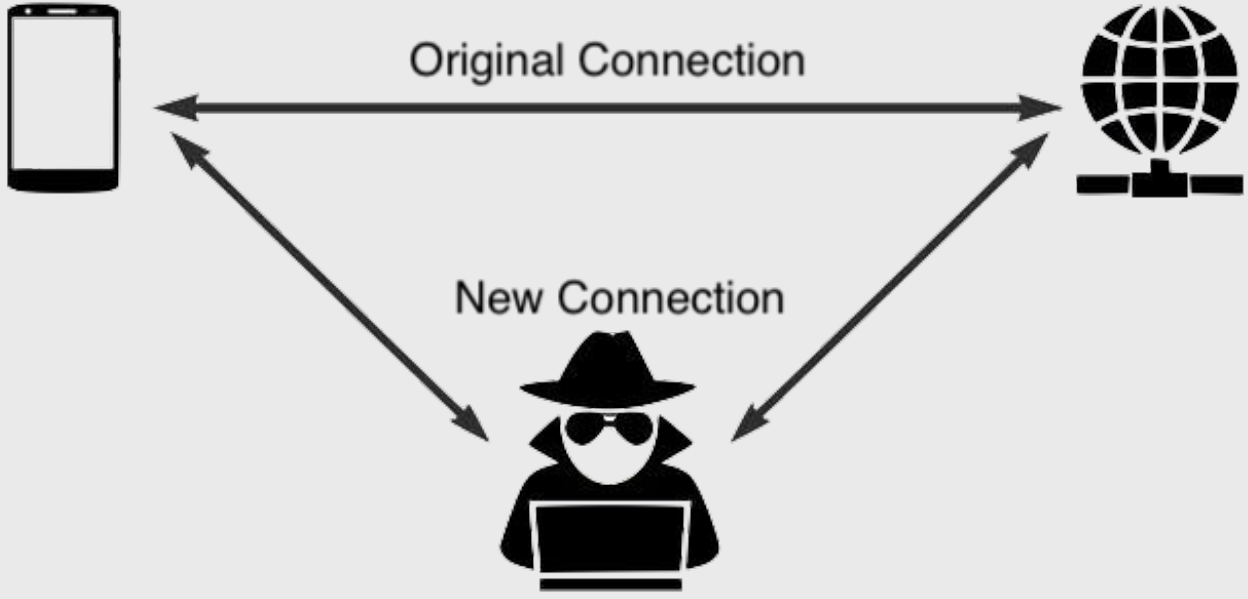
2. Some Common Threads
    1. Man-In-The-Middle
    2. App Tampering & Repackaging
    3. Root / Jailbreak

3. Recap

# Agenda

1. Introduction
2. Some Common Threads
    1. Man-In-The-Middle
    2. App Tampering & Repackaging
    3. Root / Jailbreak
3. Recap

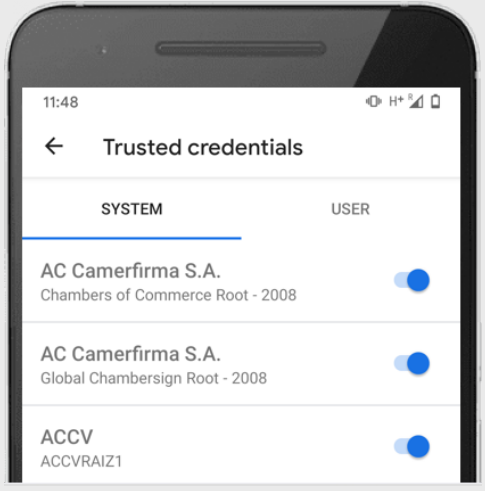
# MITM



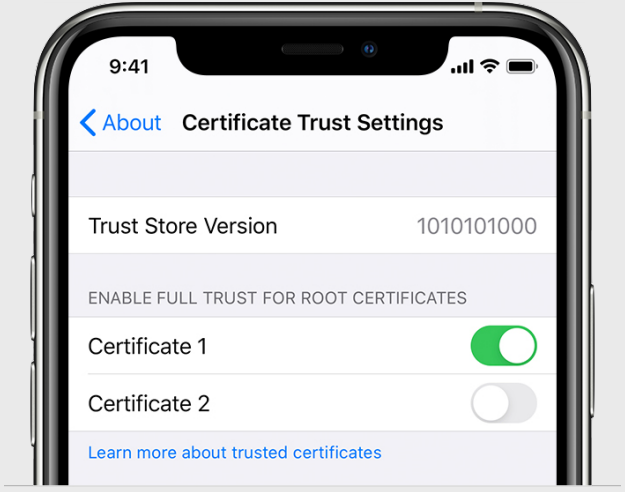

HTTPS is assumed!

# MITM with HTTPS?

No, if Certificate Pinning is used




Android: depends on OEM

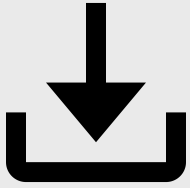



iOS: requires social engineering

# Agenda

1. Introduction
2. Some Common Threads
    1. Man-In-The-Middle
    2. App Tampering & Repackaging
    3. Root / Jailbreak
3. Recap

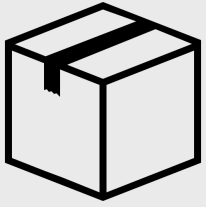
# What is it?



1. Download

2. Unpack

3. Modify



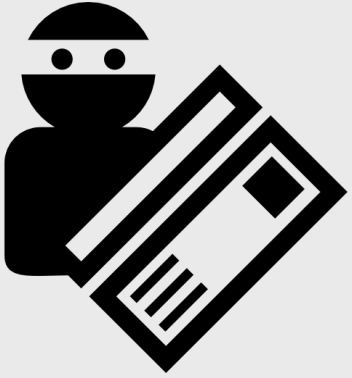
4. Repack

5. Distribute

# But, why?

Cheating on games


Getting paid features for free



Stealing user data

# Android: apktool + *smali* code

```
.method protected onCreate(Landroid/os/Bundle;)V
    .locals 2
    .param p1, "savedInstanceState"    # Landroid/os/Bundle;

    .line 12
    invoke-super {p0, p1}, Landroidx/appcompat/app/AppCompatActivity;->onCreate(Landroid/os/Bundle;)V

    .line 13
    const v0, 0x7f0a001c

    invoke-virtual {p0, v0}, Lcom/build38/smali/MainActivity;->setContentView(I)V

    .line 15
    const v0, 0x7f0700a9

    invoke-virtual {p0, v0}, Lcom/build38/smali/MainActivity;->findViewById(I)Landroid/view/View;

    move-result-object v0

    check-cast v0, Landroid/widget/TextView;

    const-string v1, "Hello, Wien!"

    invoke-virtual {v0, v1}, Landroid/widget/TextView;->setText(Ljava/lang/CharSequence;)V

    .line 16
    return-void
.end method
```

# iOS: dynamic library injection

# Protecting against app repackaging

```
(A)
    function setText(data) {
    document.getElementById("myDiv").innerHTML = data;
    }
```

```
(B)
    function ghds3x(n) {
    h = "\x69\u006En\u0065r\x48T\u004DL";
    a="s c v o v d h e , n i";x=a.split(" ");b="gztxleWentBsyf";
 r=b.replace("z",x[7]).replace("x","E").replace("s","").replace("f","I")
    ["repl" + "ace"]("W","m")+"d";
c="my"+String.fromCharCode(68)+x[10]+"v";
    s=x[5]+x[3]+x[1]+"um"+x[7]+x[9]+"t";d=this[s][r](c);if(+!![])
    { d[h]=n; } else { d[h]=c; } }
```

Obfuscation

Detect it

# Agenda

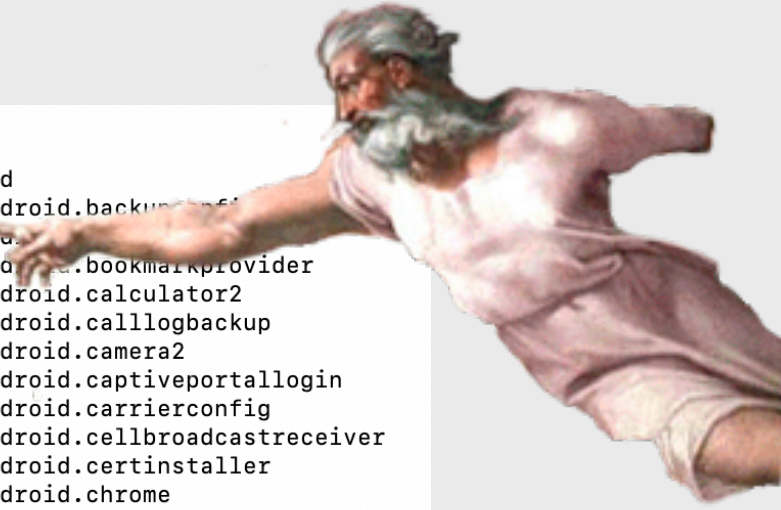
1. Introduction

2. Some Common Threads
    1. Man-In-The-Middle
    2. App Tampering & Repackaging
    3. **Root / Jailbreak**

3. Recap

# The ”sandbox” model

```
generic_x86:/ # ls -l /data/data/
total 808
drwx------  4 system system 4096 2019-05-29 20:03 android
drwx------  4 u0_a2  u0_a2  4096 2019-05-29 20:03 com.android.backup.confi
drwx------  4 u0_a34 u0_a34 4096 2019-05-29 20:03 com.and
drwx------  4 u0_a36 u0_a36 4096 2019-05-29 20:03 com.and      .bookmarkprovider
drwx------  4 u0_a47 u0_a47 4096 2019-05-29 20:03 com.android.calculator2
drwx------  4 u0_a1  u0_a1  4096 2019-05-29 20:03 com.android.calllogbackup
drwx------  5 u0_a44 u0_a44 4096 2019-05-29 20:04 com.android.camera2
drwx------  4 u0_a39 u0_a39 4096 2019-05-29 20:03 com.android.captiveportallogin
drwx------  4 u0_a3  u0_a3  4096 2019-05-29 20:03 com.android.carrierconfig
drwx------  4 u0_a6  u0_a6  4096 2019-05-29 20:03 com.android.cellbroadcastreceiver
drwx------  4 u0_a38 u0_a38 4096 2019-05-29 20:03 com.android.certinstaller
drwx------ 12 u0_a49 u0_a49 4096 2019-05-29 20:08 com.android.chrome
drwx------  4 u0_a40 u0_a40 4096 2019-05-29 20:03 com.android.companiondevicemanager
drwx------  5 u0_a13 u0_a13 4096 2019-05-29 20:04 com.android.contacts
drwx------  4 u0_a41 u0_a41 4096 2019-05-29 20:03 com.android.cts.ctsshim
drwx------  4 u0_a8  u0_a8  4096 2019-05-29 20:03 com.android.cts.priv.ctsshim
drwx------  4 u0_a42 u0_a42 4096 2019-05-29 20:03 com.android.customlocale2
drwx------  4 u0_a7  u0_a7  4096 2019-05-29 20:03 com.android.defcontainer
drwx------  4 u0_a43 u0_a43 4096 2019-05-29 20:03 com.android.development
drwx------  6 u0_a10 u0_a10 4096 2019-05-29 20:11 com.android.documentsui
drwx------  4 u0_a33 u0_a33 4096 2019-05-29 20:03 com.android.dreams.basic
drwx------  4 u0_a45 u0_a45 4096 2019-05-29 20:03 com.android.egg
```



# Root / Jailbreak Detection


/scottyab/rootbeer

/KimChangYoun/rootbeerFresh

/Stericson/RootTools

/avltree9798/isJailbroken

/thii/DTTJailbreakDetection

# What to do if Root / Jailbreak is found?

# What to do if Root is found?


Sources:

- https://techcrunch.com/2019/08/29/google-iphone-secretly-hacked/
- https://googleprojectzero.blogspot.com/2019/08/a-very-deep-dive-into-ios-exploit.html

# What to do if Root is found?

Nothing

Restrict some sensitive functionality

Deny service

Design your security model assuming that root can (and will) happen

# Agenda

1. Introduction

2. Some Common Threads
    1. Man-In-The-Middle
    2. App Tampering & Repackaging
    3. Root / Jailbreak

3. Recap

# Recap

- 100% protection does not exist – aim for “good enough”
- Certificate Pinning is a good idea
- Apps can be reverse engineered and repackaged
  - Move security-relevant logic to backend or write it in native C
- Root can be really bad – come up with a plan

build
Mobile Security

# Thank you!

# Any questions?