https://www.youtube.com/watch?v=tELZEPcgKkE

Sławomir Jasek
slawomir.jasek@securing.pl
slawekja

Can't Touch This: Cloning Any Android HCE Contactless Card

HackInTheBox Amsterdam, 13.04.2017

Cloning

1996: Dolly the sheep

Cloning

1996: Dolly the sheep

2001: cat „CopyCat”

2003: horse

Humans? Why no humans?

I. Commercial – no commercial interest in industries

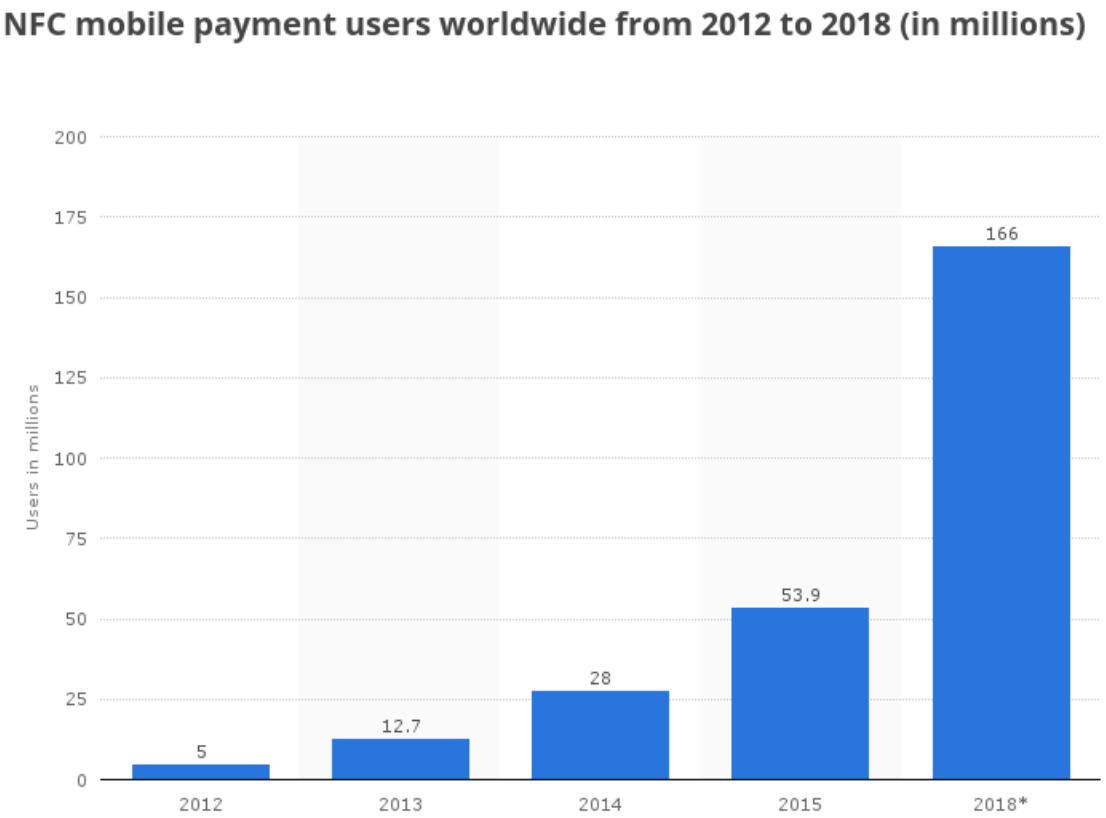
II. Ethical/legal – beliefs, laws…

III. Technical - pets easy, primates very hard

Are we sure?

2017 – mobile contactless payment cards cloning?

I. Commercial



https://www.statista.com/statistics/461512/nfc-mobile-payment-users-worldwide/

2017 – mobile contactless payment cards cloning?

I. Commercial

II. Ethical/legal

III. Technical

https://www.statista.com/statistics/461512/nfc-mobile-payment-users-worldwide/

Technical countermeasures

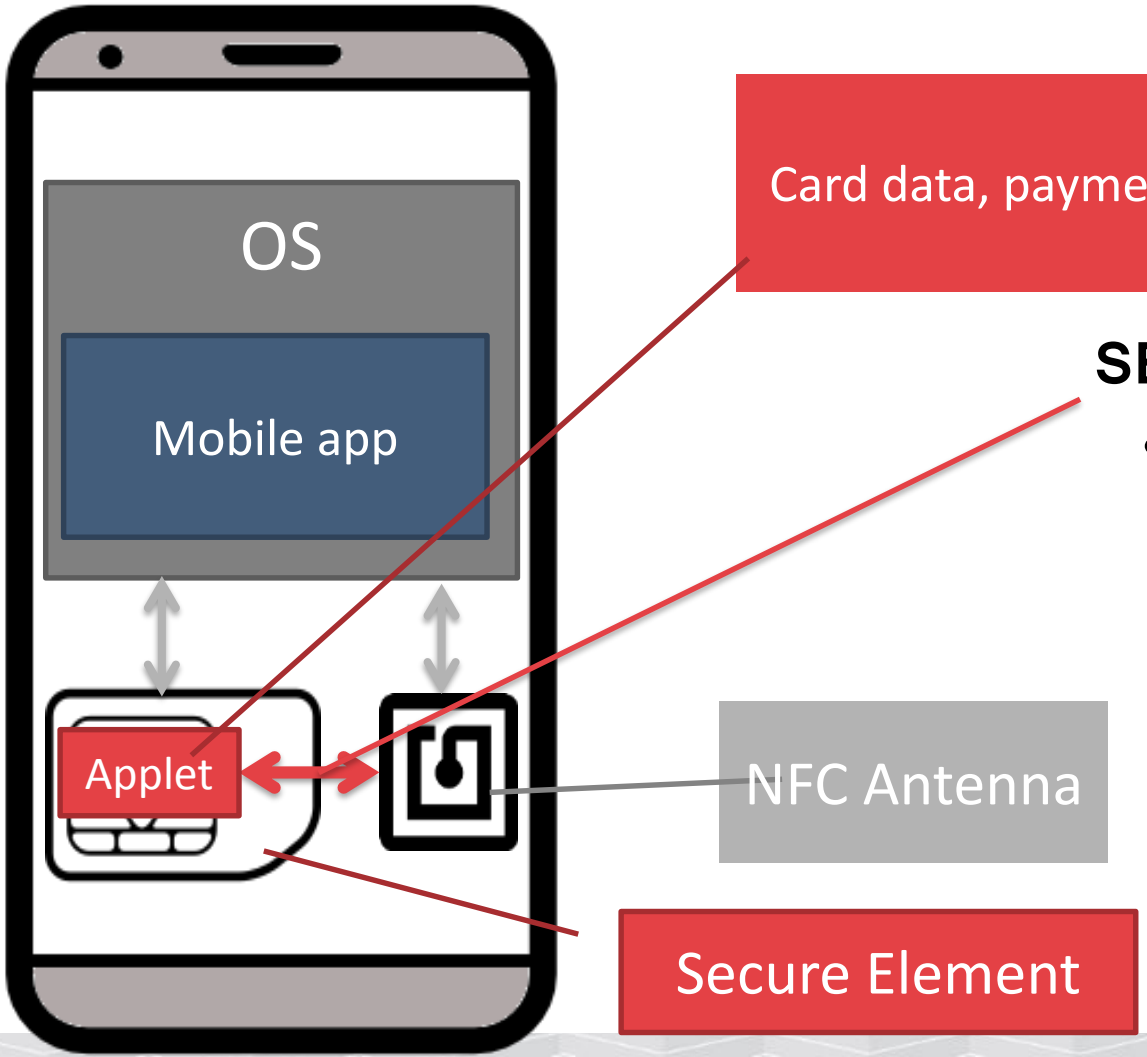
www.wordclouds.com

https://www.flickr.com/photos/un_photo/6872103103/

https://www.flickr.com/photos/mujitra/8606262790/

HCE TECHNOLOGY

„Secure Element” (since 2007)



SE communicates directly with NFC

- Apps and OS have no access to card data and to communication during payment.

SE dominance hierarchy clashes

Banks vs mobile operators, handset manufacturers, payment service providers...

Painful process

- special SIM required
- limited support

https://www.flickr.com/photos/jsouthorn/6616455243/

Google vs Isis Wallet

2011: Google Wallet with Galaxy Nexus embedded SE

Isis wallet (AT&T, Verizon, T-Mobile) - blocked Google Wallet for their devices.

Google: we will go our way - without SE.

Host Card Emulation

Android >=4.4, Blackberry OS, Windows Phone

No need for troublesome Secure Element, moved to „cloud”.

Software emulates contactless smart card.

How does it work?

Demo

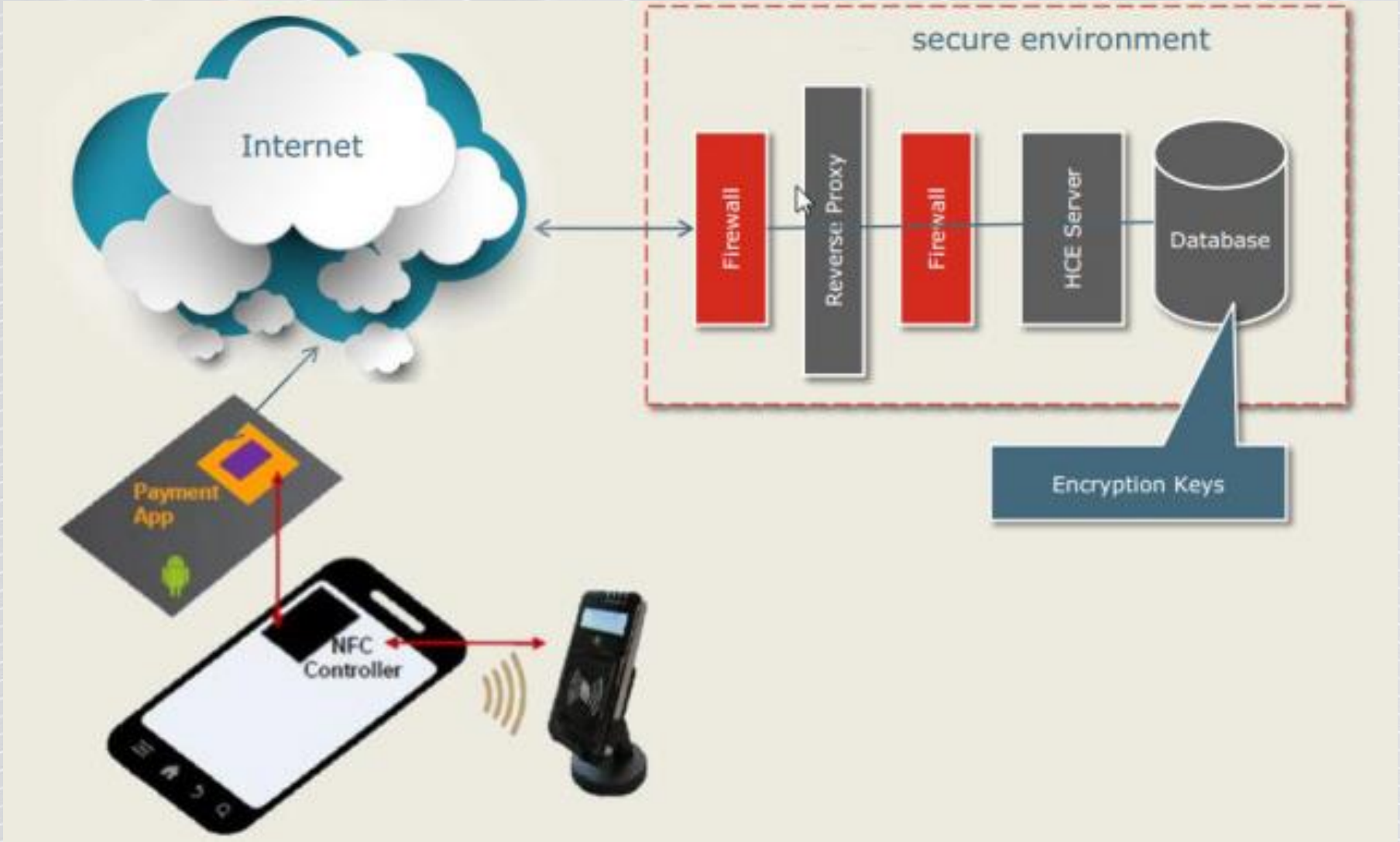
INTEGRATION

How to embed it in mobile app?

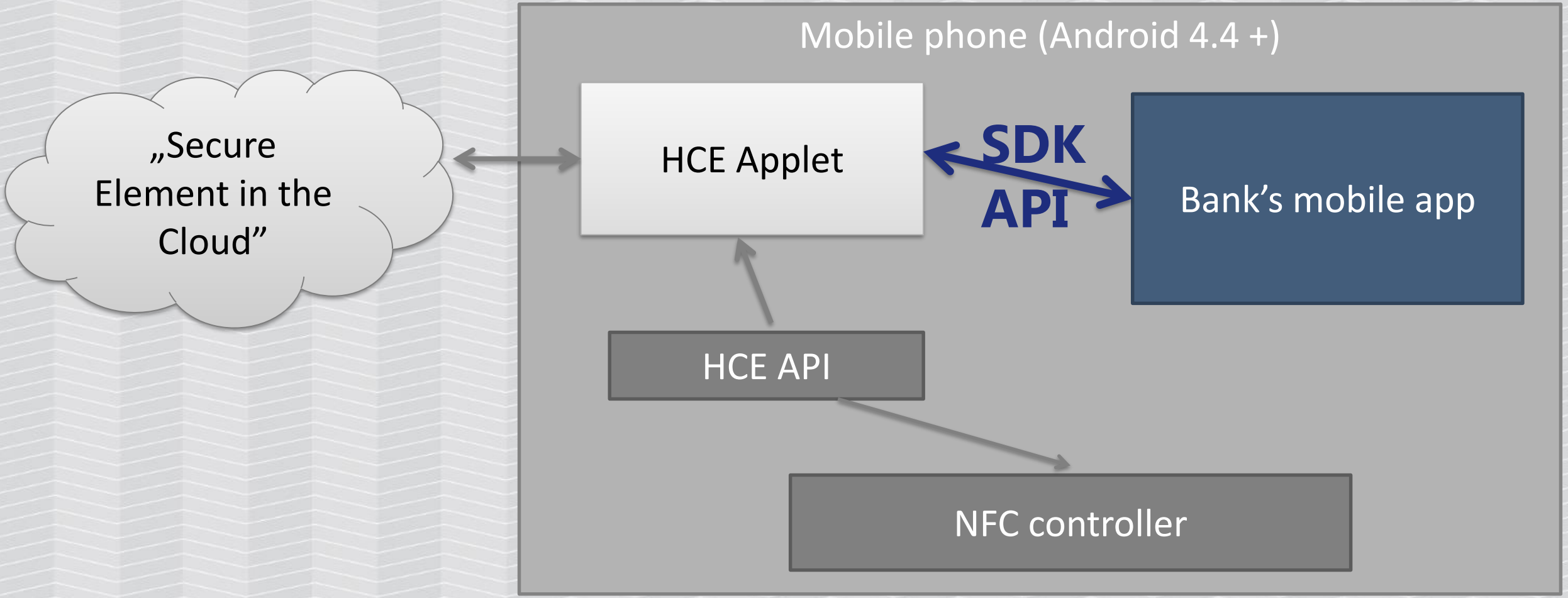
- Own implementation
- External, „blackbox” library
  - Visa, Mastercard SDK
  - several other products

https://www.flickr.com/photos/lluniau_rich/580859948/

Vendors' doc


secure environment
Internet
Firewall
Reverse Proxy
Firewall
HCE Server
Database
Encryption Keys
Payment App
NFC Controller

Vendors' doc


Mobile phone (Android 4.4 +)
„Secure Element in the Cloud"
HCE Applet
SDK
API
Bank's mobile app
HCE API
NFC controller

Sławomir Jasek

Enjoy appsec (dev, break, build...) since 2003.

Pentesting, consultancy, training - web, mobile, embedded...

Significant part of time for research.

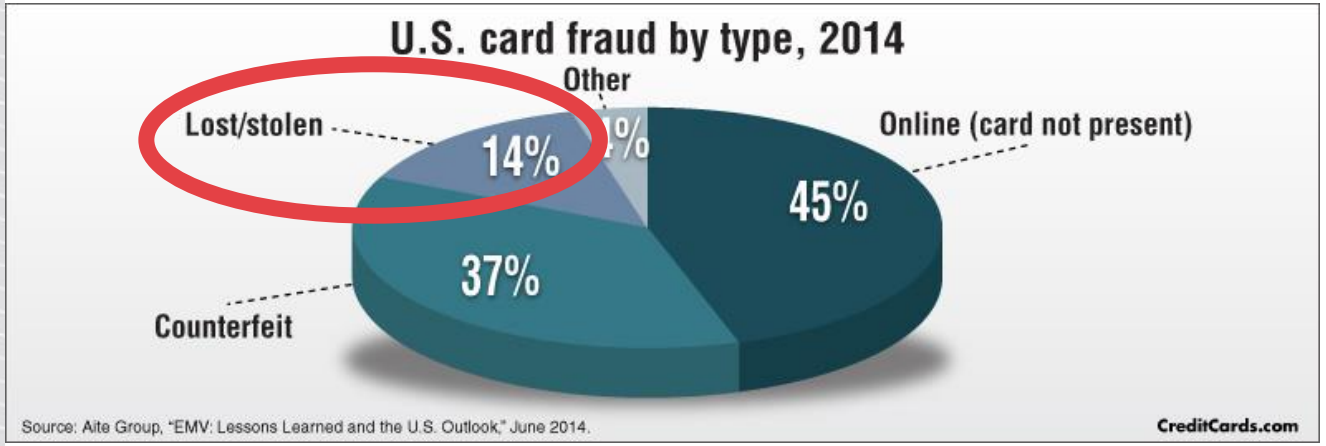
HOW TO STEAL THE MONEY?

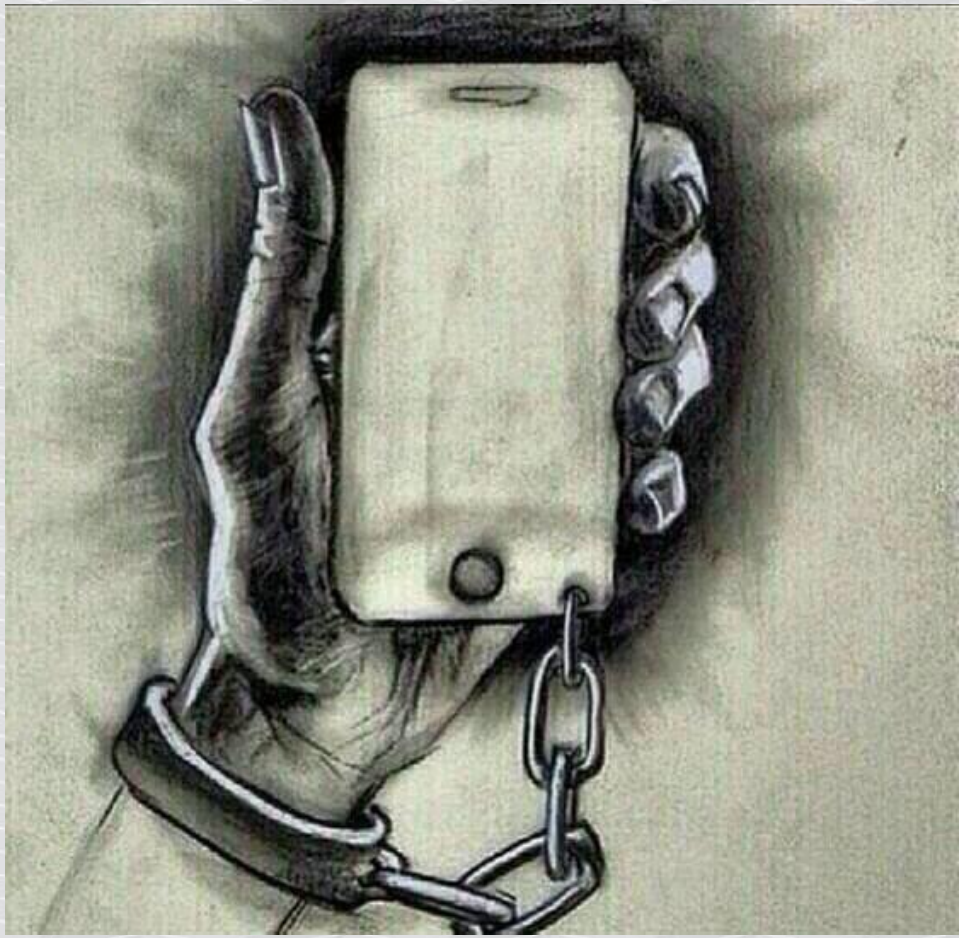
Right, so how to steal the money?

http://en.wikipedia.org/wiki/Olsen_Gang

Steal the phone?

immediate report and cancel





https://twitter.com/therealbanksy/status/842853661407678464

Steal card data via NFC?

Credit card reader?

Let's try!

The screen has to be on. In some cases unlock is required.

You won't make online payments using it.

Creating magstripe track may be possible.

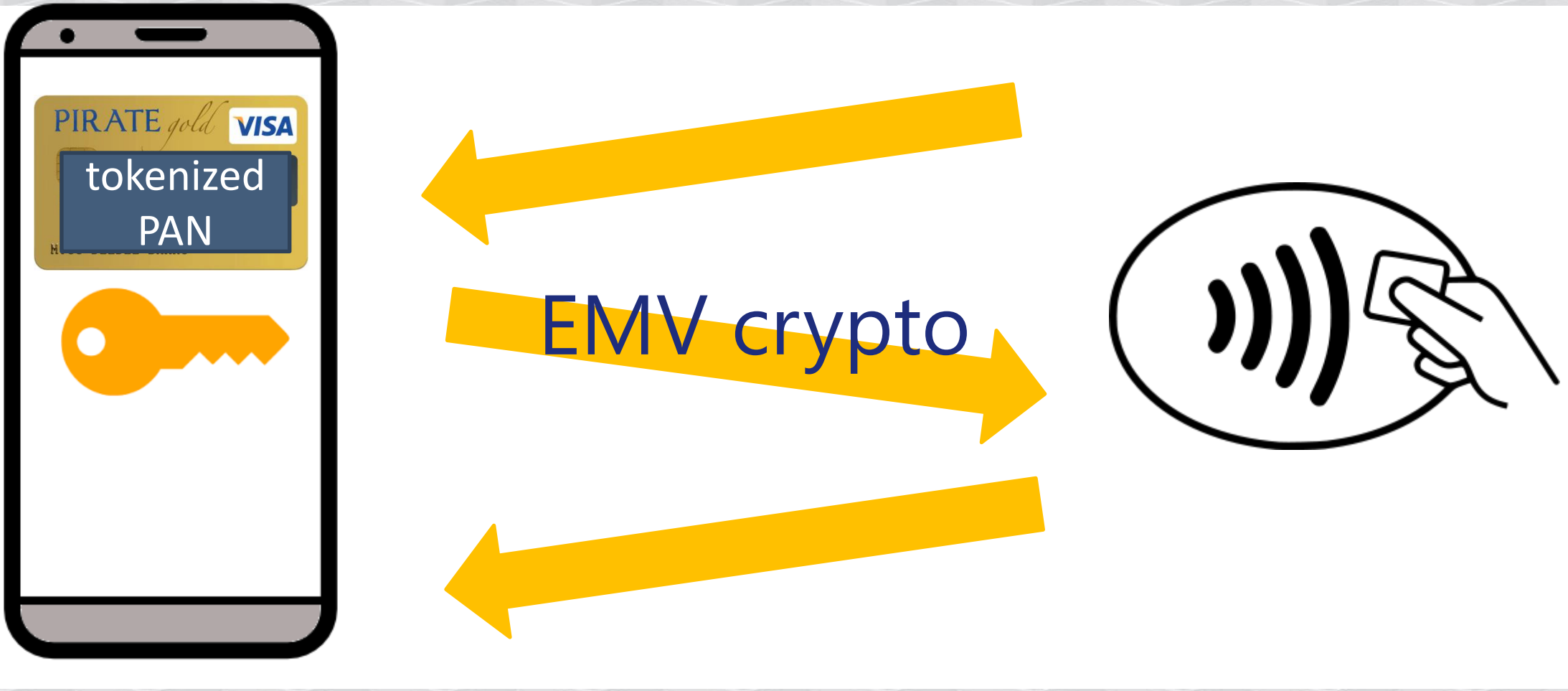
https://play.google.com/store/apps/details?id=com.github.devnied.emvnfccard

Tokenization

Random card numbers (tokens) replacing single static PAN

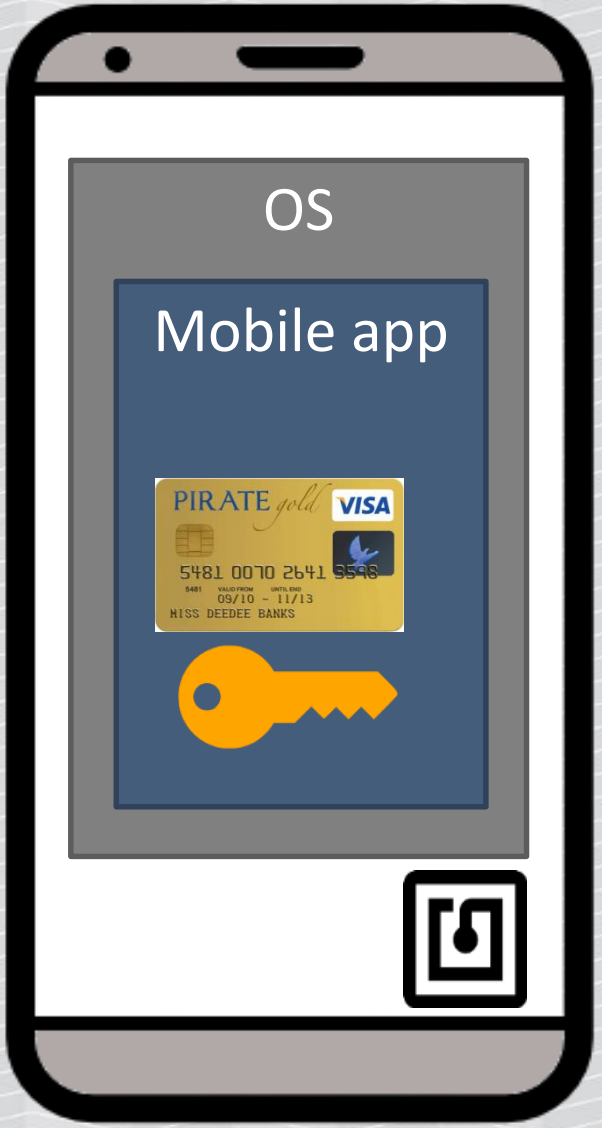
Limited „domain" use – only for contactless transactions

So, how are the EMV transactions executed?

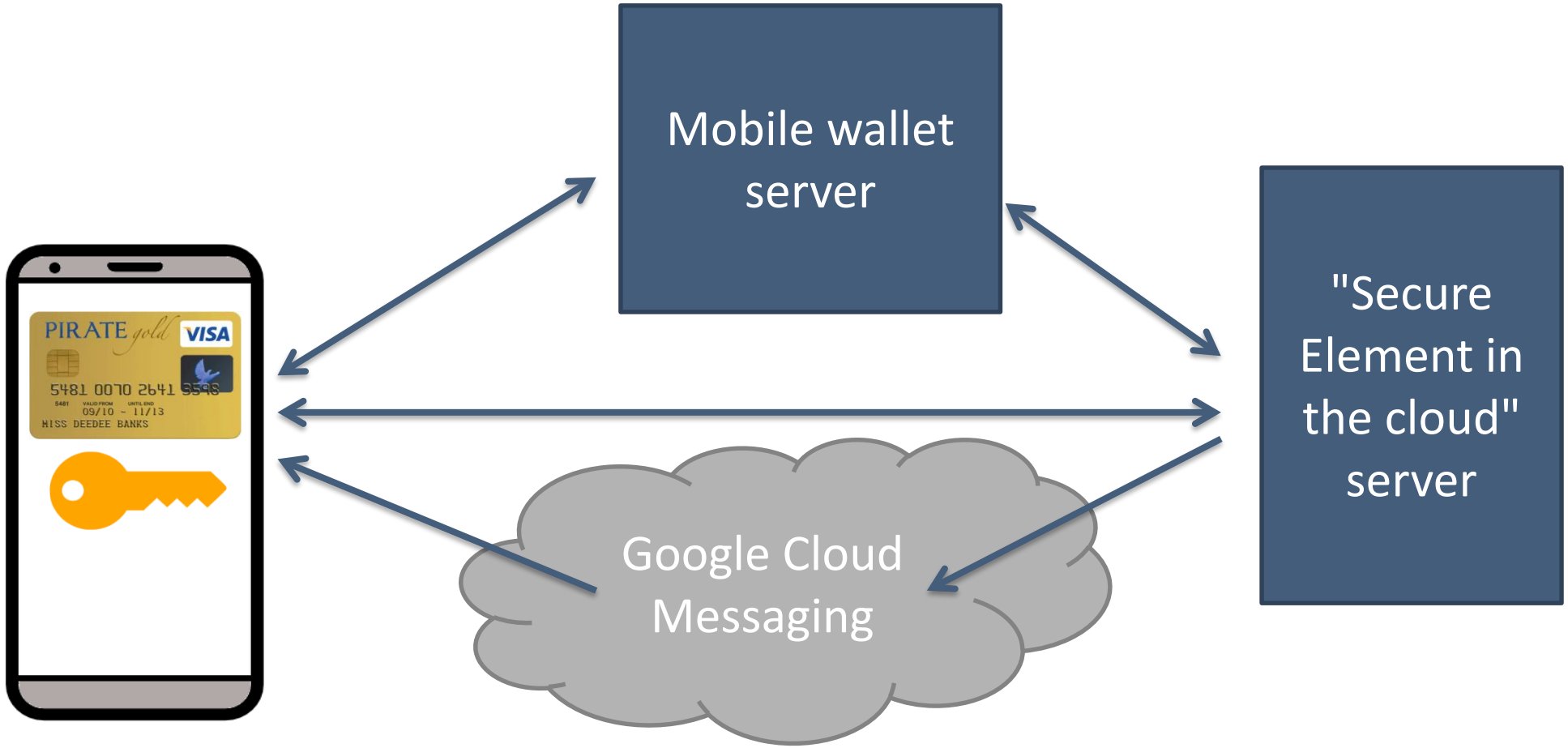


The key

How to steal it?



Intercept in transfer?



Typically

Multiple servers, push included

Certificate pinning

Second layer encryption

"encryptedData":"AAABdxcgfXea9B050gH9/a1fcJz//UpQihZrvfdHwZboTo3kNN45M0
eemFMrM1EM0BzixsDHTMFeUenl9CKMjsbJT/IvZZGceL5KmQK971NoI5wo8Kh5qgF/hazsU
2uOlyu5NxsE69QE62cffruh55DvX8f7/g=="

Flaws?

Improper pinning – accept all certs, use vulnerable lib...

Nasty bugs „deeply hidden" under the proprietary encryption layer.

- Difficult to exploit, need active access to transmission during provisioning.

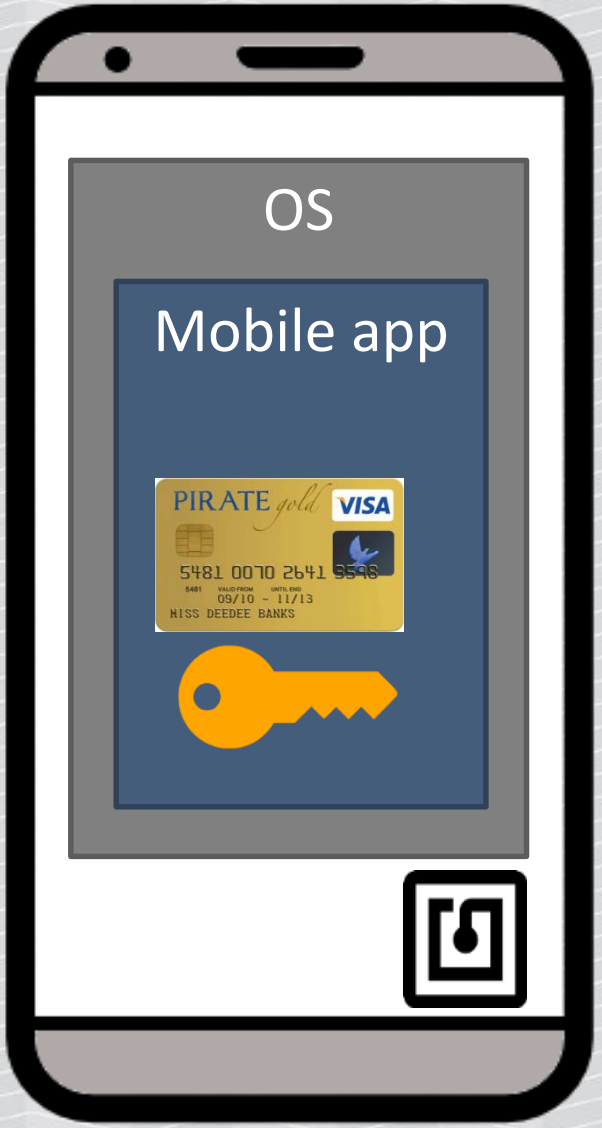
The key

How to steal it?

- Intercept in transfer?

Stored in user-space – not hardware Secure Element.

- Get it from the phone?



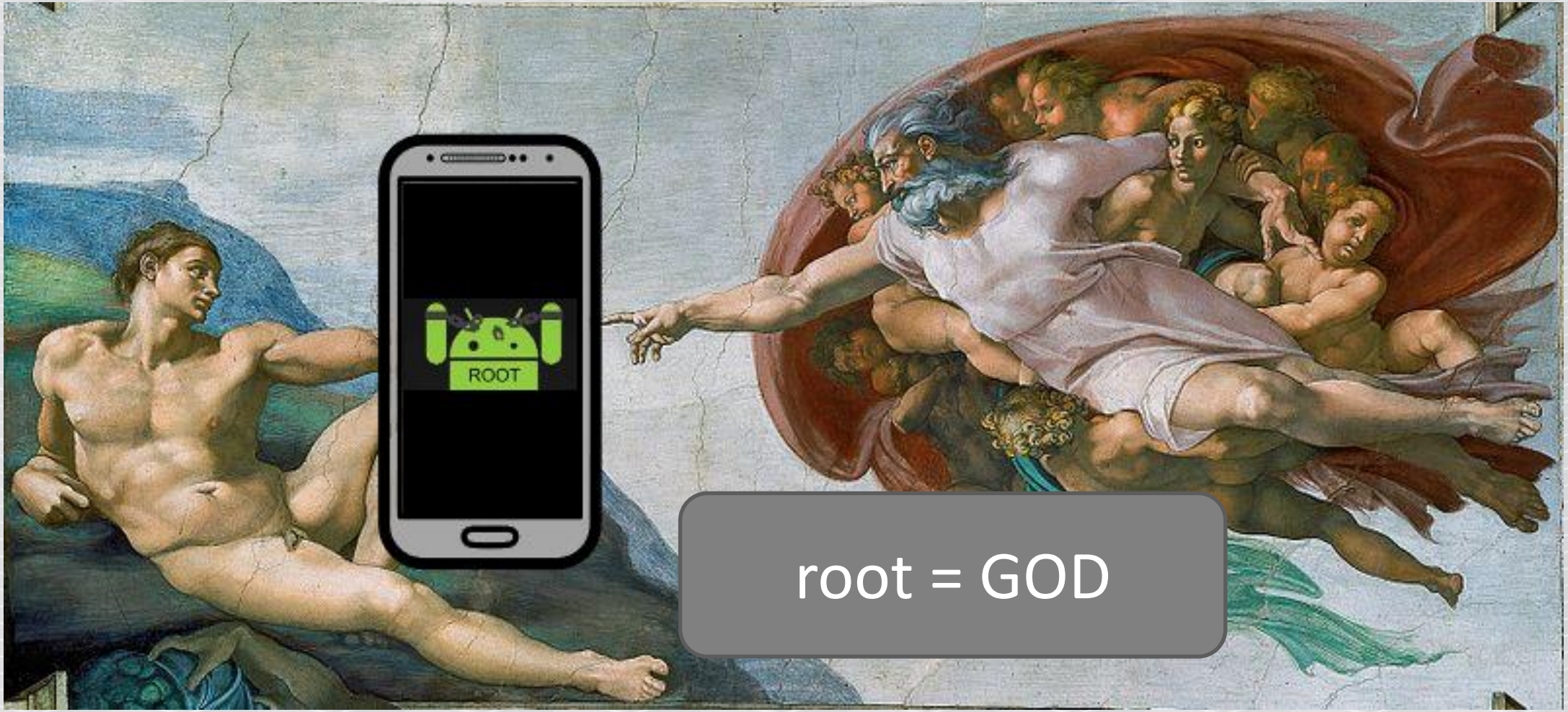
Mobile malware?

Most common:

- Overlay stealing data
- Intercept SMS
- ...
- Does not have access to card data (private folder of the app)

https://www.flickr.com/photos/freejay3/3335151608/

How to access the data?



https://en.wikipedia.org/wiki/The_Creation_of_Adam

Rooting possibilities?

Malware with root?

10 million Android phones infected by all-powerful auto-rooting apps

First detected in November, Shedun/Humn

DAN GOODIN - 7/7/2016, 7:50 PM

'GODLESS' Mobile Malware Uses Multiple Exploits to Root Devices

ANDROID MALWARE SECURITY

Tordow Banking Trojan – A Grave Threat for Android Users

Those downloading Android apps from third-party stores are the one affected by this trojan.

by **Waqas**
4 months ago

Email
@hackread

ores, including Google Play, and has affected over
0 devices worldwide.

ANDROID BANKING TROJAN FIRST TO GAIN ROOT PRIVILEGES

Sources: Ars Technica, Threatpost, Trend Micro, Hackread

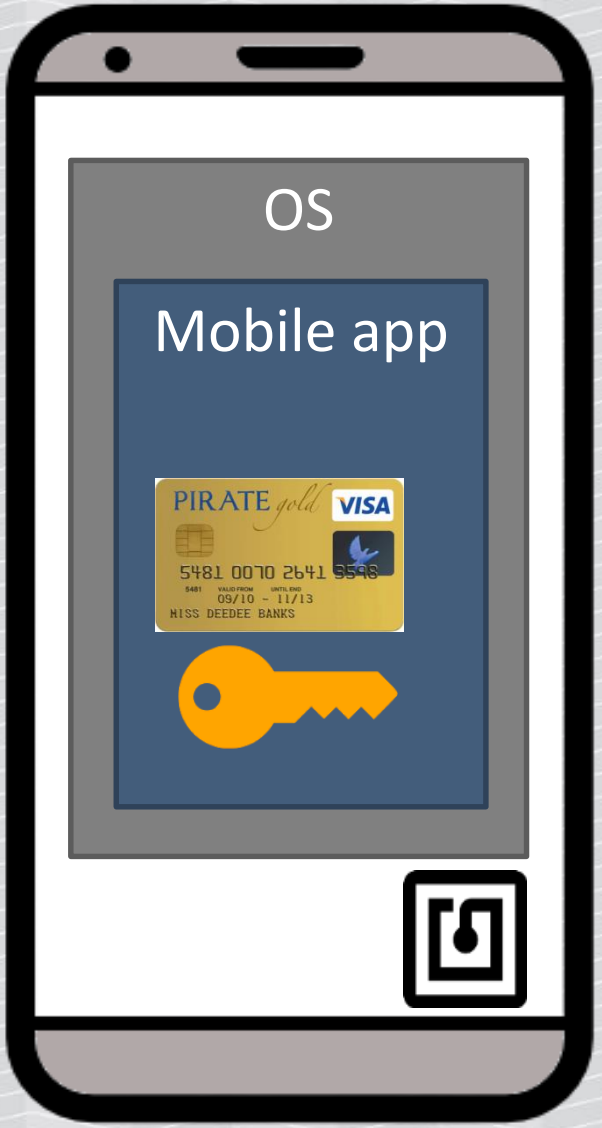
Key stored on the device

Stored in user-space – not hardware Secure Element.

How to steal it?

- Intercept in transfer?
- Get it from the phone? Root malware!

But the key is encrypted... How to decrypt?



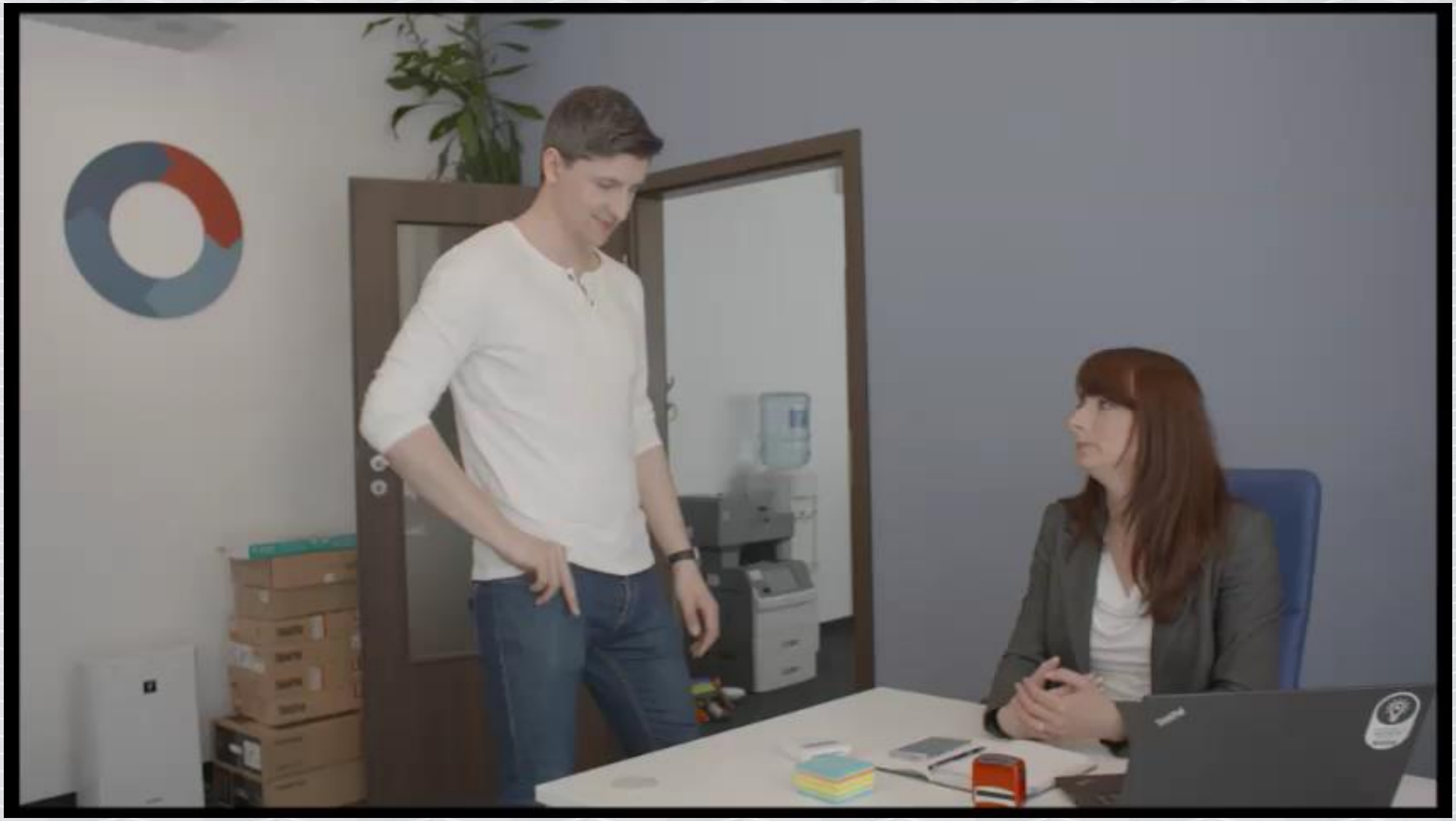
Decompile the app binary?

```
public '()
{
  try
  {
    _ = KeyStore.getInstance('(' >>> 1, -'[93], '[50]));
    .load(null);
    if ((' == null) || (' .isEmpty()))
    {
      StringBuilder localStringBuilder = new StringBuilder().append(ContextHelper.getApplic
      int i = '[92];
      int j = '[54];
      ' = '(i, j, j | 0x45);
    }
    return;
  }
  catch (Exception localException)
  {
    for (;;)
    {
      .. ('', '('[54], -'[96], -'[100]) + localException.getMessage());
      _ = null;
    }
  }
}

private static String '(int paramInt1, int paramInt2, int paramInt3)
{
  byte[] arrayOfByte2 = ';
  int i = 44 - paramInt1;
  paramInt1 = paramInt2 + 4;
  byte[] arrayOfByte1 = new byte[i];
  int k = i - 1;
  int j;
  if (arrayOfByte2 == null)
  {
    paramInt2 = 0;
    j = paramInt1;
    i = k;
    paramInt3 = paramInt1;
    paramInt1 = j + 1;
    i = i - paramInt3 - 4;
    paramInt3 = paramInt2;
    paramInt2 = i;
  }
```

```
  private static final byte[] ' = { 73, -36, -27, -98, 9, -1, 1, -17, 12, -20, 19, -11, -3, 6, -16, 6, -44, 72, 9, -8
 83, -6, -80, 67, 0, 6, -8, 14, -16, -70, 83, -13, -4, -70, 69, 2, 2, -8, -44, 67, 10, -37, 25, 19, -7, 0, -6, 2, -14
-12, 5, -2, -79, 32, 44, -11, 13, -4, -7, -6, -69, 21, -23, 65, 18, -2, -85, 84, -6, -10, 10, -18, 2, 0, -2, -69, -4,
8, 18, -16, 9, 0, -2, -84, -14, 9, -19, 29, -13, -16, 6, 19, -13, -16, -4, 16, 12, -31, 10, -4, -9, 17, -35, 0, 0, 0,
 -14, 14, 0, -4, -5, -10, -2, -69, 77, 6, -10, -1, -77, 67, -4, 16, -83, 64, 12, -11, -69, 82, -16, -4, 16, -1, 11, -
1, -4, 4, -8, -72, 69, 11, -4, -3, -78, 68, 8, -12, 14, 6, -10, 3, -49, 28, 18, -20, -16, -4, 16, -69, 53, -4, 18, 38
-16, 6, -29, 57, 9, -1, 1, -17, 12, -20, 19, -11, -3, 6, -16, 3, 1, 12, 1, -71, 53, -4, 18, 15, 14, 2, -20, -67, 64,
 -70, 37, 26, 7, 2, -8, -2, -55, -14, 9, -19, 29, -13, -16, 6, 19, -28, -4, 16, 12, -31, 10, -4, -9, 17, -35, 38, 18,
6, -29, 52, 14, -9, -62, 53, -4, 18, -2, 13, -18, 1, 14, 1, -10, 2, -63, 53, -3, -16, 6, -44, 72, 9, -84, 72, 4, -79,
 -70, 44, 33, -12, 0, -56, 25, 19, -7, 0, -6, 2, -14, -70, 50, 32, -6, 2, -14, -2, -69, 78, -2, -79, 83, -13, -4, -70
, -14, -70, 82, -5, -80, 82, 0, -6, 2, -10, 4, -8, -72, 69, 2, 2, -8, -70, 86, -15, 10, -13, -73, 74, -7, 19, -7, 0,
4, 6, -10, 3, -12, 5, -2, -4, 16, -69, 53, -4, 18, 38, 18, 9, -19, -31, -2, 13, 2, -16, 6, -29, 57, 9, -1, 1, -17, 12
, -4, 18, 38, 18, 9, -19, -31, -2, 13, 2, -16, 6, -29, 52, 0, 16, -14, -7, 1, -7, 9, -6, -6, 6, 2, 2, -8, -56, 53, -4
 0, -6, 2, -14, -34, 32, -3, 14, 6, -10, 3, -12, 5, -2, -79, 32, 44, -11, 13, -4, -7, -6, -69, 21, 38, 18, 9, -19, -3
 6, 3, 0, -22, 9, -6, -6, 6, 2, 2, -8, -56, 53, -4, 18, 67, 10, -37, 25, 19, -7, 0, -6, 2, -14, -34, 32, -3, 14, 6, -
 4, -53, 32, 44, -11, 13, -4, -7, -6, -69, 21, -23, 82, -5, -80, 84, -3, -11, 4, -8, -72, 69, 2, 2, -8, -70, 82, 5, -
-19, -31, -2, 13, 2, -16, 6, -29, 57, 9, -1, 1, -17, 12, -20, 5, 8, -12, -54, 53, -4, 18, 1, -6, -12, 12, 12, -11, -1
13, -1, 11, -27, 18, -16, 9, 0, -2, 38, 18, 9, -19, -31, -2, 13, 2, -16, 6, -29, 57, 9, -1, 1, -17, 12, -20, 5, 8, -1
4, 18, -16, 6, -44, 72, 9, -84, 84, -8, -14, 0, 9, -8, -70, 83, -6, -80, 67, 0, 6, -8, 14, -16, -70, 83, -13, -4, -70
-15, -44 };
  private static int ' = 99;

  private static String '(int paramInt1, int paramInt2, int paramInt3)
  {
    byte[] arrayOfByte2 = ';
    paramInt3 += 2;
    paramInt1 += 4;
    byte[] arrayOfByte1 = new byte[paramInt3];
    int j = paramInt3 - 1;
    int k;
    int i;
    if (arrayOfByte2 == null)
    {
      k = -1;
      paramInt2 = paramInt1;
      i = paramInt1;
      paramInt3 = j;
      paramInt1 = k;
      paramInt3 = paramInt3 + i + 1;
      paramInt2 += 1;
      i = paramInt1;
      paramInt1 = paramInt3;
    }
  for (;;)
```

https://twitter.com/FailsWork/status/827135802882813953/

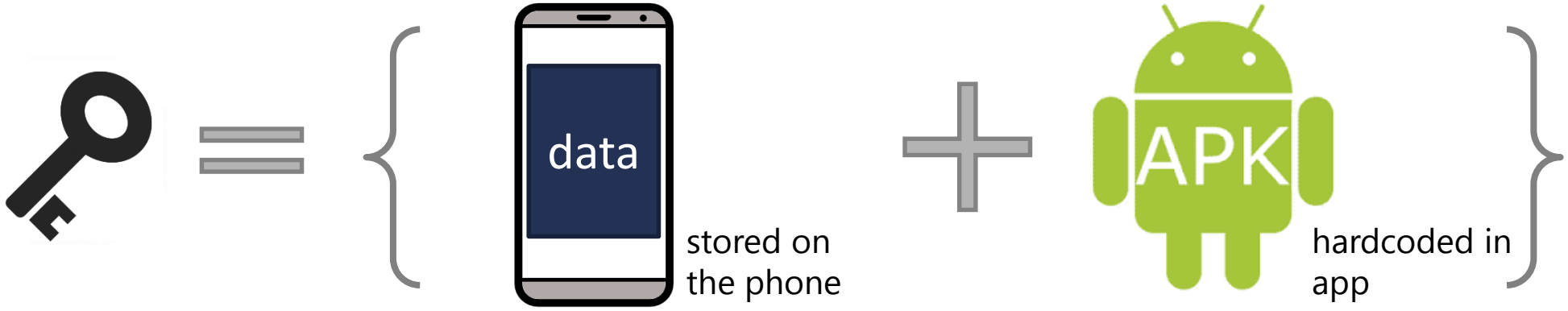
How does it work?



Encryption

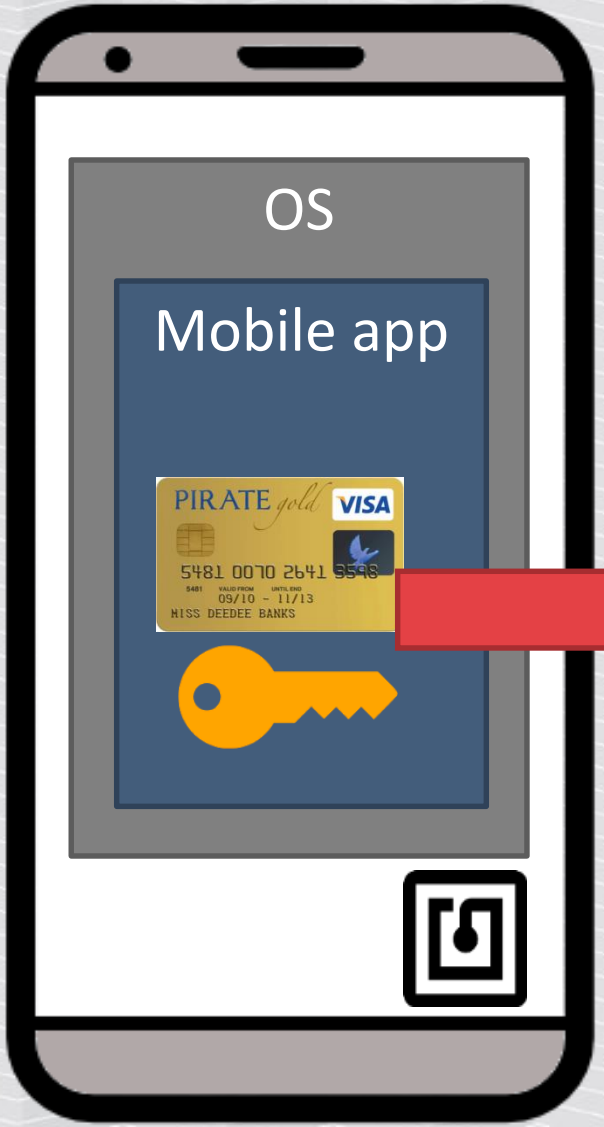
Does not require user interaction (no PIN/pass).

Works also when phone is offline.



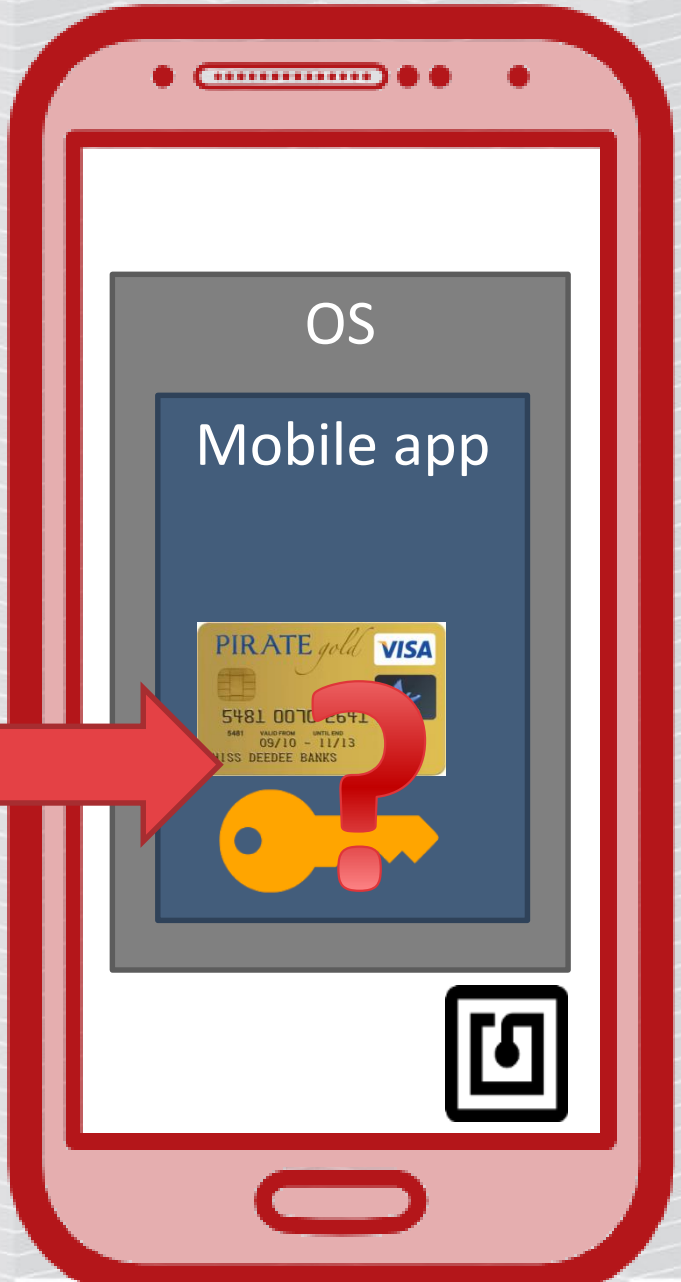
So, what can we do to clone the card?

Install the same app, copy data?

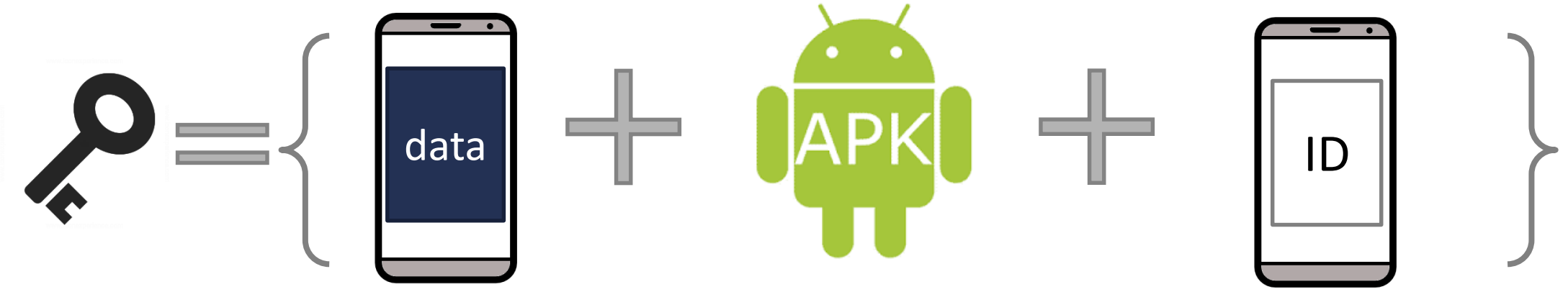


No, it does not work ☹

The key is tied to specific device



Encryption again

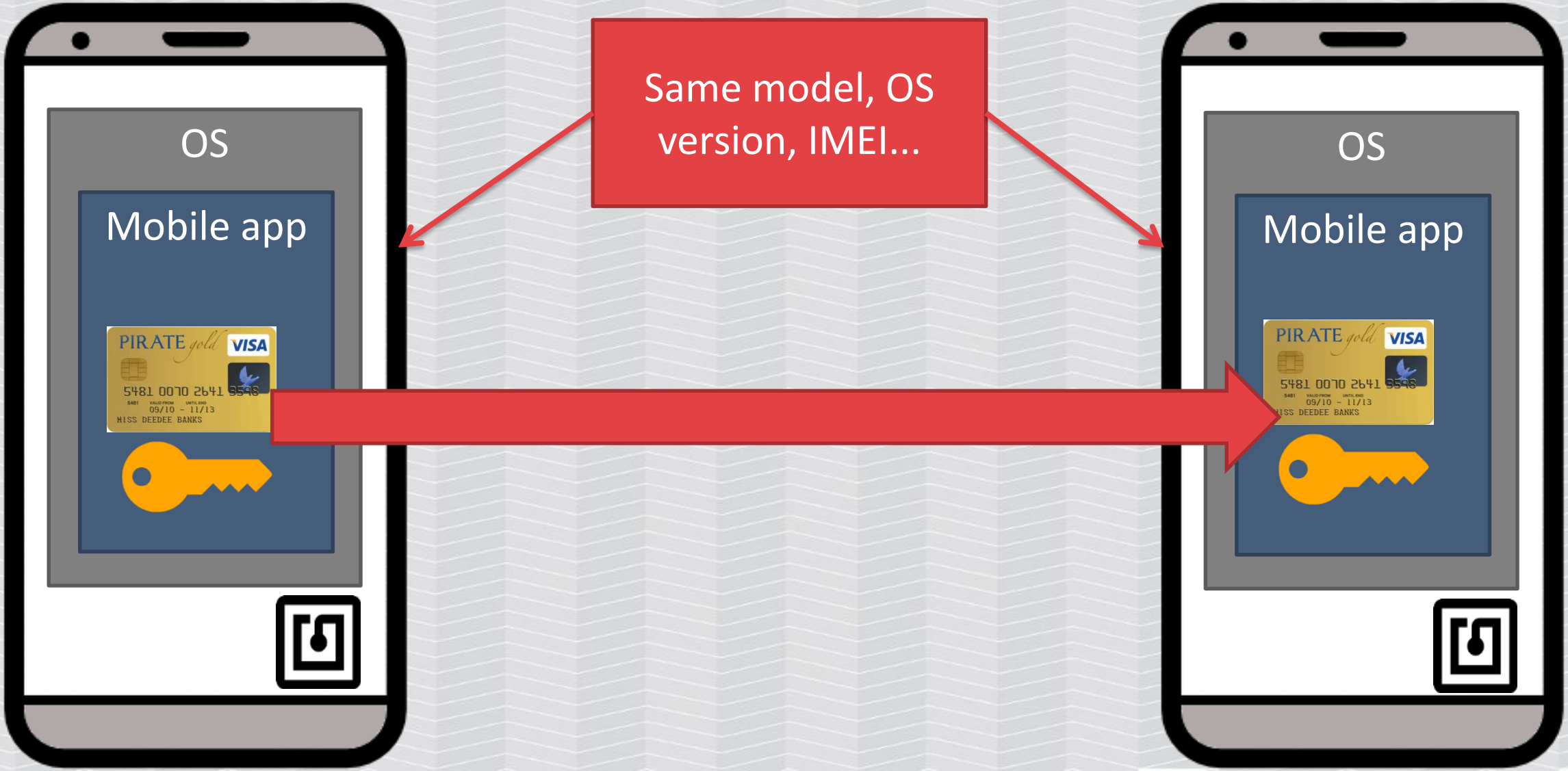


stored on the phone

hardcoded in app

Device-specific characteristic

How about exactly same hardware device?

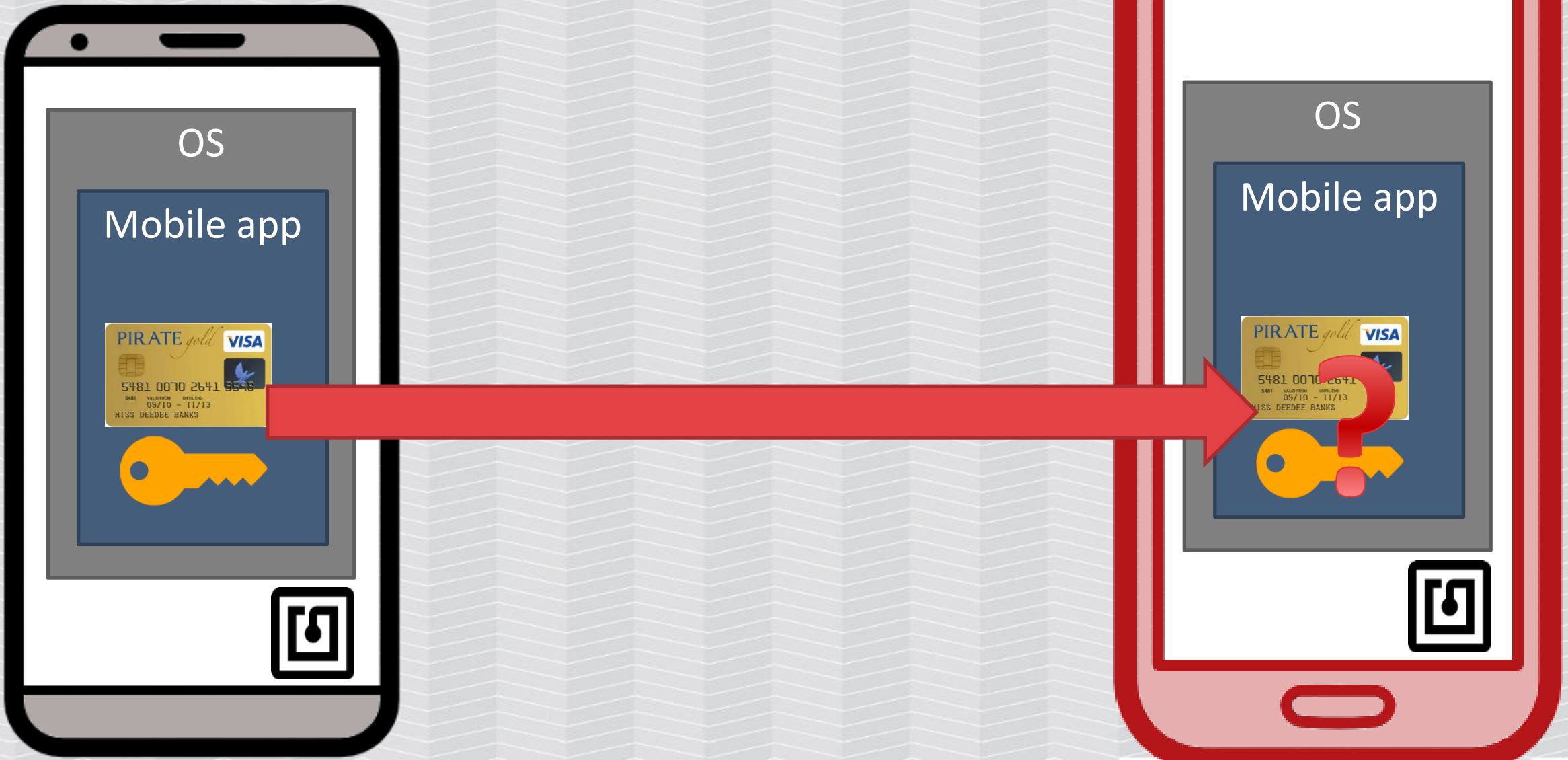


Works!

In most cases you need to copy also other user data (not just the payment app)

Not really practical attack on a mass scale...

How to make it to a different device?



Device characteristics?

AndroidID

DeviceID (IMEI)

Phone number

MAC address

Manufacturer, Model

Serial

OS version, build

Device characteristics?

AndroidID

DeviceID (IMEI)

Phone number

MAC address

Manufacturer, Model

Serial

OS version, build

May change in time

Device characteristics?

AndroidID

DeviceID (IMEI)

Phone number — Mostly inaccessible

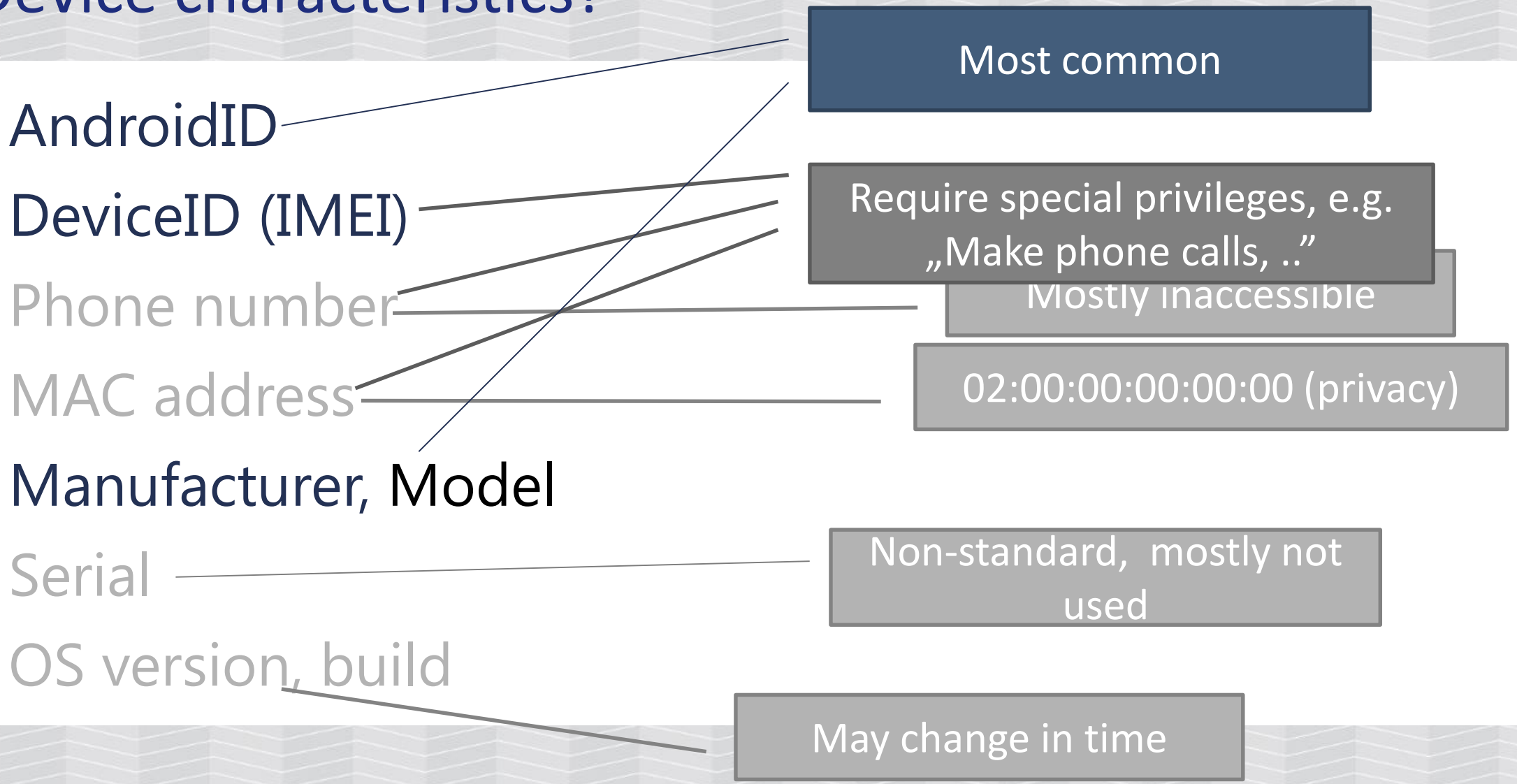
MAC address — 02:00:00:00:00:00 (privacy)

Manufacturer, Model

Serial

OS version, build — May change in time

Device characteristics?

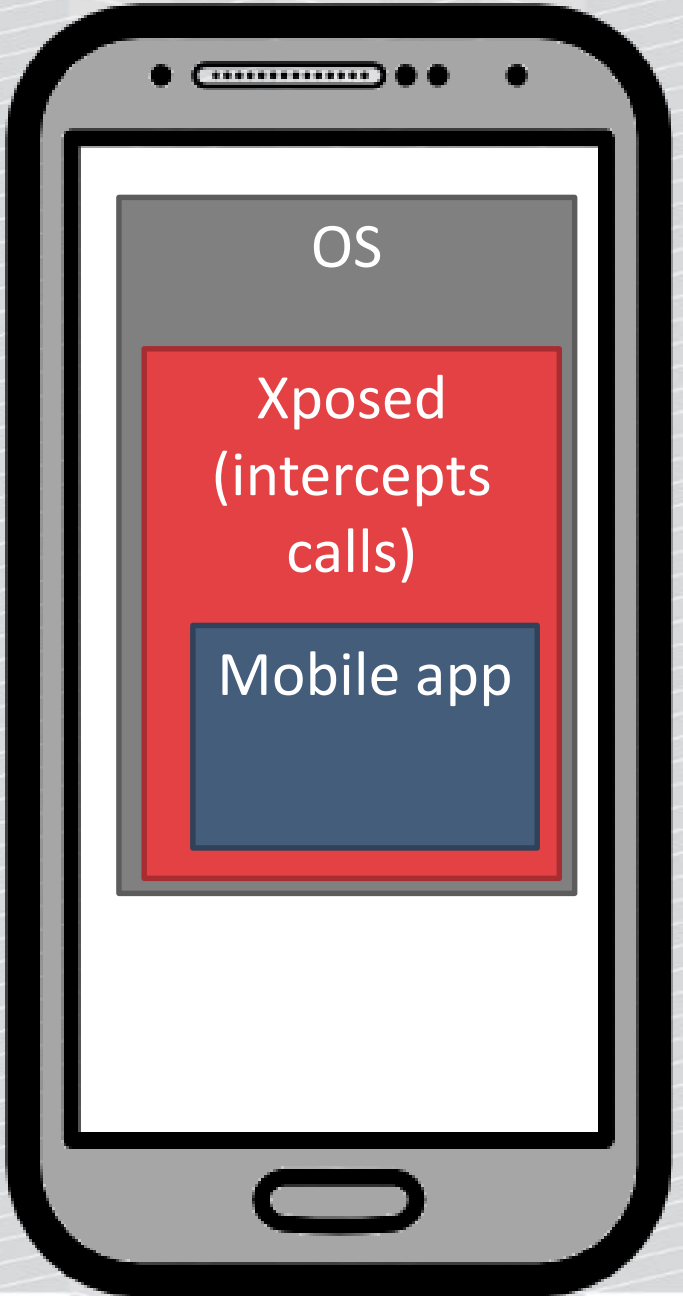


Xposed Framework

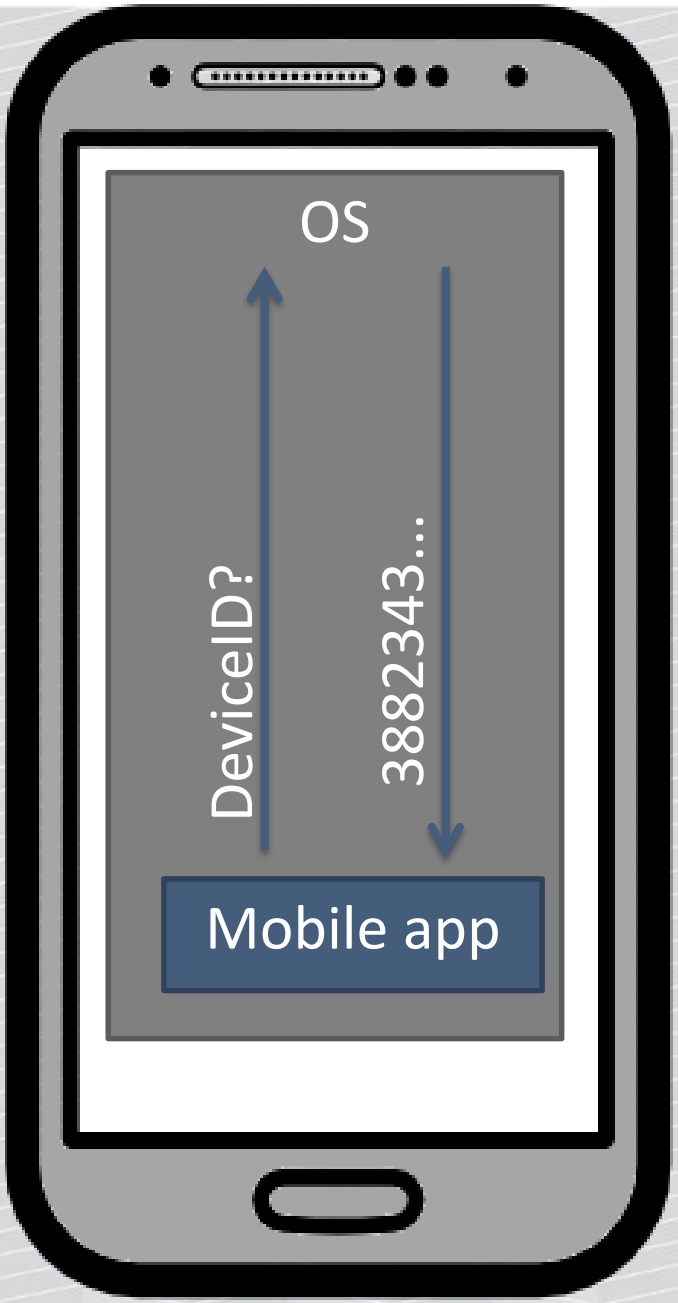
Change behavior of system and apps

Hooks into system calls.

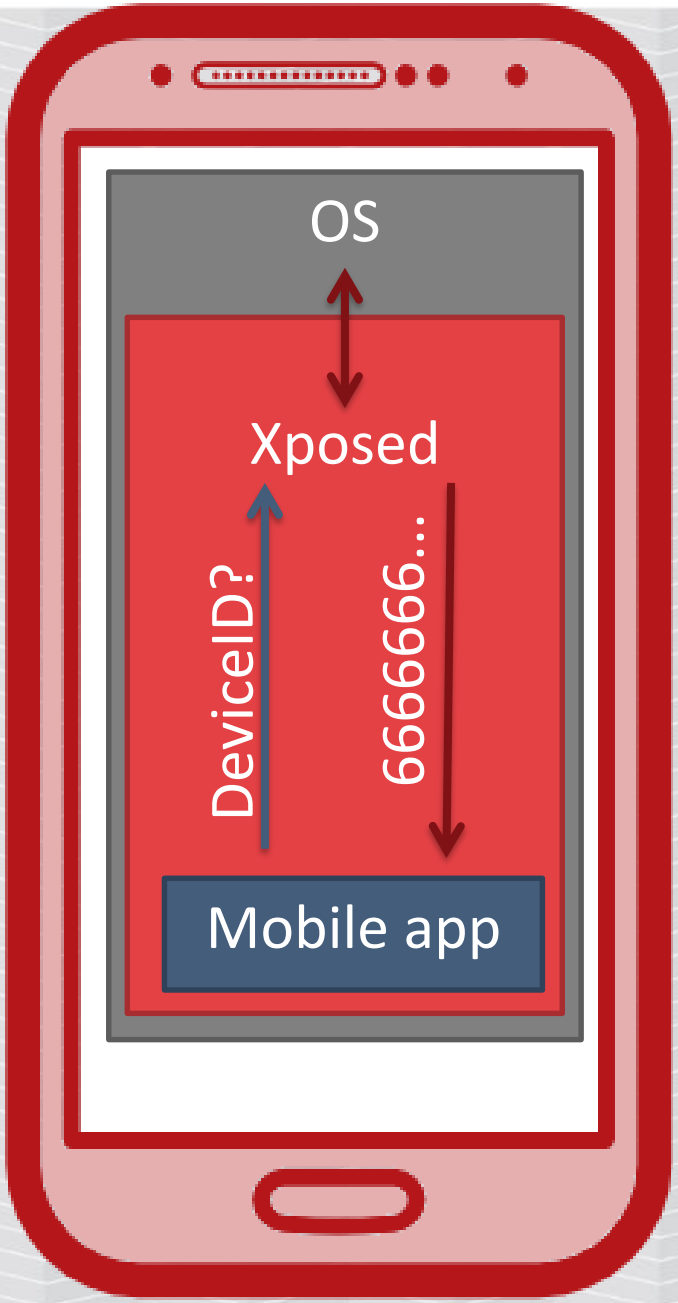
Requires root.



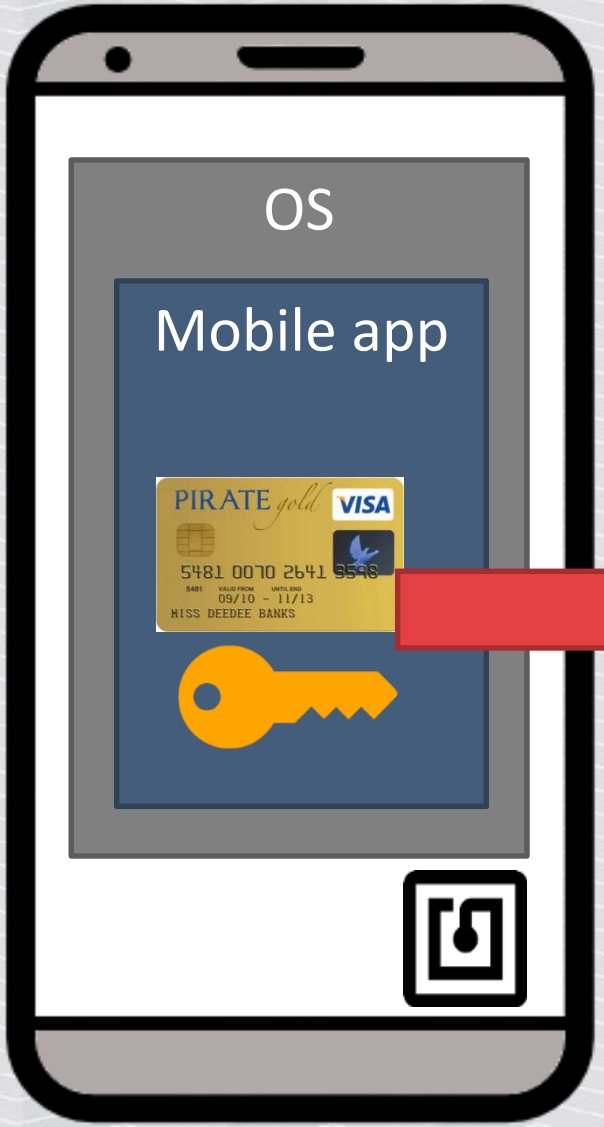
Standard device



Xposed - framework + module changing ID



Xposed – helps to imitate original device



The key is tied to specific device

ROOT DETECTION

Root detection?

Having ultimate control you can always hide from detectors.

Detection checks for popular rooting ways

http://simpsons.wikia.com/wiki/The_Itchy_%26_Scratchy_Show

SafetyNet root detection

```
private static final String[] a = {
    "/system/bin/su",
    "/system/xbin/su",
    "/system/bin/.su",
    "/system/xbin/.su" };
```

https://www.flickr.com/photos/136682034@N03/26086288495/

Cloning script

```
tar

chown

restorecon
```

https://www.flickr.com/photos/alijava/27634909445

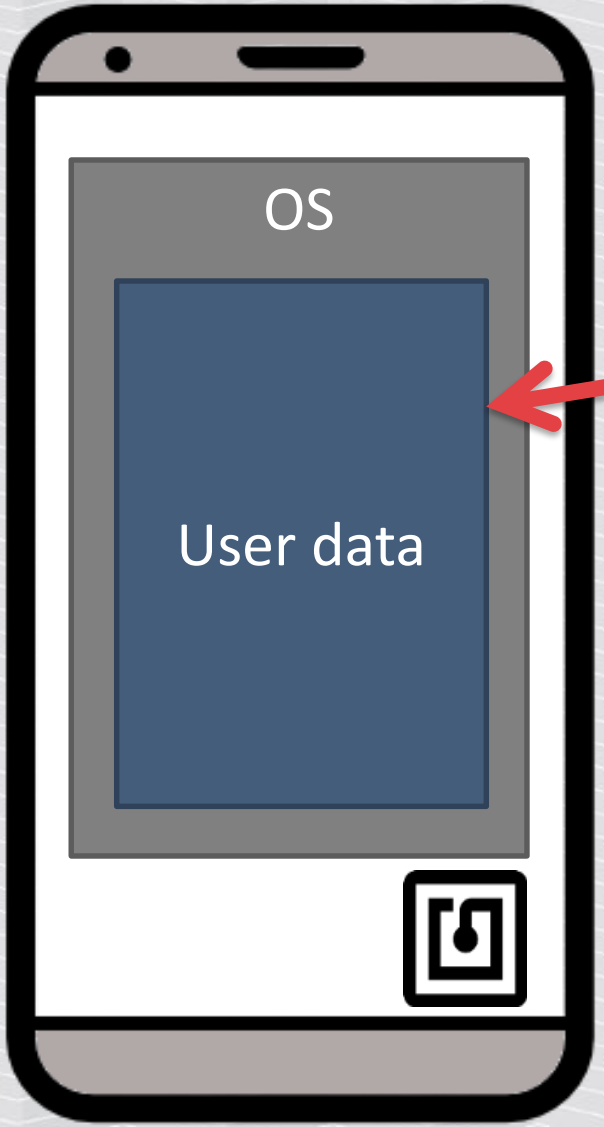
backup.sh

```
<redacted>
```

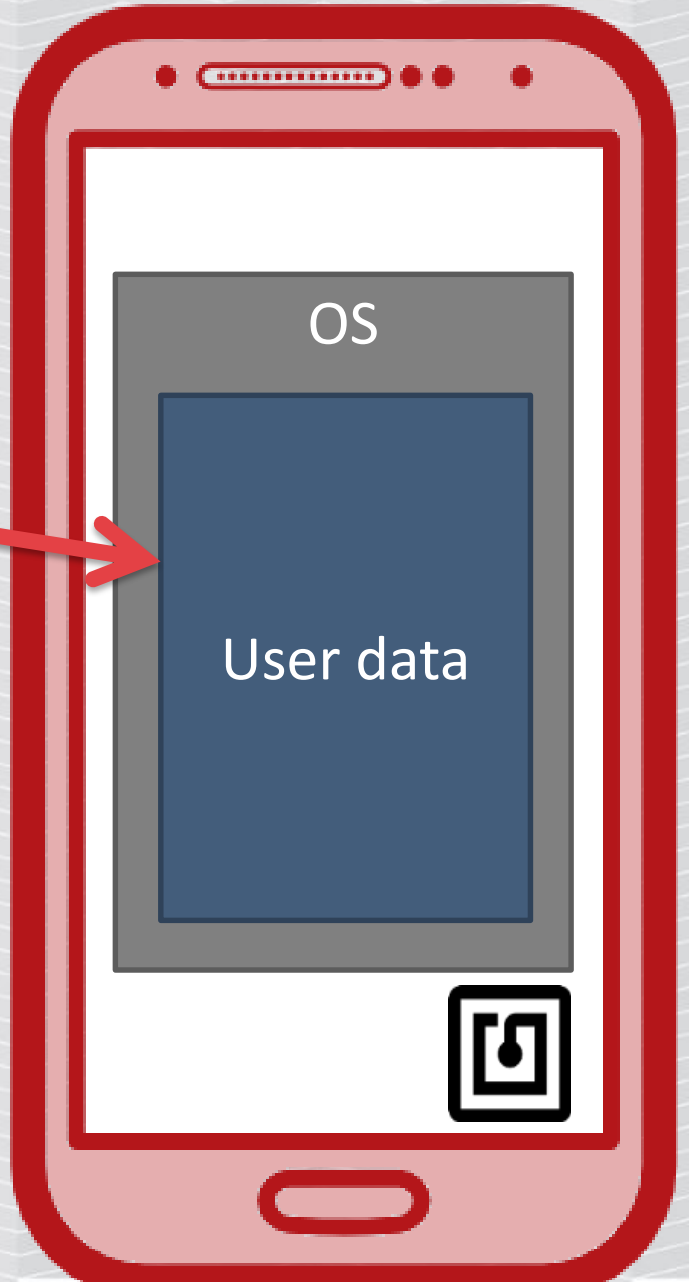
restore.sh

```
<redacted>
```

The secret ingredient



Exactly same OS and Google Services version



Real risk?

PoC was on a single, small amount transaction from the same network and physical location.

Google definitely has some FDS/behavioral analysis systems.

https://www.flickr.com/photos/tambako/3655641638/

KEYS REPLENISHMENT

Finally, we can use the card on other device!

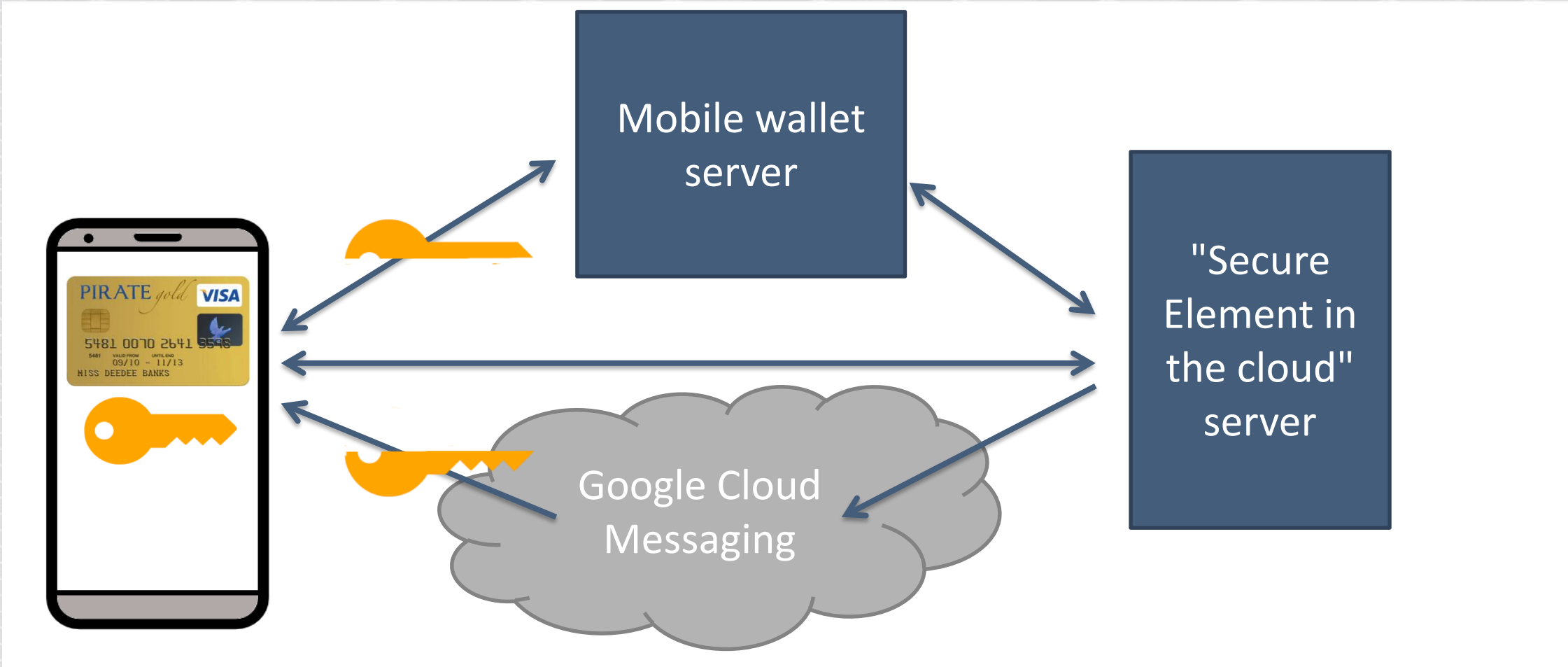
… but the keys are limited-use.

Only a few transactions < 25 EUR each?

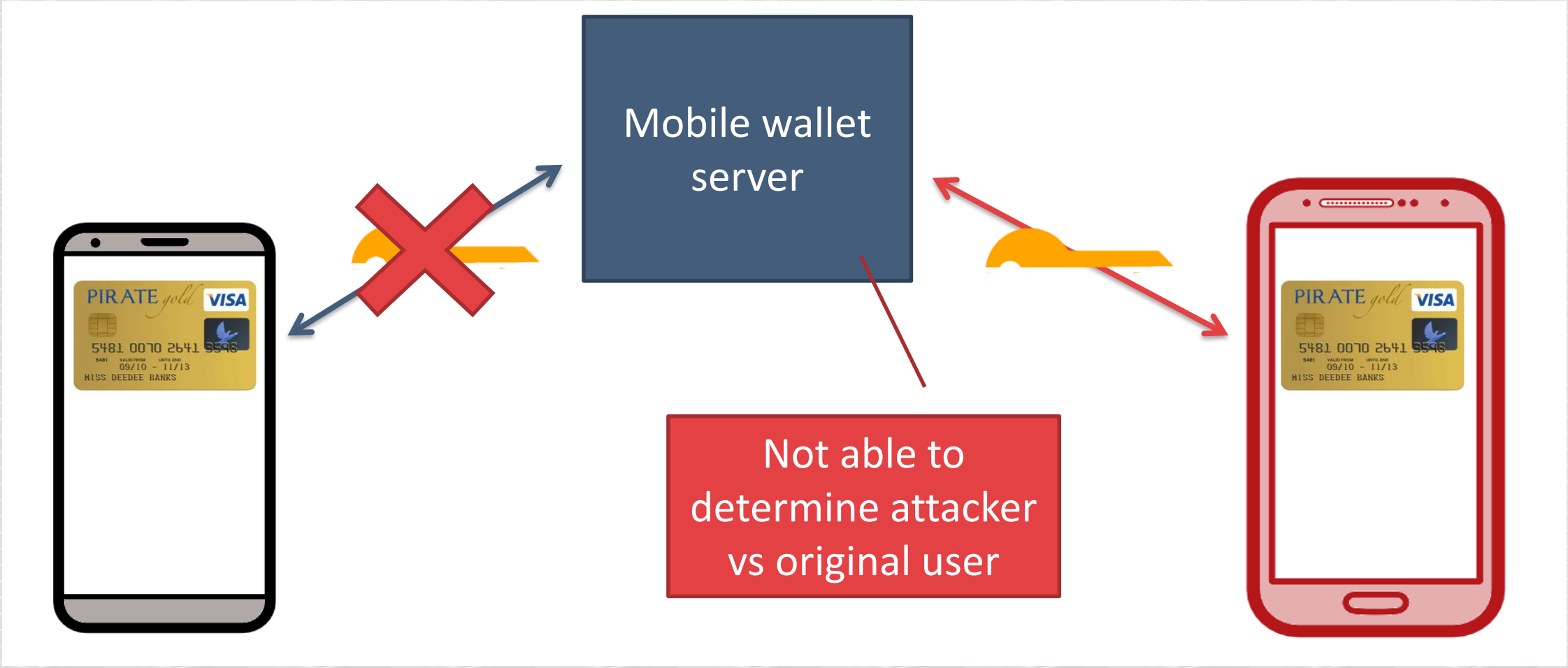
Then the keys have to be replenished.

So, how does it work?

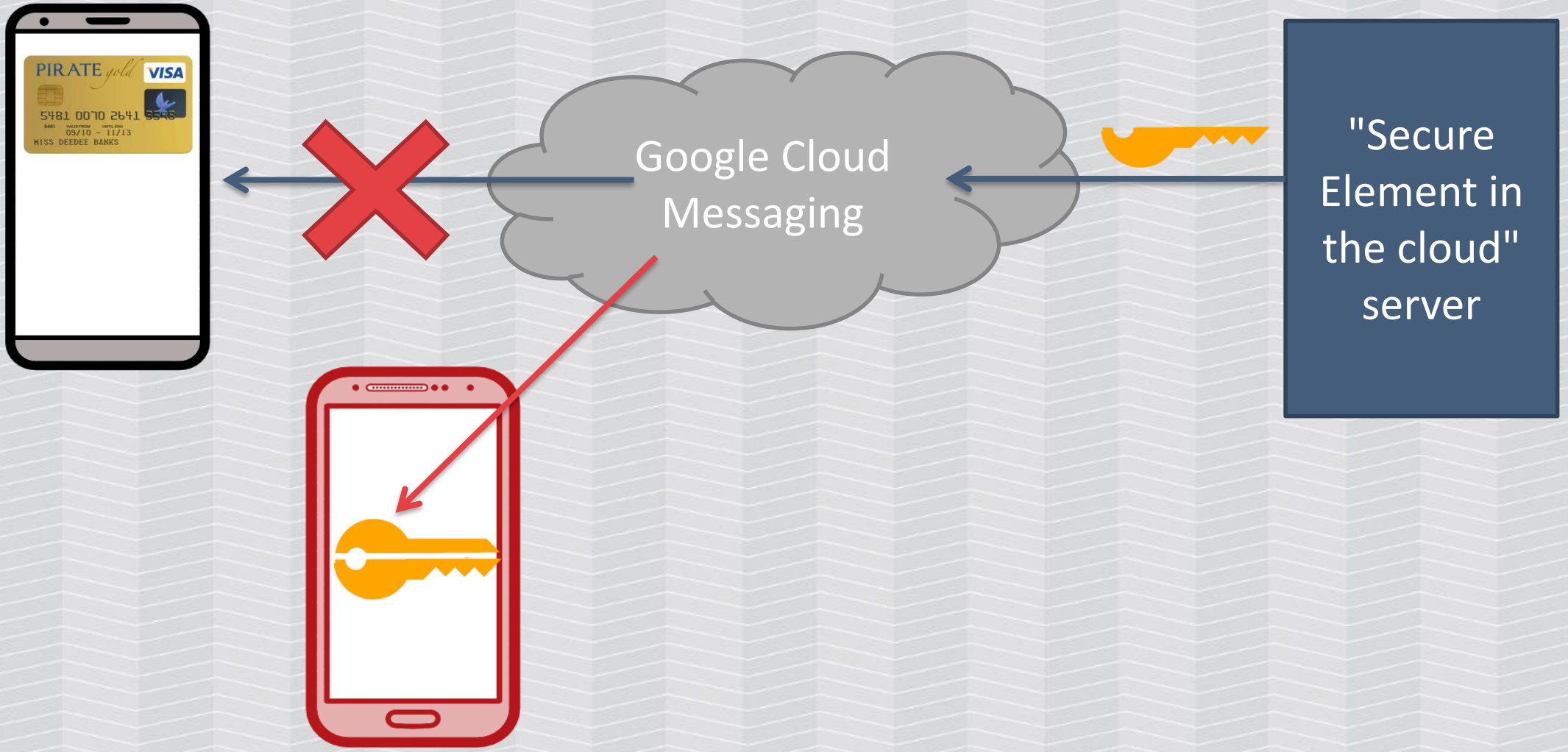
Keys replenish – most common: GCM combined



This part we already have



How to hijack GCM push?



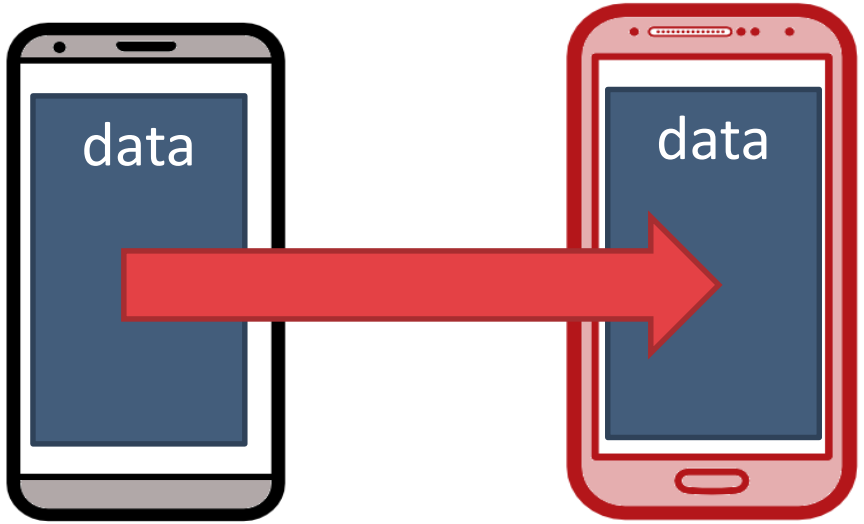
Hijack GCM push

Copy relevant user data (/data/system/users, ...)

Both devices have same AndroidID, keys, subscriptions

Test push received by:

- sometimes both
- only one (mostly „cloned")
- I can block original user



Having root access to victim's phone

- Make few low-value transactions from another device
- Make multiple transactions (renew limited-use keys)
- But... there are usually limits on number of transactions

FLOOR LIMIT

The „floor limit”

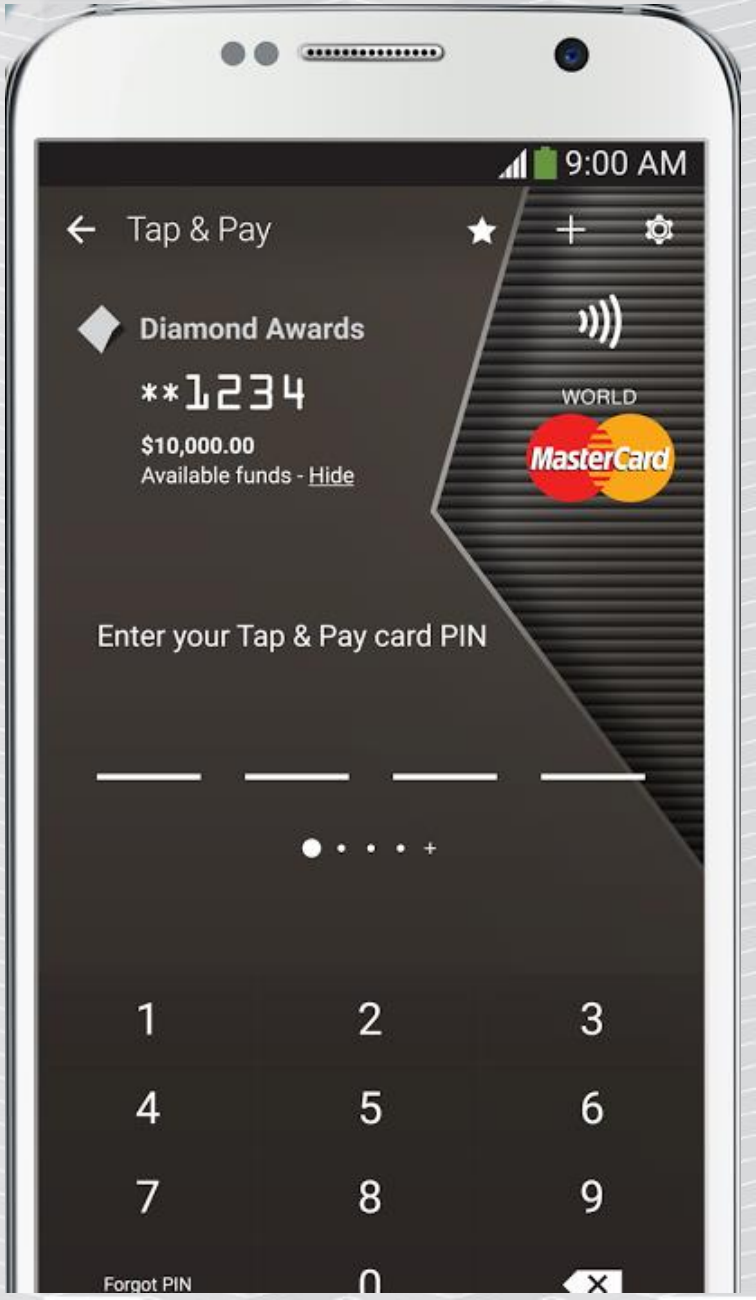
Transactions > 25 EUR need authorization

Several options:

- Enter card PIN in terminal

BUT - how do you set up the PIN?

Mobile malware -> can sniff the PIN / trick user into entering it

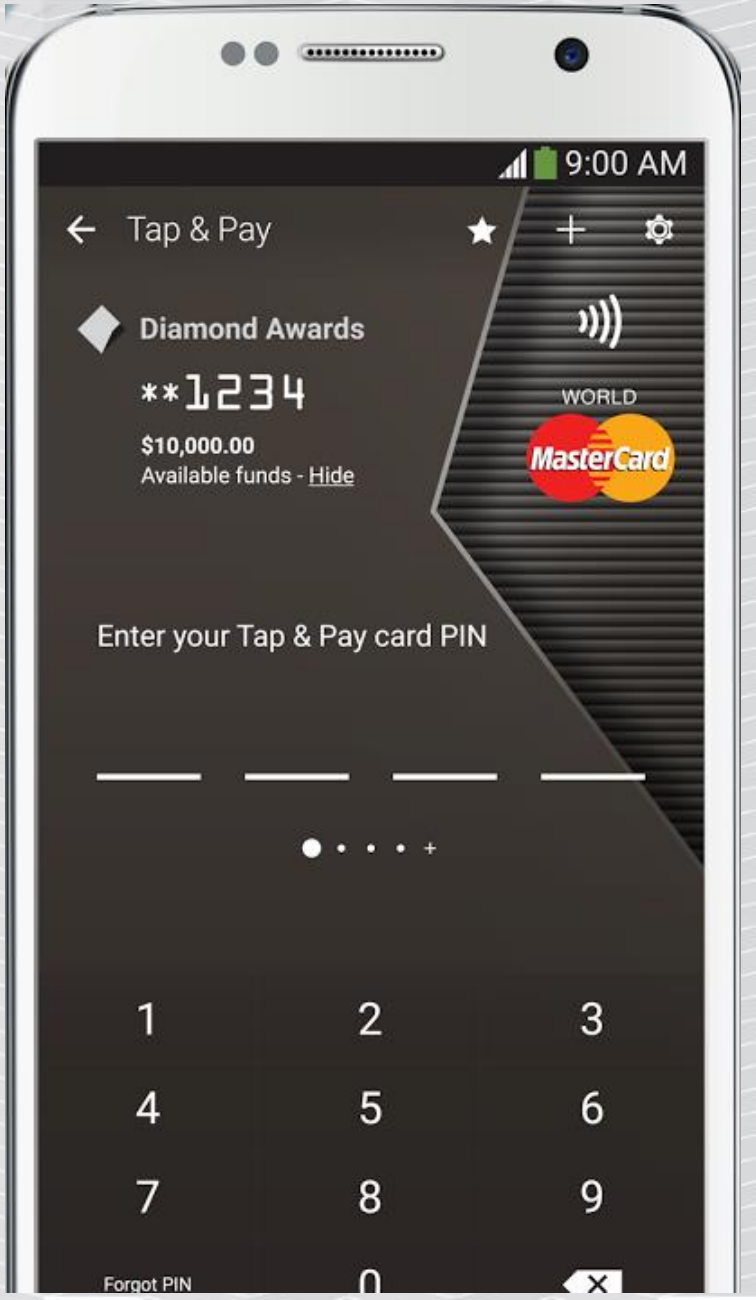


The „floor limit”

Transactions > 25 EUR need authorization

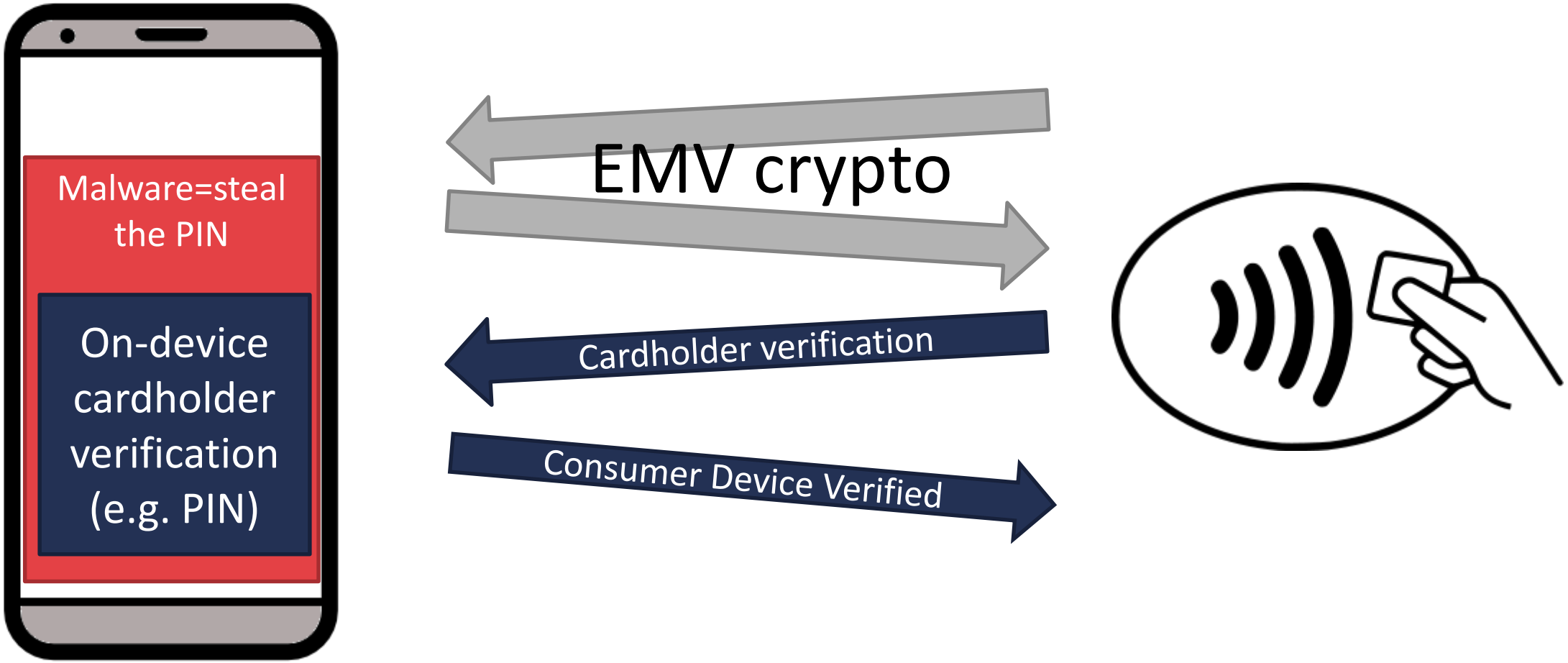
Several options:

- Enter card PIN in terminal
- CDCVM



CDCVM

Consumer Device Cardholder Verification Method



Having root access to victim's phone

- Make few low-value transactions from another device
- Make multiple transactions (renew limited-use keys)
- Make transactions on higher amounts

CDCVM – not very common

< 20% apps support it

So what if application does not support CDCVM?

CDCVM API methods in HCE library?

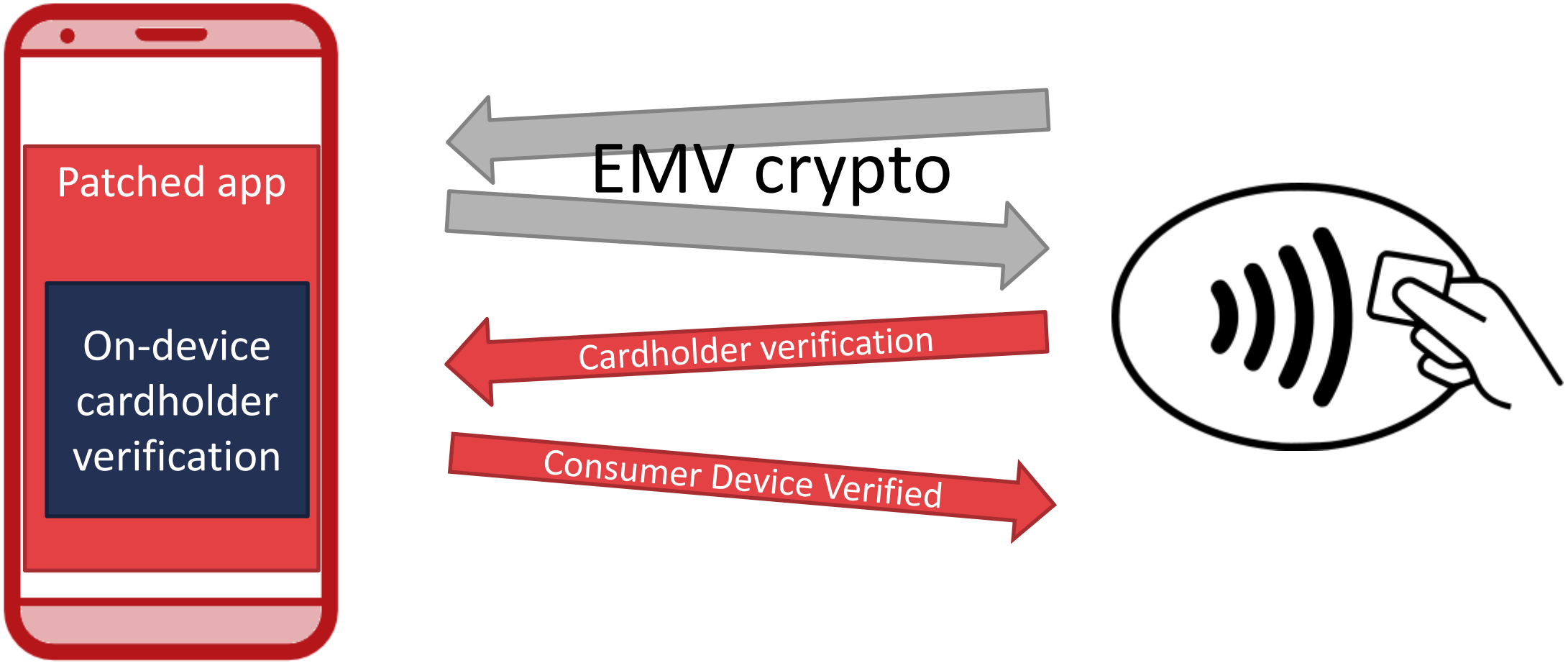
API method names (cannot be obfuscated)

```
setCvmVerificationMode(CvmMode
paramCvmMode);

setCvmVerified(boolean paramBoolean);
```

Patch the application - smali?

```
const/4 v9, 0x1
invoke-interface {v8,v9}, L<redacted>;->setCvmVerified(Z)V
new-instance v9, L<redacted>/CvmMode;
sget-object v10, L<redacted>/VerifyingEntity;->MOBILE_APP:
L<redacted>/VerifyingEntity;
sget-object v11, L<redacted>/VerifyingType;->PASSCODE:
L<redacted>/VerifyingType;
invoke-direct {v9, v10, v11}, L<redacted>/CvmMode;-><init>
(L<redacted>/VerifyingEntity;L<redacted>/VerifyingType;)V
invoke-interface {v8, v9}, L<redacted>;->setCvmVerificationMode
(L<redacted>/CvmMode;)V
```

CDCVM in app which does not support it ;)



Results are inconsistent...

- Terminal did not ask for PIN
- Transaction was declined (but the card was incorrect anyway)

Definitely worth digging deeper

OTHER APPS

Other applications

Most banks think of/are during/after implementation.

We have physically proved cloning possible in 8 apps (and 7 libs).

Others we can estimate based on libs used (PoC requires account in bank).

The easiest one

No root detection

Simple device checks

No GCM push for replenish

http://shaunthesheep.wikia.com/wiki/File:Pushing_Shirley.jpg

The hardest one

Checks multiple device characteristics

Native lib root detection

Good integrity checks and obfuscation

Had to use unrooted phone - same model, with cloned IMEI

WHAT CAN WE DO BETTER?

Don’t be the last one…

Check for more device characteristics?

- Device serial
- SIM serial/IMSI
- Display size?
- CPU?
- Sensors?

https://www.flickr.com/photos/volvob12b/11248541865/

Improve root detection

Craft your own

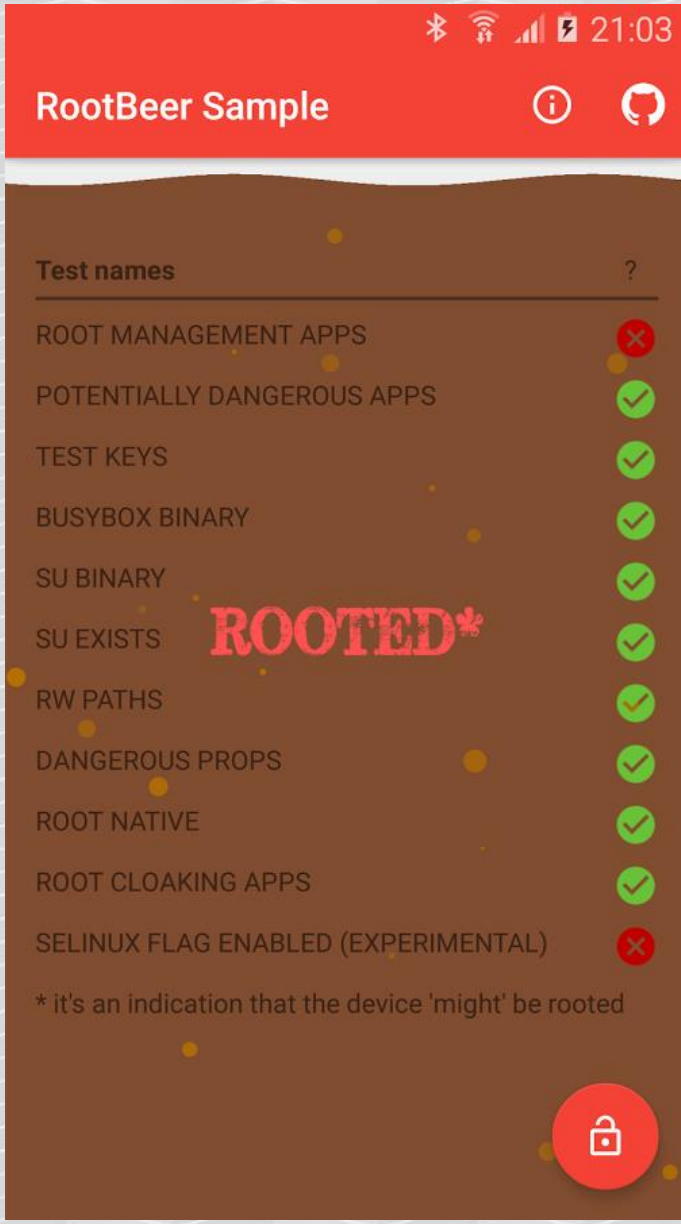
SafetyNet

- will definitely improve

RootBeer

- Open-source

https://github.com/scottyab/rootbeer/



https://play.google.com/store/apps/details?id=com.scottyab.rootbeer.sample

Integrity checks, binary protections...

Code obfuscation

Install source, signing keys

Tamper, debug detection

Notifications, reporting

Wipe on compromise

https://www.flickr.com/photos/carolynwill/1118743053

Backend - fraud management

Detect duplicated card use not enough

Device scoring - os version, patch level, bootloader unlock, installed soft

Malware handling

Behavioral analysis

https://www.flickr.com/photos/widnr/6545526341/

Future?

Devices will be more resilient, TPM?

More widespread mobile payments = more attention of fraudsters.

Hope for the best, but prepare - and verify - for the worst!

http://www.techiestate.com/spiderman-android-game/

„with great power comes great responsibility"

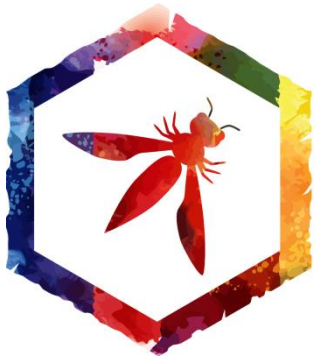
http://www.wallpaperup.com/52270/Wolves_flock_sheep_shepherd_nature_field_sky_rocks_situation_humor_wolf_people_men_males_landscapes_sky_mountains.html

See also:

How to steal mobile wallet? – Mobile contactless payments apps attack and defense

Wojtek Dworakowski
SecuRing

Friday May 12, 2017 11:35 - 12:20
Waterfront Center: Hall 1A
Hacker



OWASP
AppSec EU
Belfast
May 2017

HITB Lab: Blue Picking: Hacking Bluetooth Smart Locks

SMARTLOCKPICKING.COM

LOCATION: **Track 3 / HITB Labs**
DATE: **April 14, 2017**
TIME: **2:00 pm - 4:00 pm**

SLAWOMIR
JASEK

Lunch time!

https://www.flickr.com/photos/34739556@N04/5361451866/

securing

MORE THAN SECURITY TESTING

Thank you! Questions?

Slawomir.Jasek@securing.pl slawekja