Extracting the Painful (Blue)tooth

Matteo Beccaro || Matteo Collura
ZeroNights 2015 – November 26, 2015

Who we are ||

Matteo Beccaro

Chief Technology Officer at Opposing Force, the first Italian company specialized in offensive physical security – **Physical Intrusion** | **SE**

@_bughardy_

Who we are ||

Matteo Collura

Student in Electronic Engineering at **Politecnico di Torino**

Researcher in multiple fields –focused on calculus and modeling

@eagle1753

Agenda ||

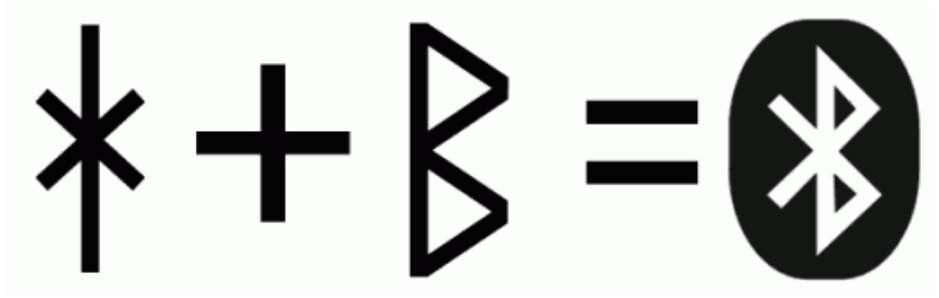
- What the hell is Bluetooth?
- Known and unknown risks
  - BlueSnarf
  - BlueBug
  - BlueChop
  - New Stuff!
- Demo
- Future work

Agenda ||

- **What the hell is Bluetooth?**
- Known and unknown risks
  - BlueSnarf
  - BlueBug
  - BlueChop
  - New Stuff!
- Demo
- Future work

What the hell is Bluetooth? ||

- Wireless standard for exchanging data over short distances
- Short wavelength UHF: 2.4 – 2.485 GHz
- 79 channels (usually) + Adaptive Frequency Hopping
- Name coming from Harald Bluetooth



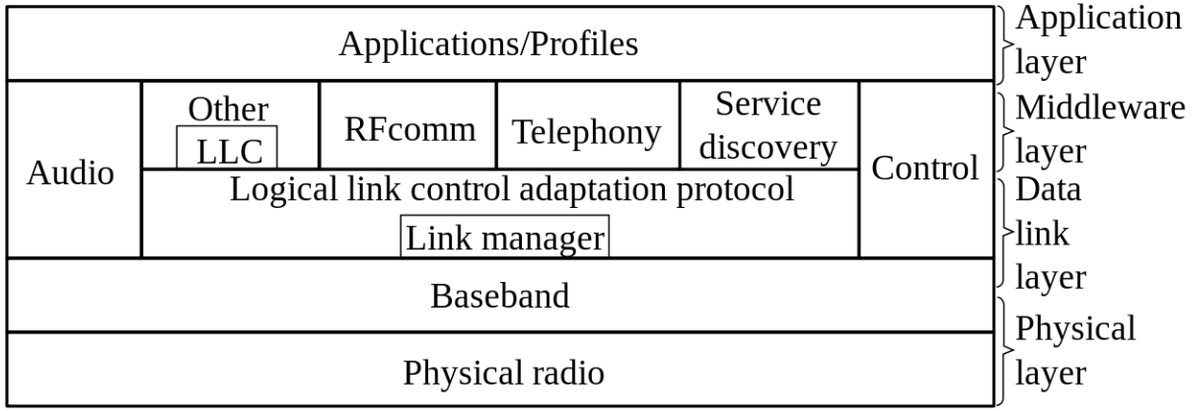
Scandinavian humour... ☺

What the hell is Bluetooth? ||

Layer protocol architecture

- Core protocols
- Cable replacement protocols
- Telephony control protocols
- Adopted protocols

So many different stacks!



LMP, L2CAP, SDP are mandatory!

What the hell is Bluetooth? ||

Such updates!

Version 1:

- 1.0: Mandatory BD_ADDR
- 1.1: IEEE Standard (2002)
- 1.2: Adaptive frequency-hopping spread spectrum → resistance to interferences and eavesdropping (theoretically ☺)

Version 2:

- 2.0: EDR (optional) for faster data transfer, GFSK+PSK modulation
- 2.1: Secure Simple Pairing, Extended Inquiry Response

What the hell is Bluetooth? ||

Such updates!

Version 3: → 3.0: Alternative MAC/PHYs for high data transfer, Unicast Connectionless Data

Version 4:

- 4.0: Includes now Bluetooth Low Energy protocol (or Smart)
- 4.1: Limited discovery time, lower consumptions
- 4.2: LE Data packet extension, LE «secure» connections

Agenda ||

- What the hell is Bluetooth?
- **Known and unknown risks**
  - **BlueSnarf**
  - **BlueBug**
  - **BlueChop**
  - **New Stuff!**
- Demo
- Future work

Known and unknown risks ||

BlueSnarf, by Holtmann & Laurie

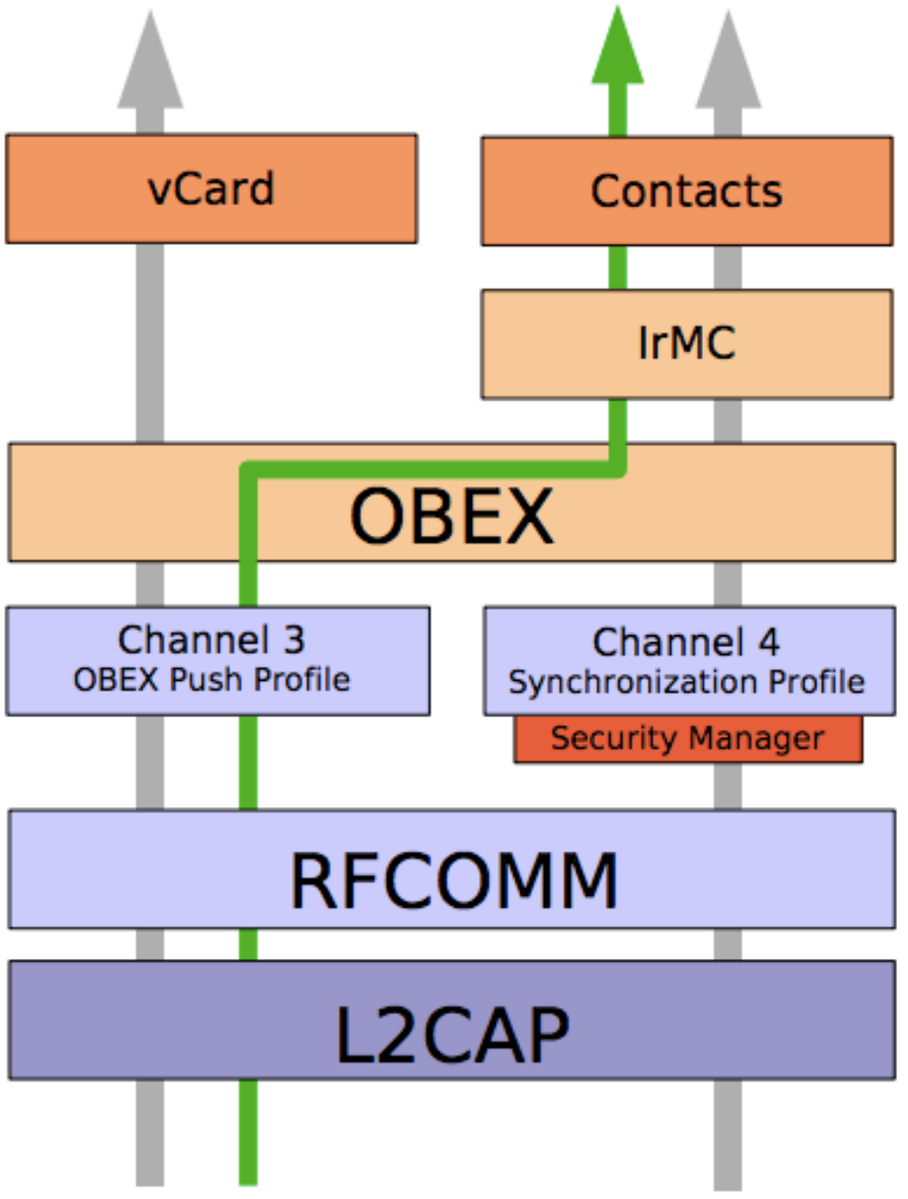
When? → Late 2003

What? → Bluetooth implementation on mobile phones and pocket palms

Why? → «security» of OBEX protocol

- Easy GET requests to common files (calendar, contacts..)
- No authentication needed
- No prompts on the user's side

Known and unknown risks ||



Kindly stolen from
Trifinite group

Known and unknown risks ||

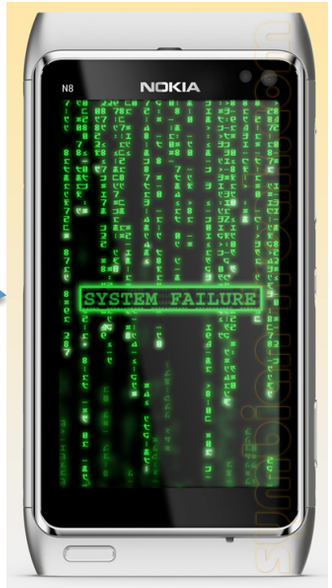
BlueBug, by Adam Laurie & Martin Herfurt

When? → 2004 @Defcon12

What? → Bluetooth implementation on mobile phones, especially Symbian OS

Why? → Security loophole

- No secure auth. prior to v2.0
- Control device through plain serial connection
- Download items via OBEX protocol w/out prompts



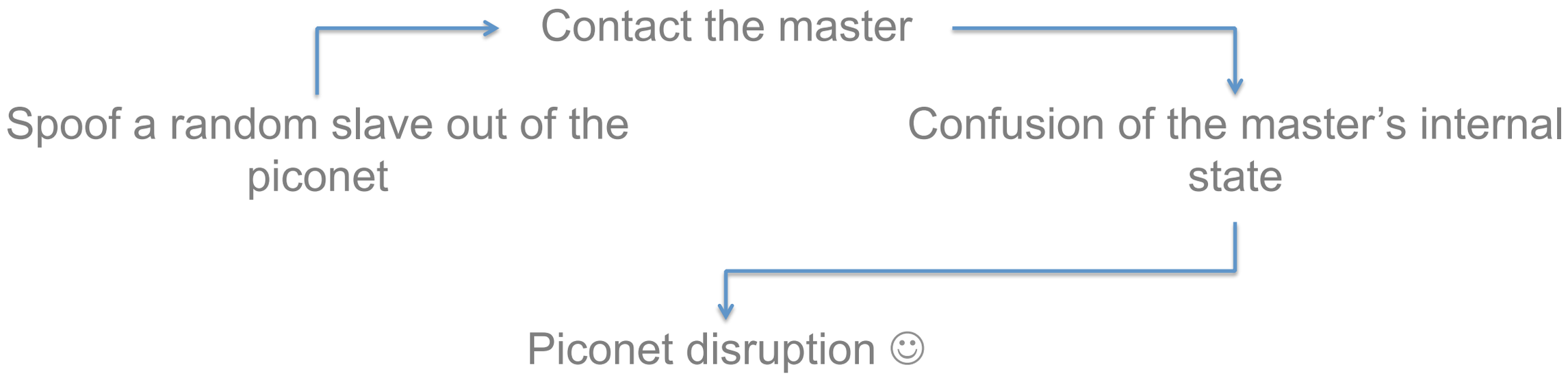
BlueBug™

Known and unknown risks ||

BlueChop, following BlueSnarf

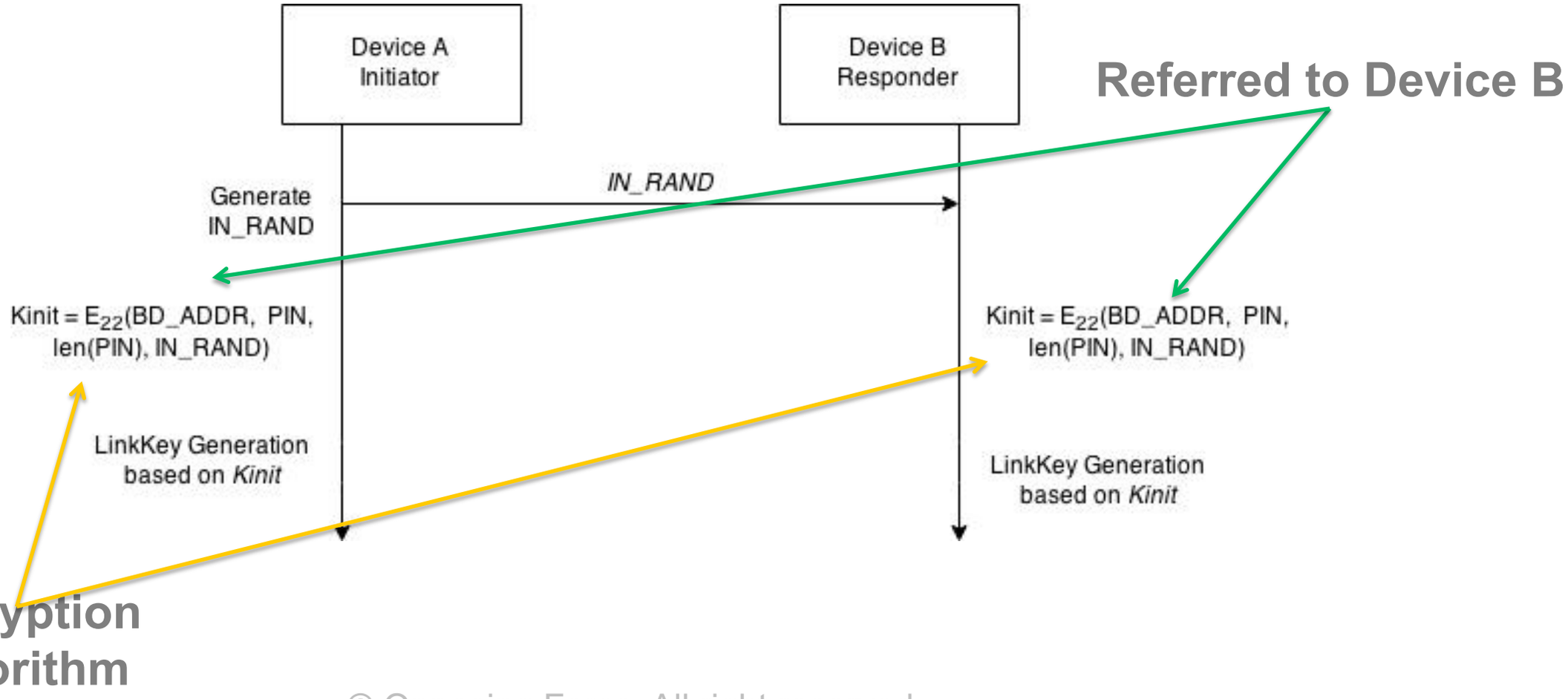
What? → It disrupts any bluetooth piconet from the outside

Provided → Master must support multiple connections



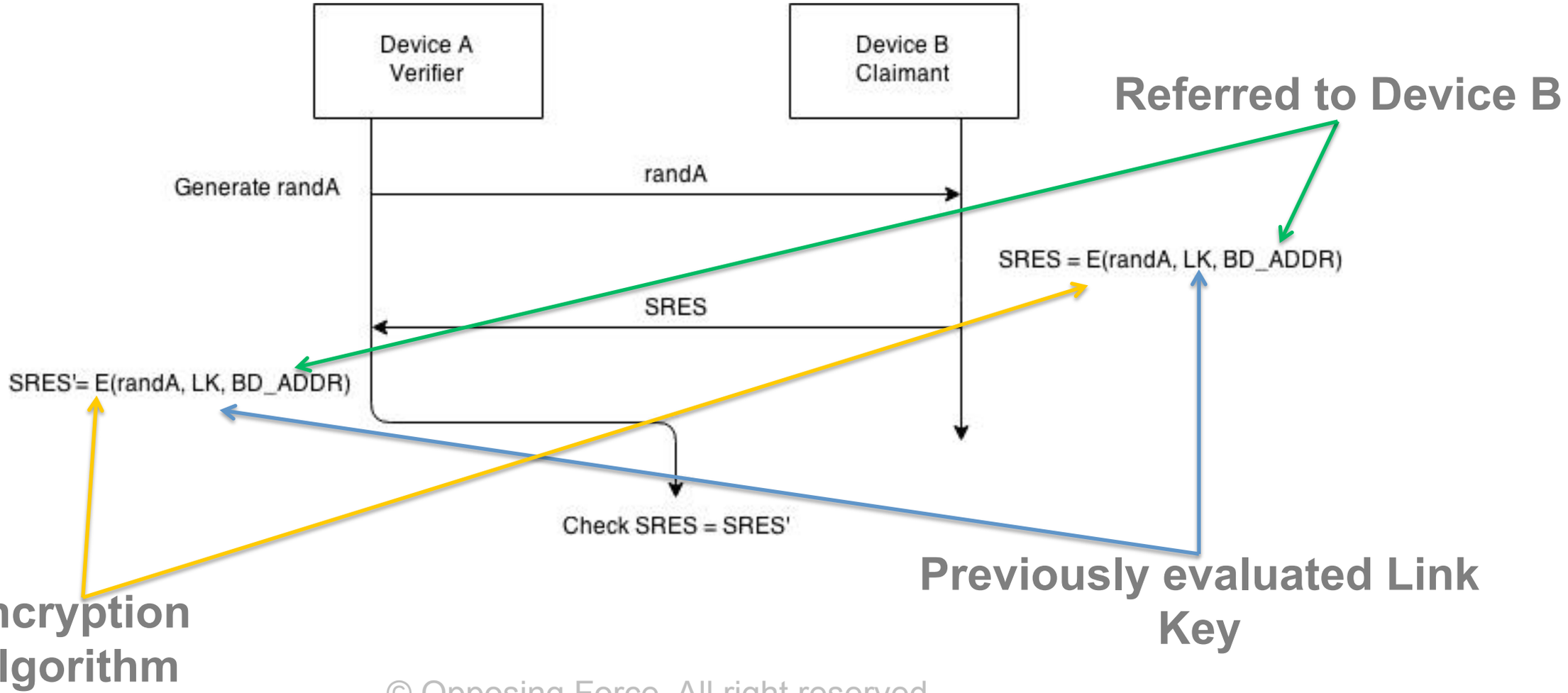
Known and unknown risks ||

Legacy (prior to v2.0) pairing procedure:



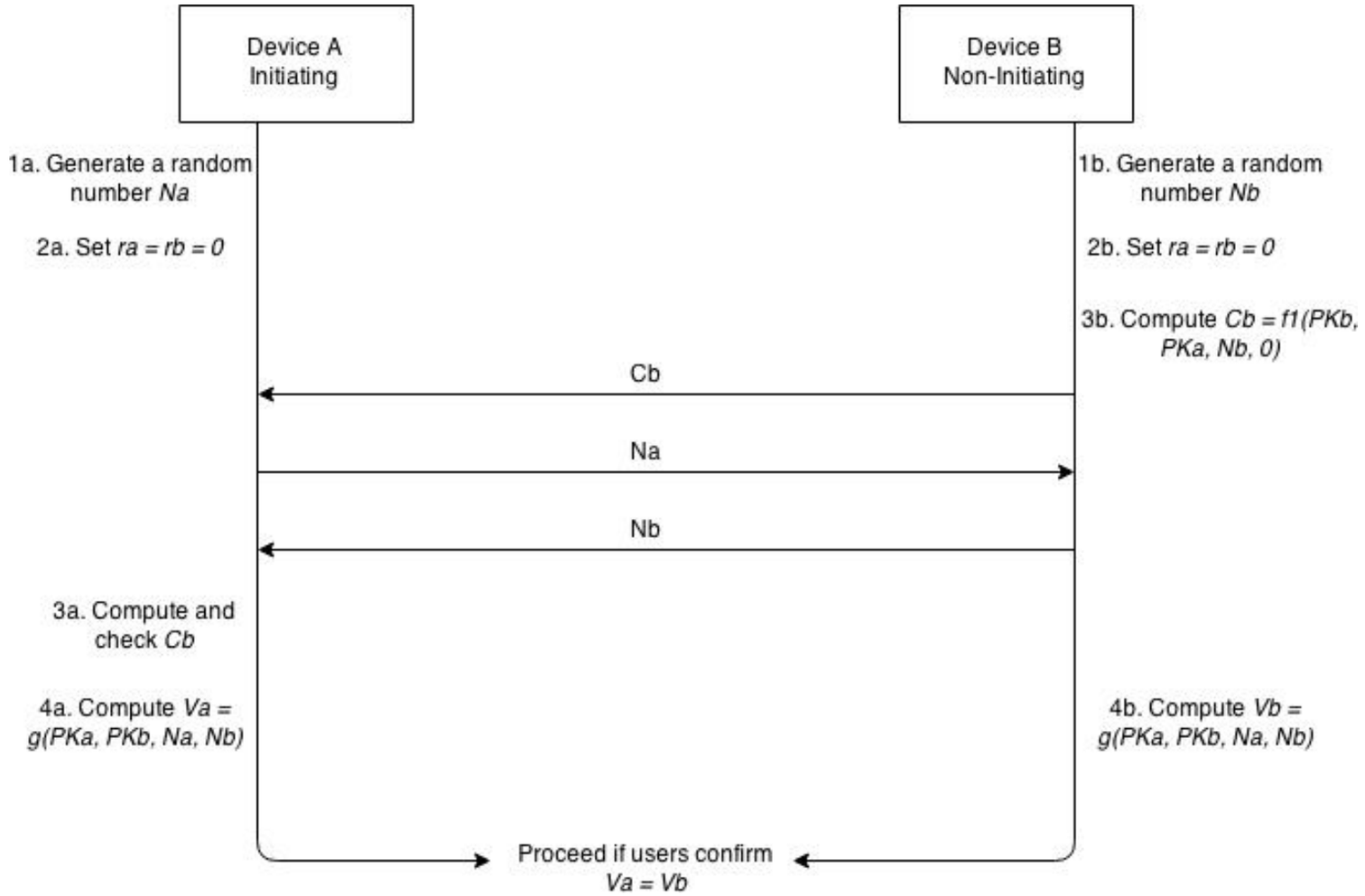
Known and unknown risks ||

Legacy (prior to v2.0) authentication procedure:



Known and unknown risks ||

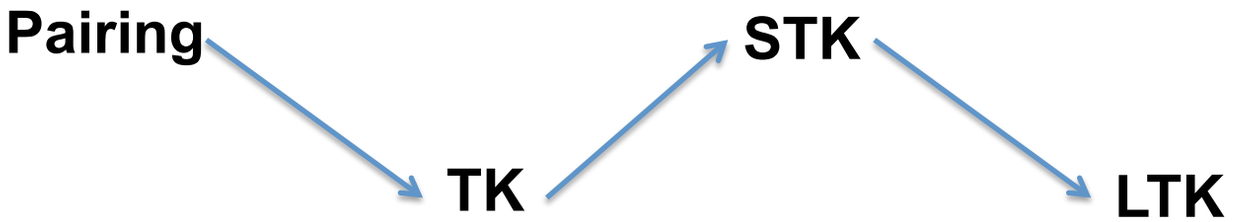
Secure simple pairing:



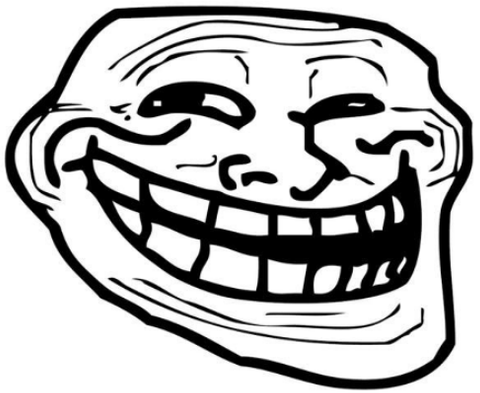
Known and unknown risks ||

Bluetooth LE encryption bypass, by Mark Ryan:

- Eavesdropping vs Decrypting
- 3 different keys needed to establish a connection, TK, STK, LTK
- If we are able to save the key exchange procedure, we are done ☺

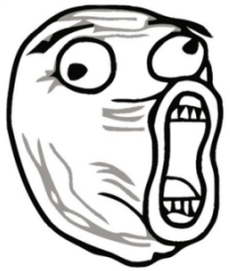


What if I get TK?



Known and unknown risks ||

TK, 128 bit AES key, depends on the pairing mode:

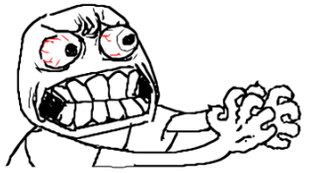


Just Works
TK = 0

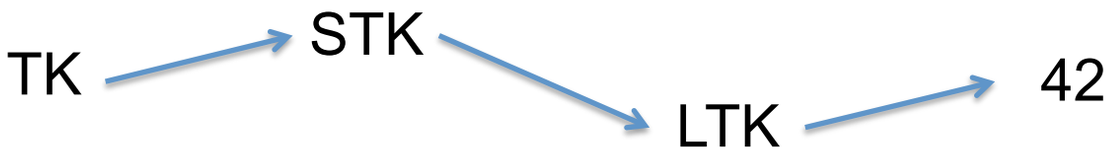
6-digit PIN

TK = 128-bit number

Out Of Band (OOB)
TK = #fuckyourself



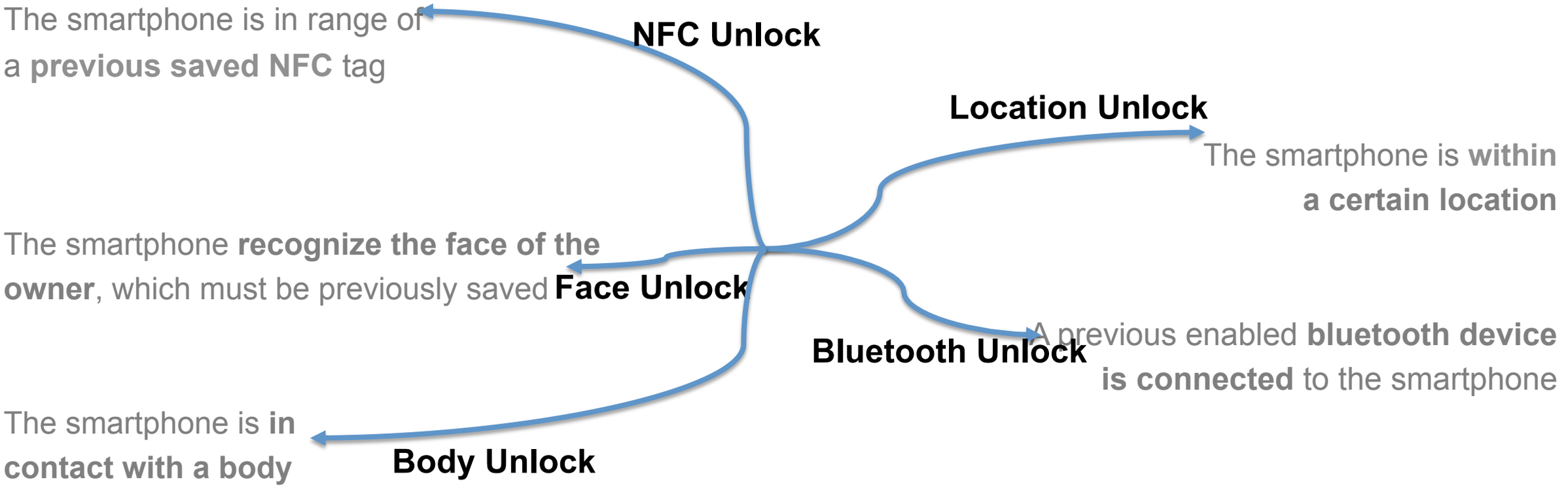
Bruteforce is the way. Intel i7, just one core → less than 1 sec



The whole procedure may be computed offline

SmartUnlock ||

Officially introduced with Android 5.0 it enables to unlock the smartphone without user interaction if at least one of the following conditions apply:



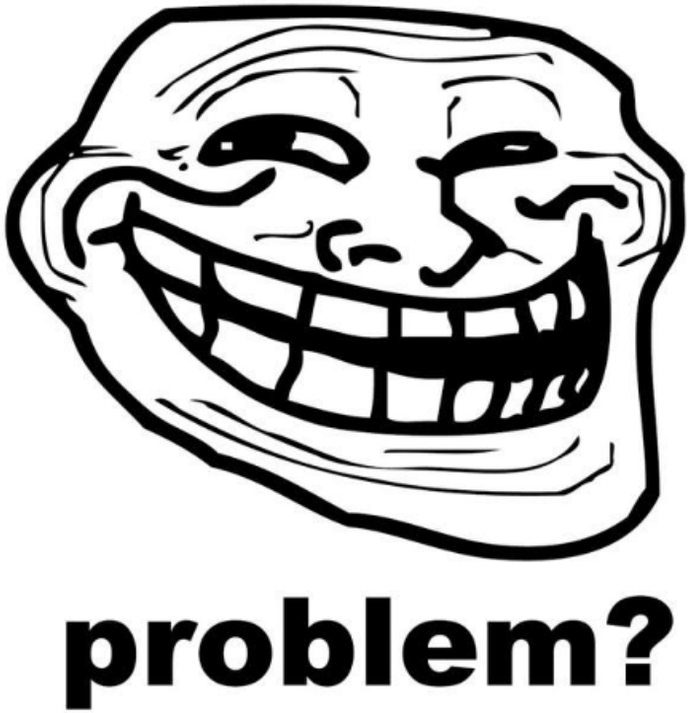
Bluetooth Unlock

This may be the most interesting and most used function of all the above

The user set a paired bluetooth device as **Trusted**, and from now on every time that device is linked to the smartphone the lockscreen is bypassed

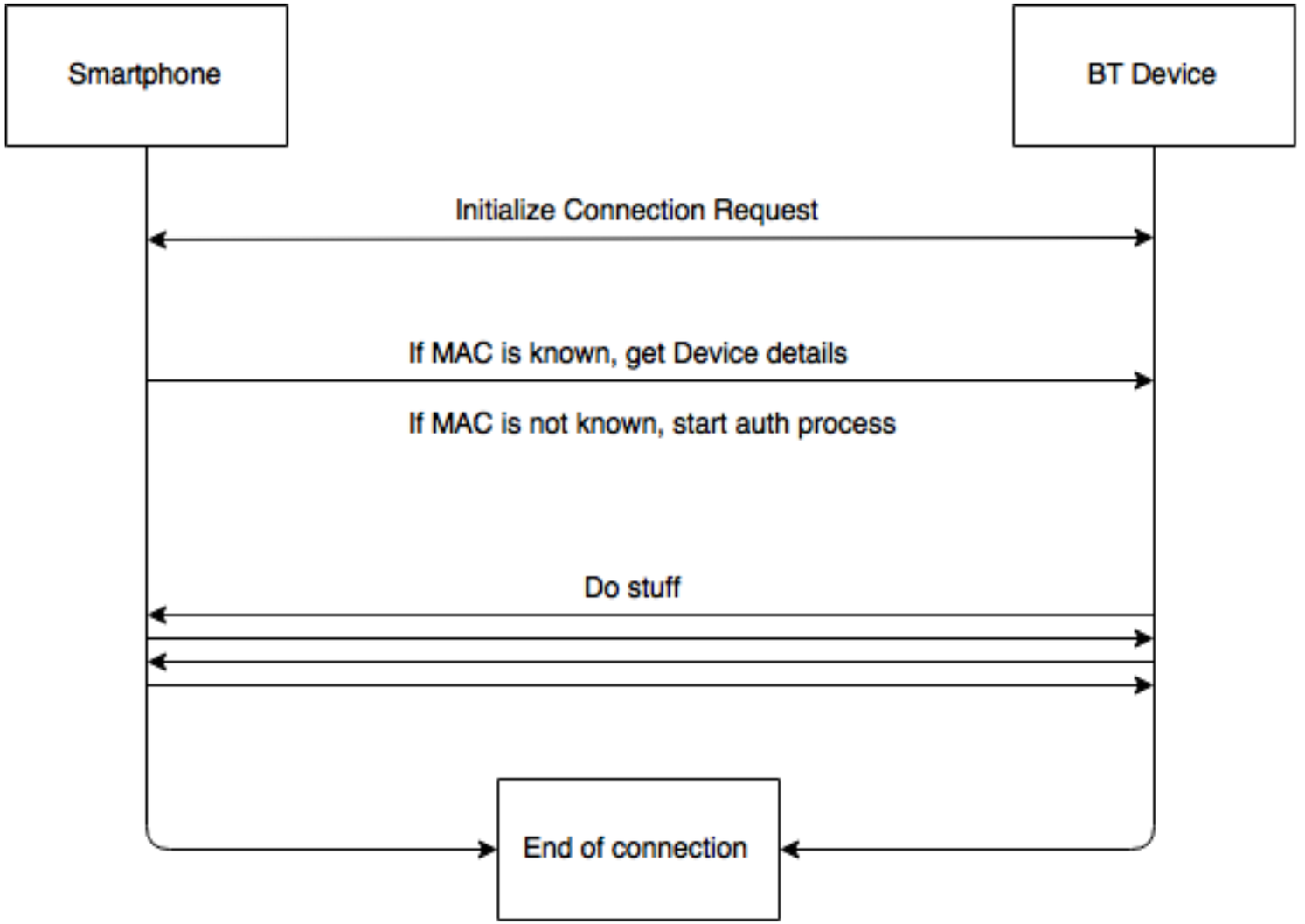
Good, so.. what is the problem?

Bluetooth unlock ||



In Android < 5.1 the LK (LinkKey) **is not checked** to verify the Bluetooth device

Bluetooth unlock ||



Bluetooth unlock ||

Now the question is:

How to get the 4 bytes of the MAC address required?

Two possible solutions:

Bruteforce

- Slow
- Expensive
- Not such a good idea

Sniffing

- Requires vicinity
- Target can become aware
- Authentication process is required

Bruteforce ||

Slow → We cannot bruteforce the MAC address offline, we need to try a new connection every time

Expensive → We can speed it up parallelizing it but costs increase

Not such a good idea → 42 bits definitely requires too much time

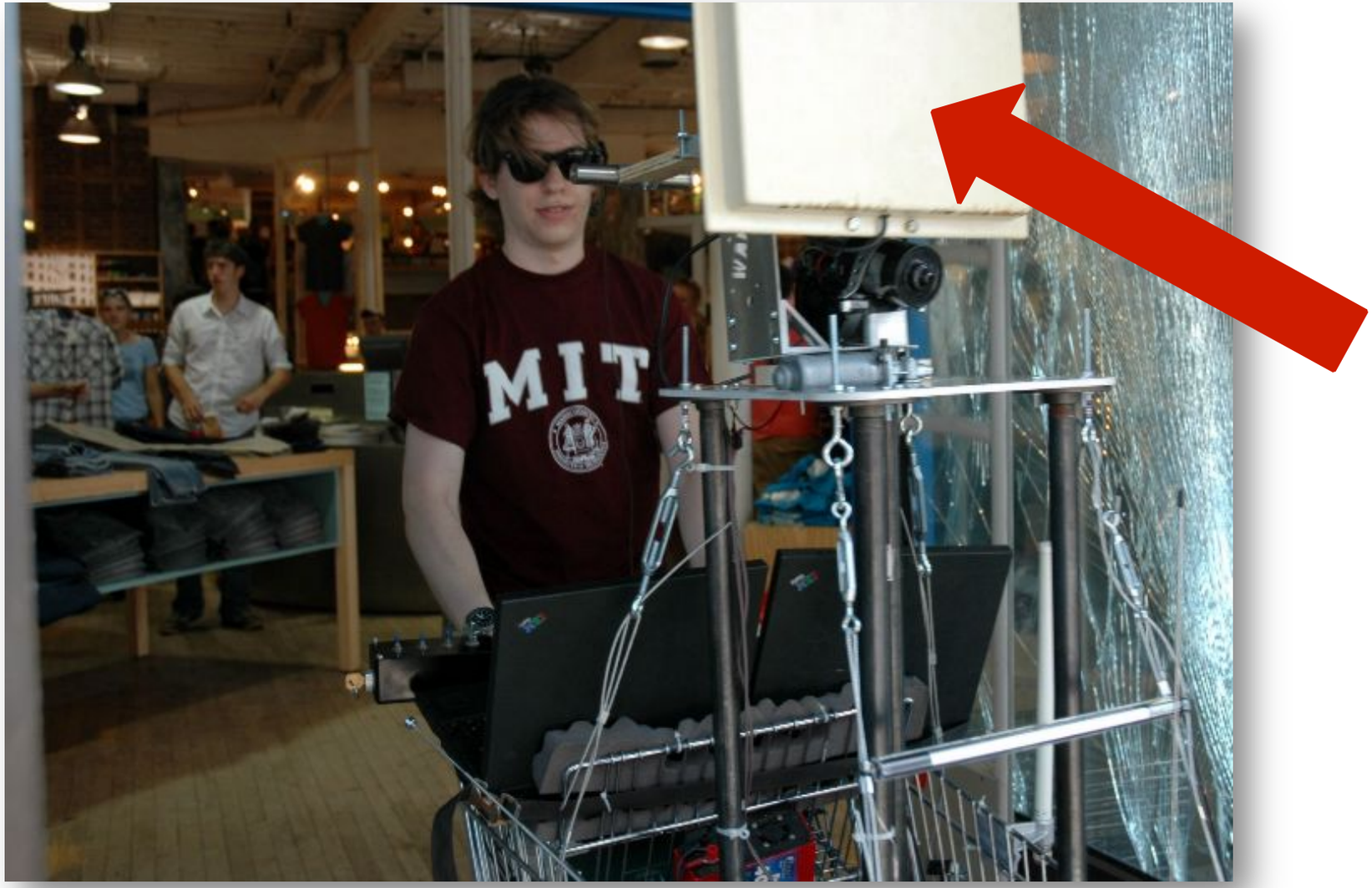
Sniffing ||

Requires vicinity → Target must be near enough for our ubertooth to intercept packets

Auth. process is required → Usually only 3 bytes of MAC address are transmitted

Target can become aware → Target can be suspicious of strange guy with big antenna(s)

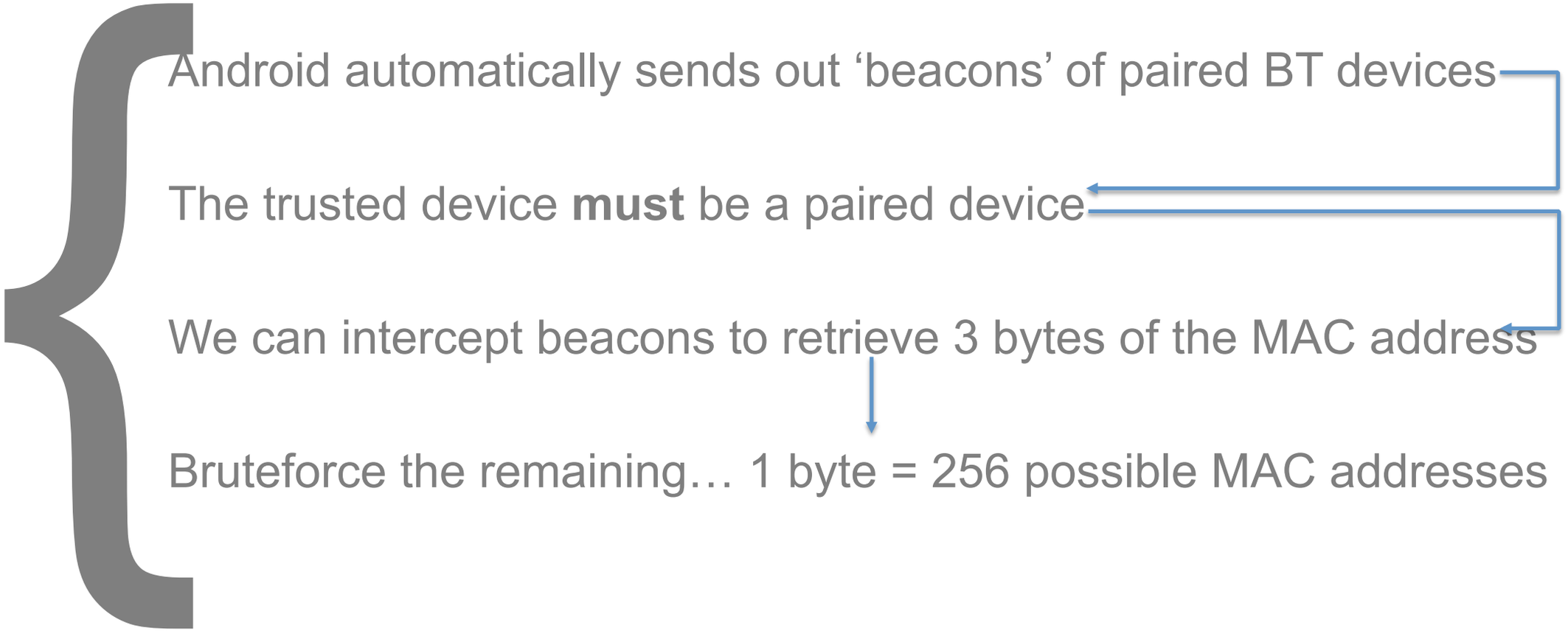
Sniffing ||



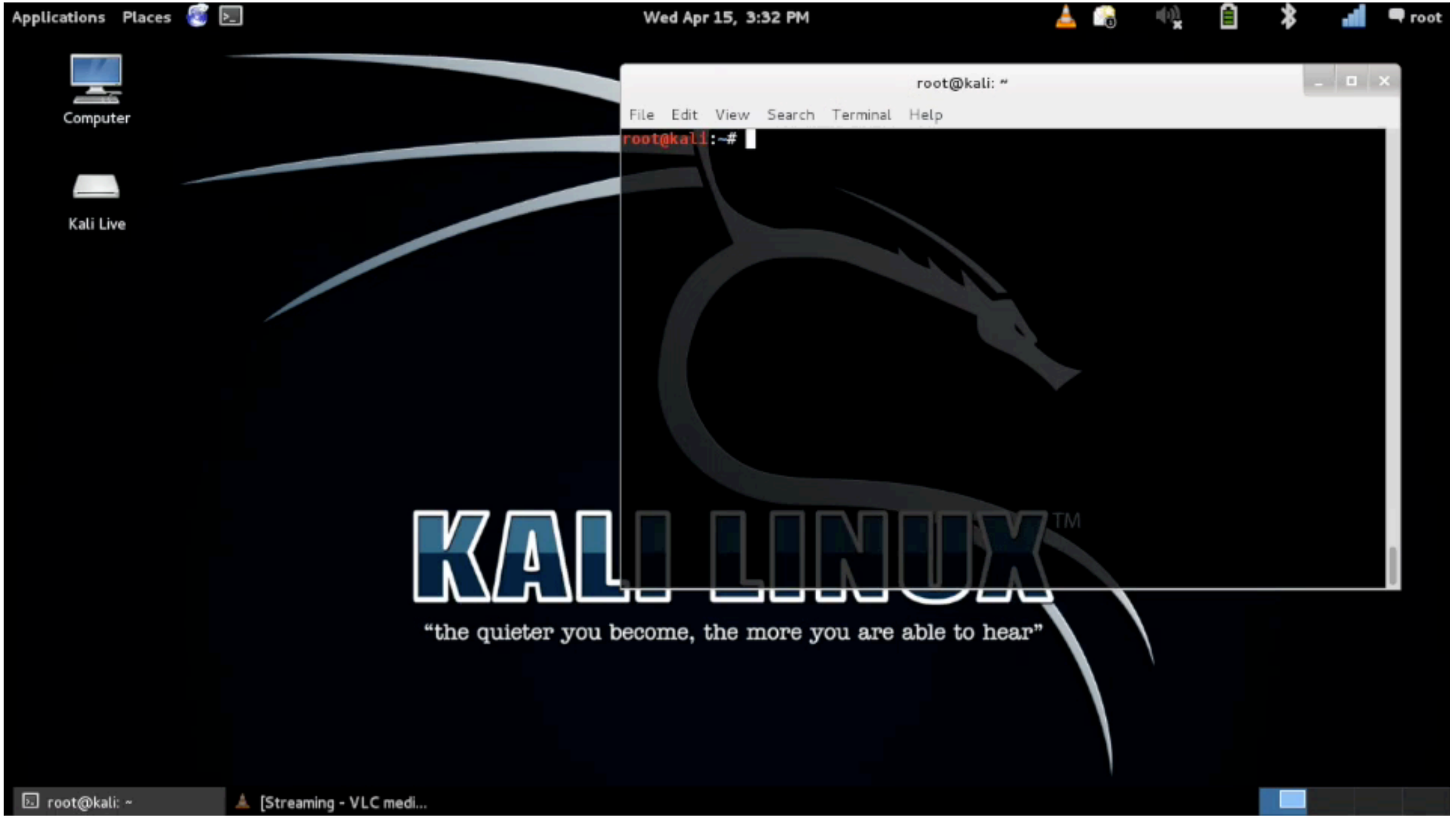
Strange guy with
big antenna

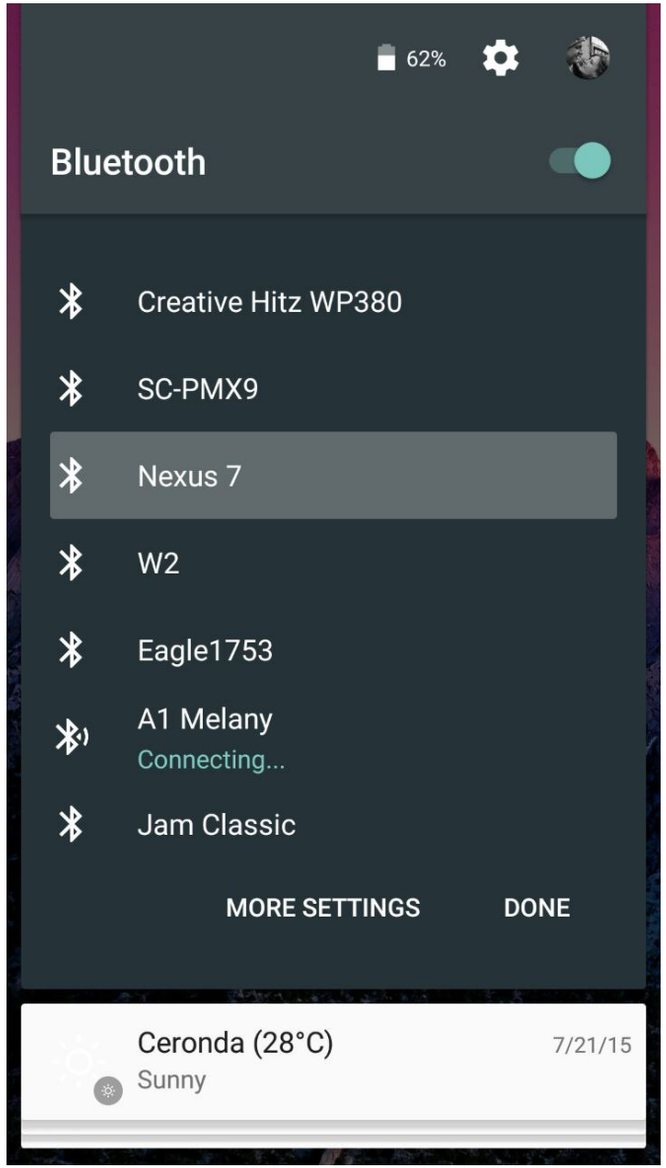
Our approach ||

Hybrid is always the **solution**



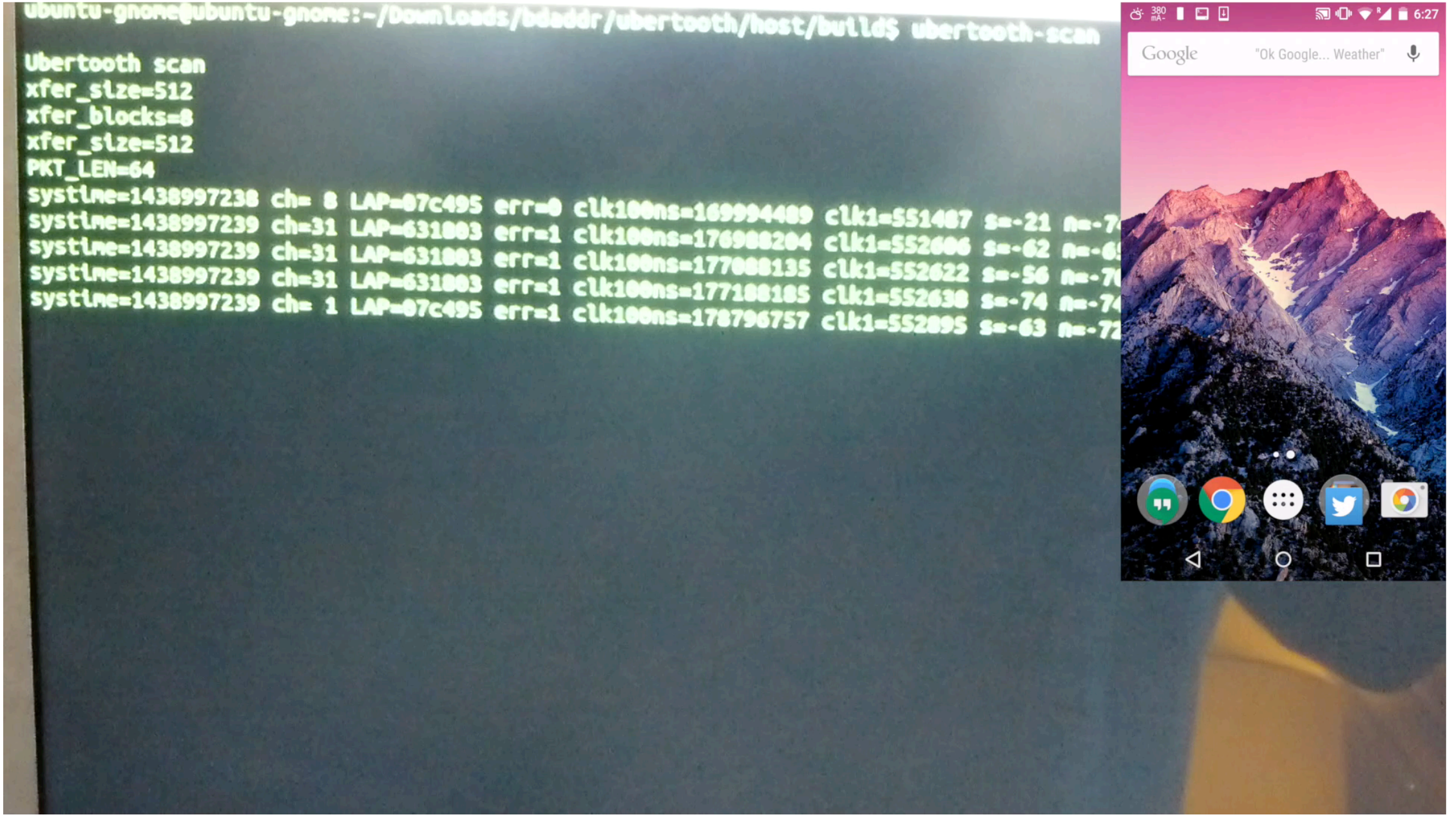
DEMO Time ||


Applications
Places
Wed Apr 15, 3:32 PM
root
Computer
Kali Live
root@kali: ~
File Edit View Search Terminal Help
root@kali:~#
KALI LINUX™
"the quieter you become, the more you are able to hear"
root@kali: ~
[Streaming - VLC medi...

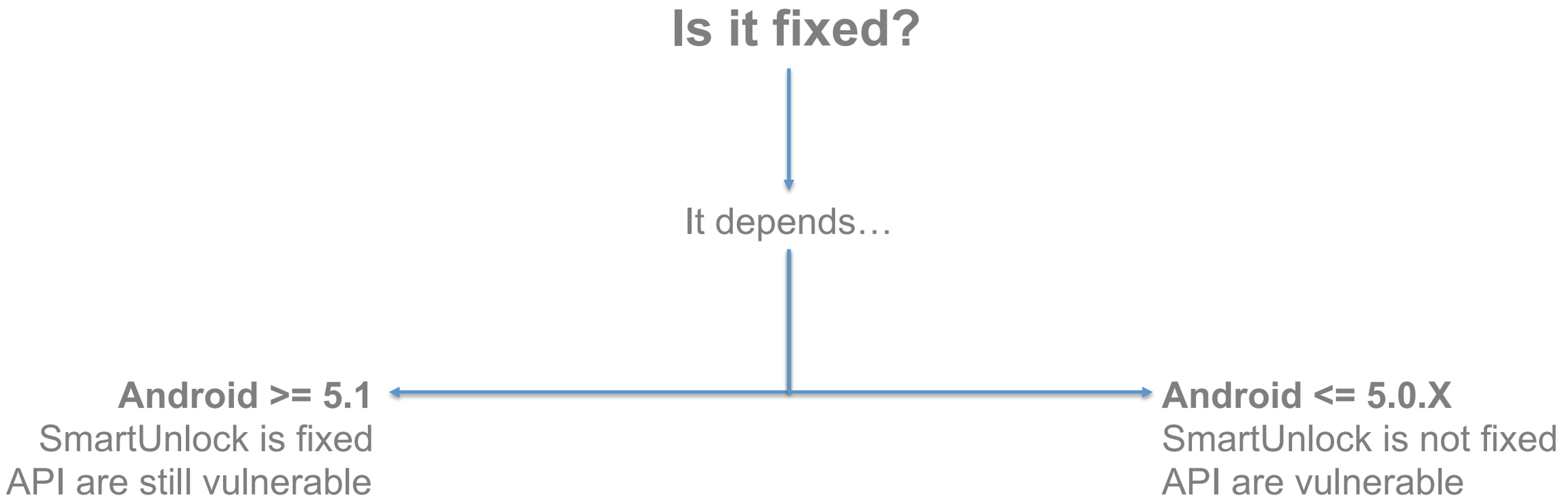


Android 5.1 adds a new nice feature...

DEMO time ||


ubuntu-gnome@ubuntu-gnome:~/Downloads/bdaddr/ubertooth/host/build$ ubertooth-scan
Ubertooth scan
xfer_size=512
xfer_blocks=8
xfer_size=512
PKT_LEN=64
systime=1438997238 ch= 8 LAP=07c495 err=0 clk100ns=169994489 clk1=551487 s=-21
systime=1438997239 ch=31 LAP=631803 err=1 clk100ns=176988204 clk1=552606 s=-62
systime=1438997239 ch=31 LAP=631803 err=1 clk100ns=177088135 clk1=552622 s=-56
systime=1438997239 ch=31 LAP=631803 err=1 clk100ns=177188185 clk1=552638 s=-74
systime=1438997239 ch= 1 LAP=07c495 err=1 clk100ns=178796757 clk1=552895 s=-63
6:27
Google
"Ok Google... Weather"

New findings ||

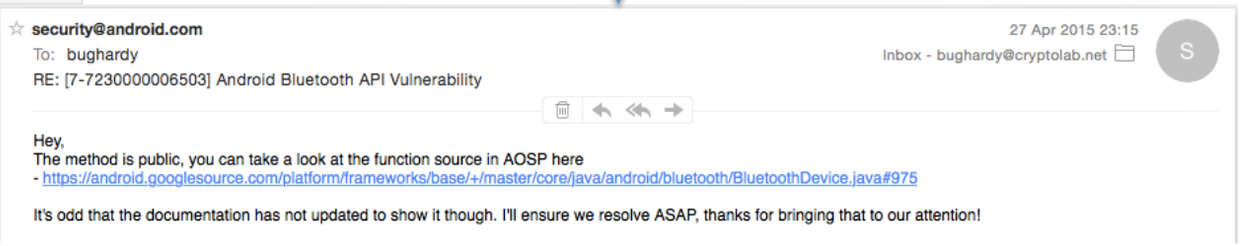


New findings ||

Summary

Constants		
String	ACTION_ACL_CONNECTED	Broadcast Action: Indicates a low level (ACL) connection has been established with a remote device.
String	ACTION_ACL_DISCONNECTED	Broadcast Action: Indicates a low level (ACL) disconnection from a remote device.
String	ACTION_ACL_DISCONNECT_REQUESTED	Broadcast Action: Indicates that a low level (ACL) disconnection has been requested for a remote device, and it will soon be disconnected.
String	ACTION_BOND_STATE_CHANGED	Broadcast Action: Indicates a change in the bond state of a remote device.
String	ACTION_CLASS_CHANGED	Broadcast Action: Bluetooth class of a remote device has changed.
String	ACTION_FOUND	Broadcast Action: Remote device discovered.

API does not have a safe method to check if a device is connected with a proper LK



☆ **security@android.com** 27 Apr 2015 23:15

To: bughardy Inbox - bughardy@cryptolab.net

RE: [7-7230000006503] Android Bluetooth API Vulnerability

Hey,
The method is public, you can take a look at the function source in AOSP here
- https://android.googlesource.com/platform/frameworks/base/+/master/core/java/android/bluetooth/BluetoothDevice.java#975

It's odd that the documentation has not updated to show it though. I'll ensure we resolve ASAP, thanks for bringing that to our attention!

```
975.     public boolean isEncrypted() {
976.         if (sService == null) {
977.             // BT is not enabled, we cannot be connected.
978.             return false;
979.         }
980.         try {
981.             return sService.getConnectionState(this) > CONNECTION_STATE_CONNECTED;
982.         } catch (RemoteException e) {
983.             Log.e(TAG, "", e);
984.             return false;
985.         }
986.     }
987.
```

Android Security Team told us that there is a method f
or
this, but it was not yet in SDK, as 27th April, 2015. And
it's
still not present

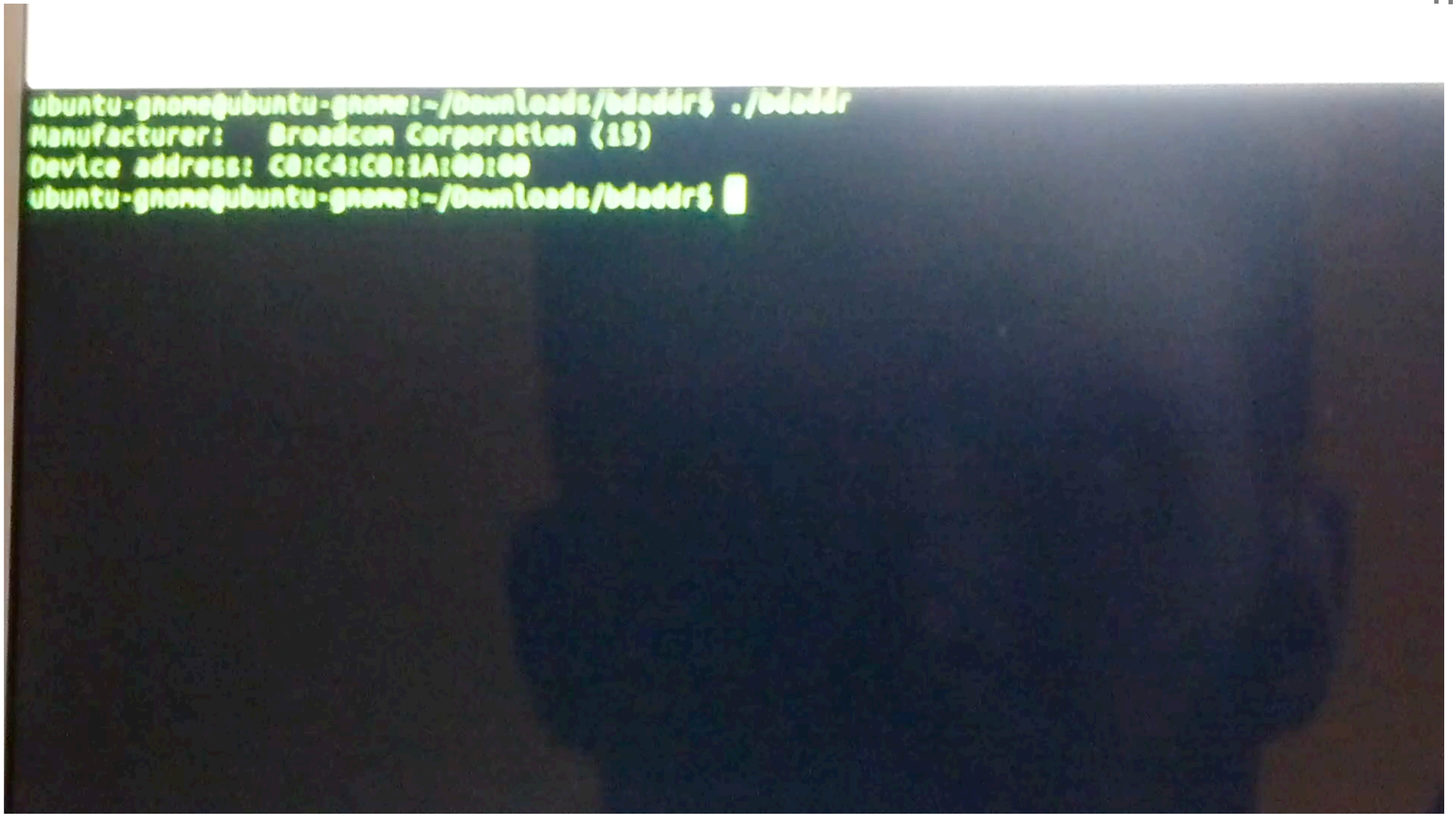
DEMO time ||

Why fixing the API is important if SmartUnlock function is fixed?

3rd party applications!

Demo time!

DEMO time ||


ubuntu-gnome@ubuntu-gnome:~/Downloads/bdaddr$ ./bdaddr
Manufacturer: Broadcom Corporation (15)
Device address: C0:C4:C0:1A:00:00
ubuntu-gnome@ubuntu-gnome:~/Downloads/bdaddr$

Agenda ||

- What the hell is Bluetooth?
- Known and unknown risks
  - BlueSnarf
  - BlueBug
  - BlueChop
  - New Stuff!
- Demo
- **Future work**

Future work ||

Bluetooth is everywhere, we are focusing on

- IoT Devices
- Smart Locks
- Fit Band
- Etc.

Thanks

Opposing Force - challenging your security

https://opposingforce.it | engage@opposingforce.it

Q&A Time!

Opposing Force - challenging your security

https://opposingforce.it | engage@opposingforce.it

Thanks to our sponsor

https://

beyondsecurity.com