YOU'D BETTER SECURE YOUR BLE DEVICES OR WE'LL KICK YOUR BUTTS !

@virtualabs | DEF CON 26, Aug. 12th 2018

digital.security

WHO AM I ?

Head of R&D @ Econocom Digital Security

Studying **Bluetooth Low Energy** for 3 years

Developer & maintainer of **BtleJuice**

Having fun with Nordic's **nRF51822** ☺

digital.security

AGENDA

BLE sniffing 101

Improving the BLE arsenal

- Sniffing BLE connections in 2018
- Introducing **BtleJack**, a flexible sniffing tool

BtleJacking: a brand new attack

- How it works
- Vulnerable devices & demos

Recommendations

BLE SNIFFING 101

digital.security

MUCH CHEAP TOOLS, (NOT) WOW RESULTS

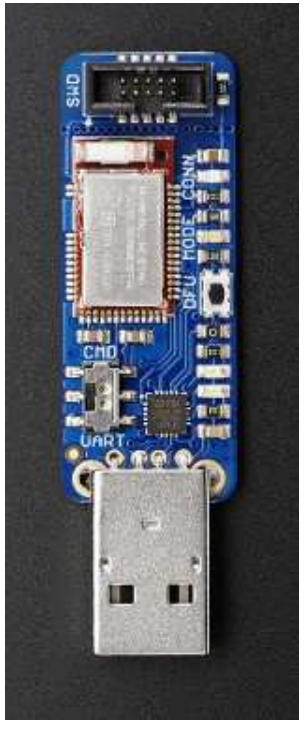
- Sniffing **existing/new connections** with an **Ubertooth One**
- Sniffing **new connections** with an Adafruit's **Bluefruit LE Sniffer**
- Sniffing BLE packets with **gnuradio**

UBERTOOTH ONE

- Sniffs existing and new connections
- Does **not** support *channel map updates*
- Costs **$120**

BLUEFRUIT LE SNIFFER

- Up-to-date software (Nov. 2017)
- **Proprietary** firmware from Nordic Semiconductor
- Sniffs only **new connections**
- Costs **$30 - $40**



SOFTWARE DEFINED RADIO

- Sniffs **only** BLE advertisements
- Unable to **follow** any existing/new connection
- Latency
- Requires 2.4GHz compatible SDR device

BLE SNIFFING 101

BLE is designed to make sniffing difficult:

- 3 separate advertising channels
- Uses Frequency Hopping Spread Spectrum (**FHSS**)
- Master or slave can **renegotiate** some parameters at any time

Sniffing BLE connections is either hard or expensive

MAN IN THE MIDDLE

"Watch where you're going, Larry — you walked right through my wireless data stream!"

digital.security

HOW BLE MITM WORKS

- Discover the target device (advertisement data, services & characteristics)
- Connect to this target device, it is not advertising anymore (connected state)
- Advertise the same device, await connections and forward data

BTLEJUICE

localhost:8080/#

BtleJuice

Action	Service	Characteristic	Data
Connected			
notification	180f	2a19	.G
read	180f	2a19	.G
read	7b122568-6677-7f8c-f8e9-af0eedb36e3a	7b121991-6677-7f8c-f8e9-af0eedb36e3a	01 06
read	7b122568-6677-7f8c-f8e9-af0eedb36e3a	7b121993-6677-7f8c-f8e9-af0eedb36e3a	00 00 00 00
read	7b122568-6677-7f8c-f8e9-af0eedb36e3a	7b121998-6677-7f8c-f8e9-af0eedb36e3a	13
write	1803	2a06	02
write	b0ad1523-99b2-7e1d-fc0d-6d399e1edf02	b0ad1525-99b2-7e1d-fc0d-6d399e1edf02	00

https://github.com/DigitalSecurity/btlejuice

GATTACKER

```
<<<<<<<<<<<<<<<<<< INITIALIZED >>>>>>>>>>>>>>>>>>>>>>>
Client connected: 72:e5:36:f5:05:23
>> Write:  ffe0 -> fff1 : a137343136383905789a3b246c6c17164f0121 ( 741689 x ;$ll  0 !)
<< Read:   ffe0 -> fff1 : a20500f0c77f162e8bd21110841e641e641480 (       .      d d  )
>> Write:  ffe0 -> fff1 : a137343136383909bcaafbae83b5babc02b8f7a0 ( 741689             )
<< Read:   ffe0 -> fff1 : a20900 (    )
>> Write:  ffe0 -> fff1 : a136363636363606 ( 666666 )
<< Read:   ffe0 -> fff1 : a206002c010000 (    ,    )
>> Write:  ffe0 -> fff1 : a136363636363606 ( 666666 )
<< Read:   ffe0 -> fff1 : a206002c010000 (    ,    )
>> Write:  ffe0 -> fff1 : a136363636363606 ( 666666 )
<< Read:   ffe0 -> fff1 : a206002c010000 (    ,    )
>> Write:  ffe0 -> fff1 : a136363636363601 ( 666666 )
<< Read:   ffe0 -> fff1 : a20100 (    )
```

https://github.com/securing/gattacker

Pros:

- Get rid of the **3 advertising channels issue**
- You see **every BLE operation** performed
- You may **tamper on-the-fly** the data sent or received

Cons:

- **Complex** to setup: 1 VM & 1 Host computer
- Only capture **HCI events**, not BLE Link Layer
- Does not support all types of **pairing**
- Only compatible with 4.0 adapters

WE ARE DOING IT WRONG !

- *Ubertooth-btle* **is outdated** and does not work with recent BLE stacks
- Nordic Semiconductor' sniffer is **closed source** and does not allow active connection sniffing and **may be discontinued**
- The MitM approach seems great but **too difficult** to use and does not intercept link-layer packets

IMPROVING THE BLE ARSENAL

digital.security

THE IDEAL TOOL

- Able to sniff existing and new connections
- Uses cheap hardware
- Open-source

SNIFFING ACTIVE CONNECTIONS

MIKE RYAN'S TECHNIQUE

LSB MSB

Preamble (1 octet)	Access Address (4 octets)	PDU (2 to 257 octets)	CRC (3 octets)

1. Identify Access Address (32 bits)
2. Recover the *CRCInit* value used to compute CRC
3. hopInterval = time between two packets / 37
4. hopIncrement = LUT[time between channel 0 & 1]

MIKE'S ASSUMPTION (2013)

All 37 data channels are used

DATA CHANNELS IN 2018

- Not all channels are used to improve reliability
- Some channels are *remapped* to keep a 37 channels hopping sequence

0, **4**, 8, 12, 16, 20, 24, 0, **4**, 8, 3, 7, 11, 15, 19, 23, 27, 3, 7, 2, 6, 10, 14, 18, 22, 26, 2, 6, 1, 5, 9, 13, 17, 21, 25, 1, 5

Mike's technique does not work anymore !

HOW TO DEDUCE CHANNEL MAP AND HOP INTERVAL

- **Channel map**
  - Listen for packets on every possible channels
  - May take until 4 x 37 seconds to determine !

- **Hop interval**
  - Find a unique channel
  - Measure time between 2 packets and divide by 37

DEDUCE HOP INCREMENT

- Pick 2 unique channels
- Generate a lookup table
- Measure time between two packets on these channels
- Determine increment value

More details in **PoC||GTFO 0x17**

SNIFFING NEW CONNECTIONS

CONNECT_REQ PDU

LLData									
AA (4 octets)	CRCInit (3 octets)	WinSize (1 octet)	WinOffset (2 octets)	Interval (2 octets)	Latency (2 octets)	Timeout (2 octets)	ChM (5 octets)	Hop (5 bits)	SCA (3 bits)

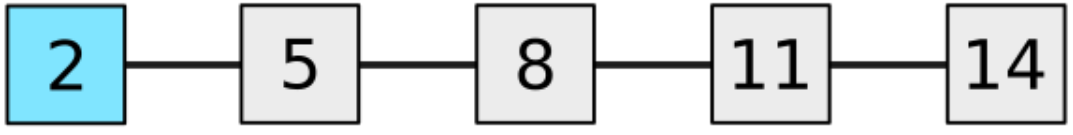
Figure 2.11: LLData field structure in CONNECT_REQ PDU's payload

- Every needed information are in this packet
- Sniffer must listen on the correct channel

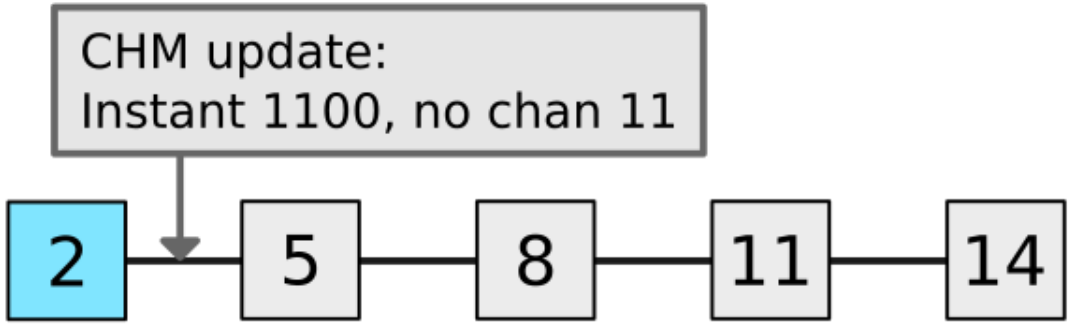
"INSTANT" MATTERS

- Defines when a parameter update is effective
- Used for:
  - Channel map updates
  - Hop interval updates

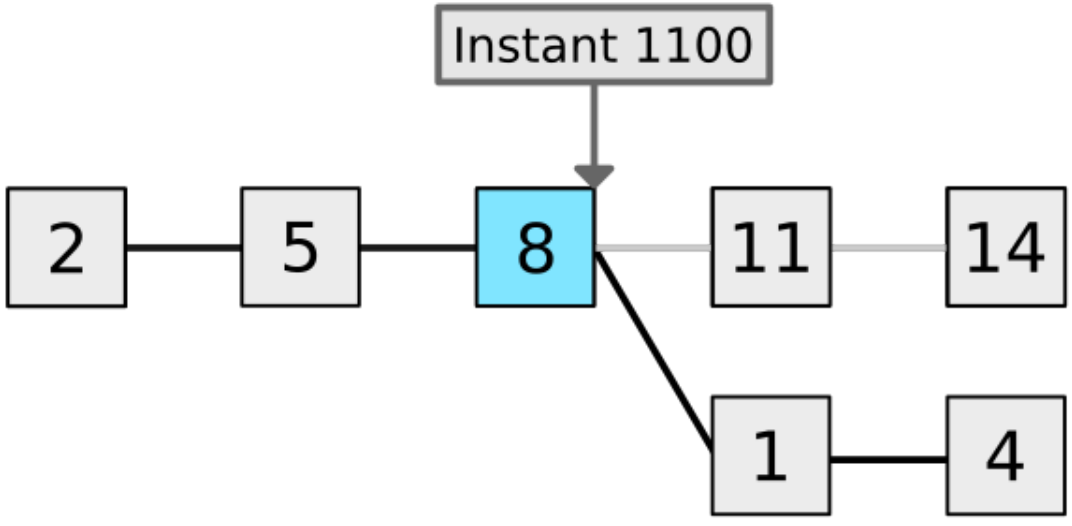
WE DON'T CARE AT ALL



WE DON'T CARE AT ALL

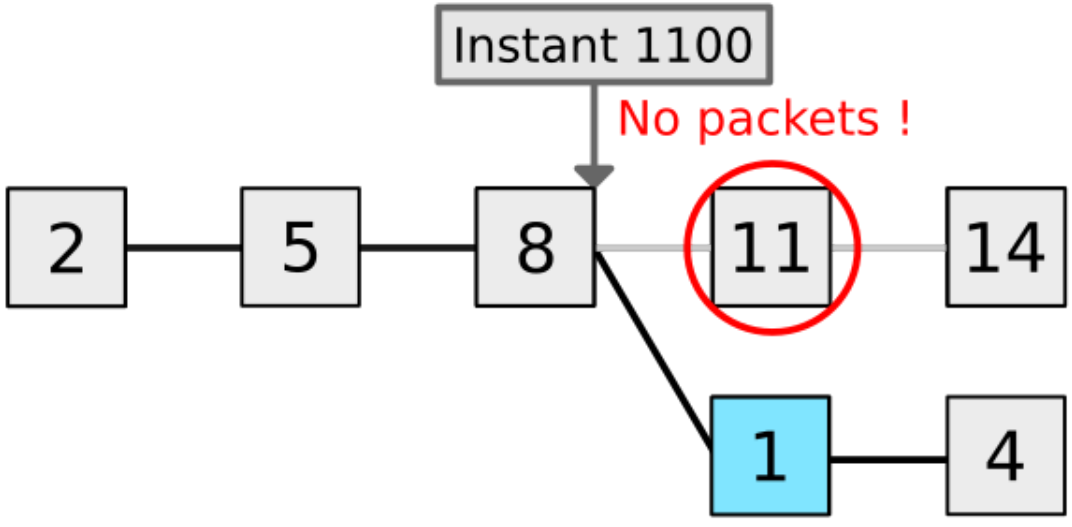


WE DON'T CARE AT ALL

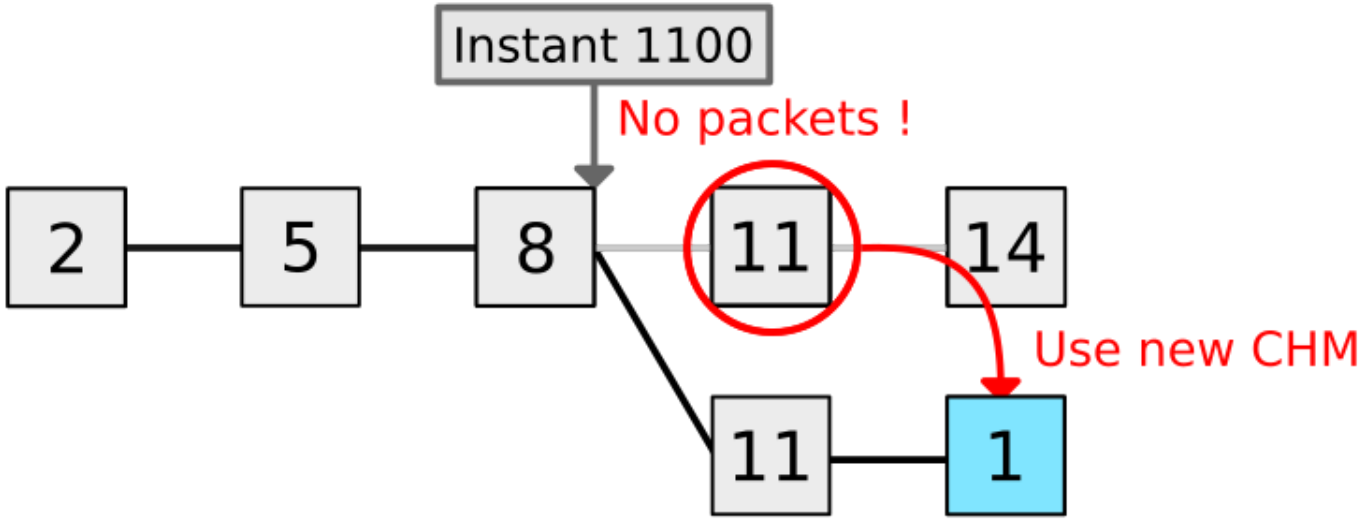


digital.security

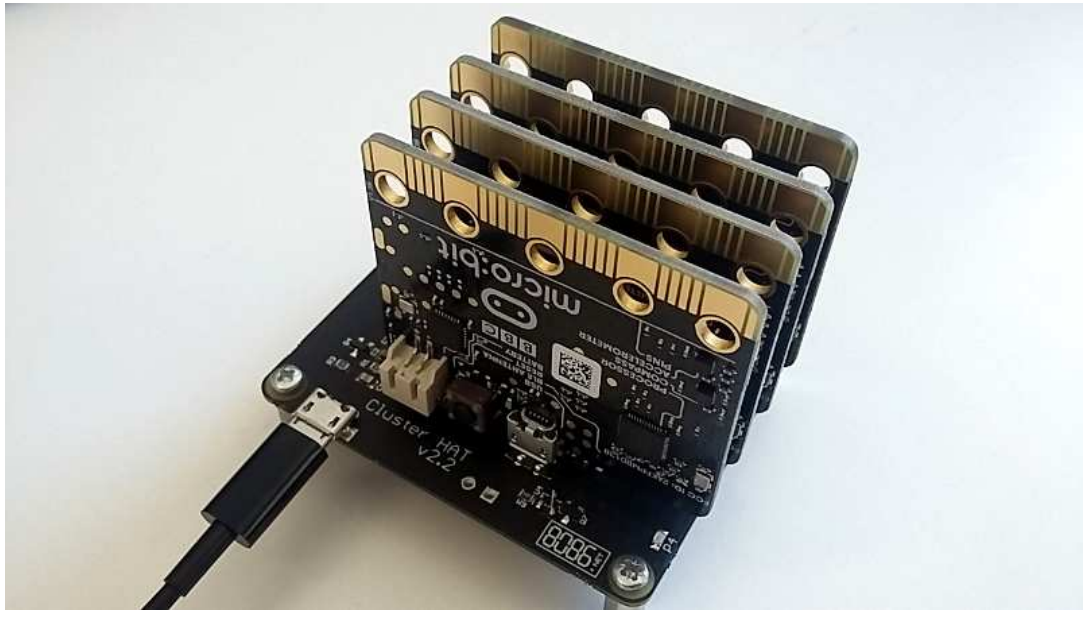
WE DON'T CARE AT ALL



WE DON'T CARE AT ALL



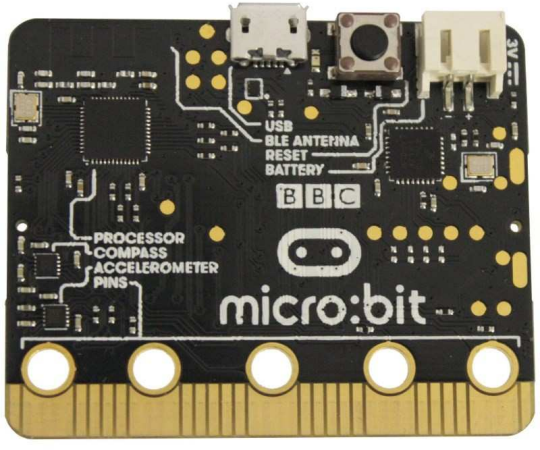
MULTIPLE SNIFFERS FOR THE ULTIMATE SNIFFING TOOL



digital.security

A BRAND NEW TOOL ...

... BASED ON A MICRO:BIT



$15

BTLEJUICE

digital.security

BTLE~~JUICE~~JACK

digital.security

NO LIVE DEMO, I KNOW YOU.

digital.security

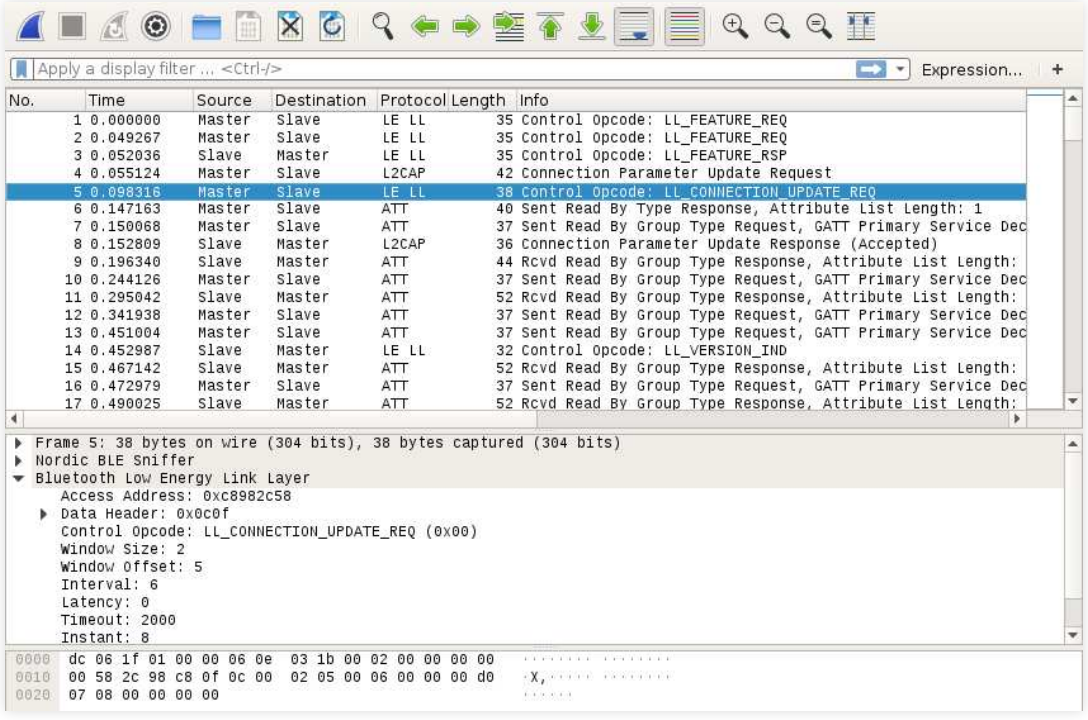
SNIFFING A NEW CONNECTION

```
virtualabs@virtubox:~/demo$
```

SNIFFING AN EXISTING CONNECTION

```
virtualabs@virtubox:~/demo$
```

PCAP EXPORT



Supports Nordic and legacy BTLE formats

digital.security

BTLEJACKING

A NEW ATTACK ON BLE

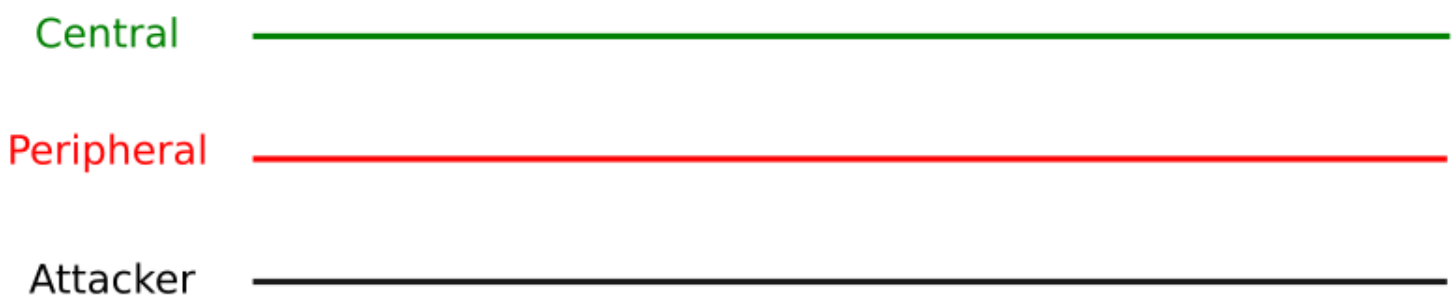
digital.security

SUPERVISION TIMEOUT

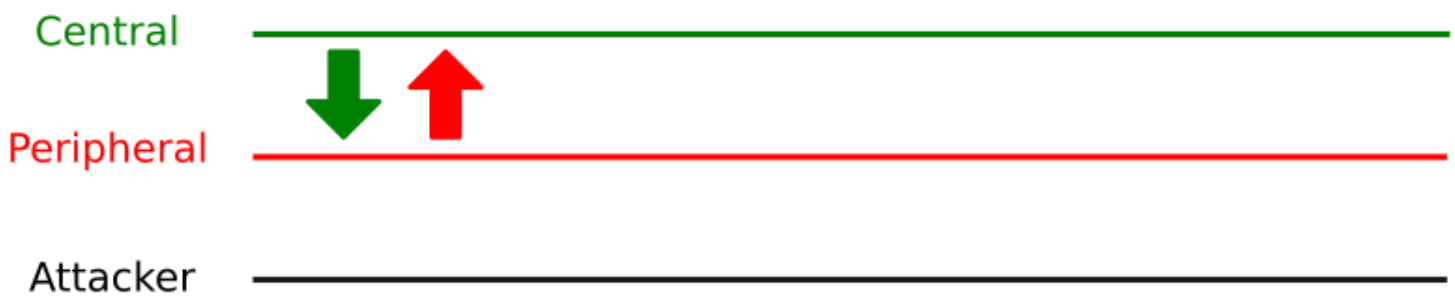
- Defined in CONNECT_REQ PDU
- Defines the time after which **a connection is considered lost** if no valid packets
- Enforced by both Central and Peripheral devices


OMG OMG OMG
WHATZ DAT TIMEOUT SH*T ?
imgflip.com

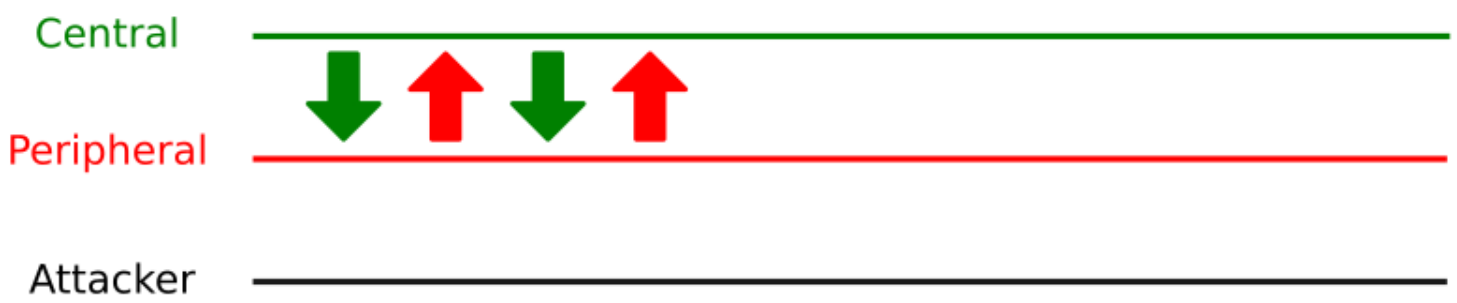
SUPERVISION TIMEOUT VS. JAMMING



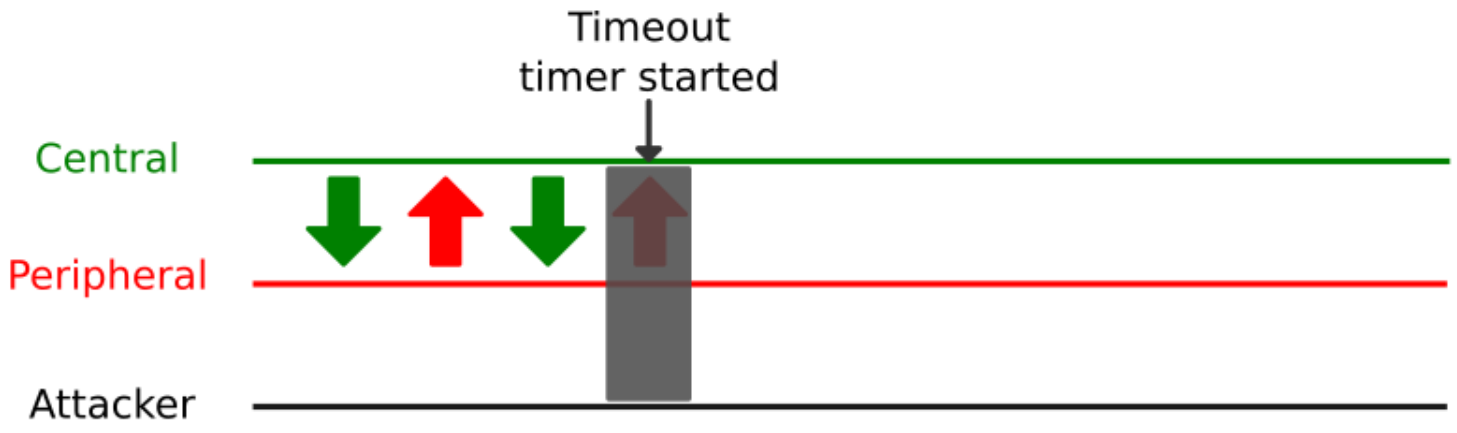
SUPERVISION TIMEOUT VS. JAMMING



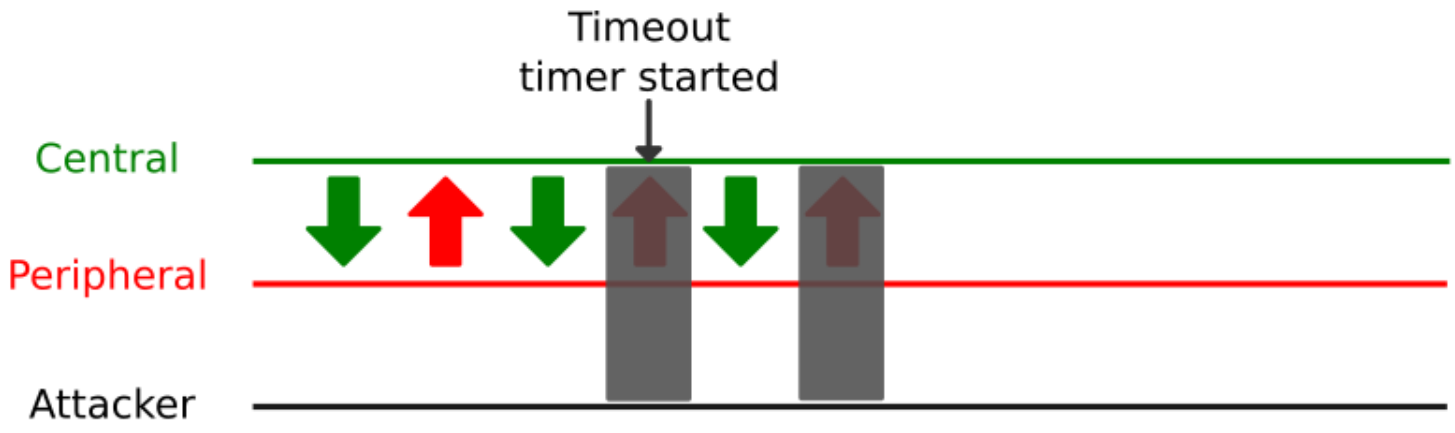
SUPERVISION TIMEOUT VS. JAMMING



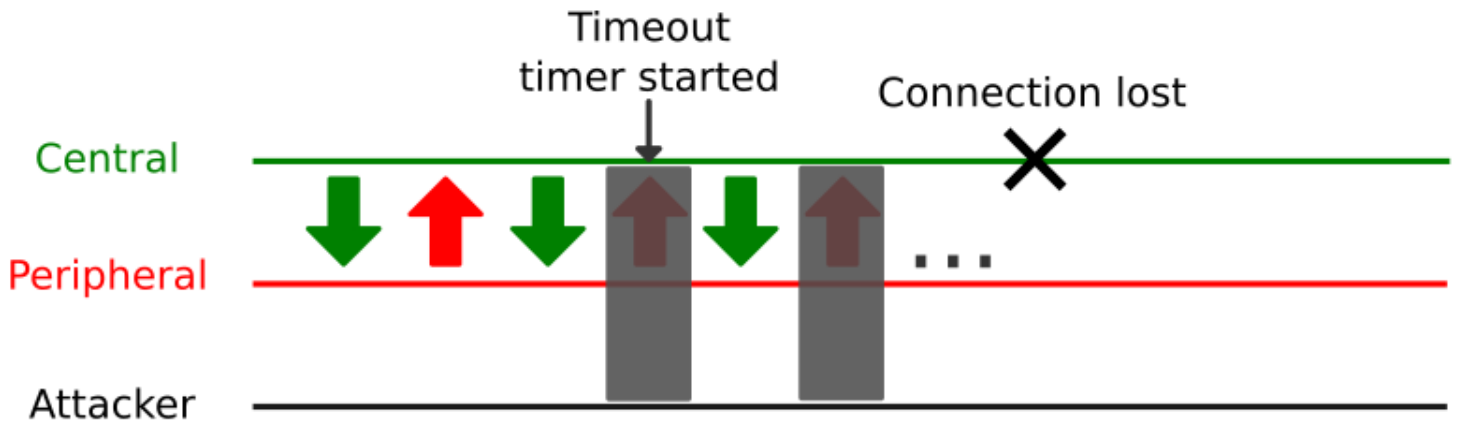
SUPERVISION TIMEOUT VS. JAMMING



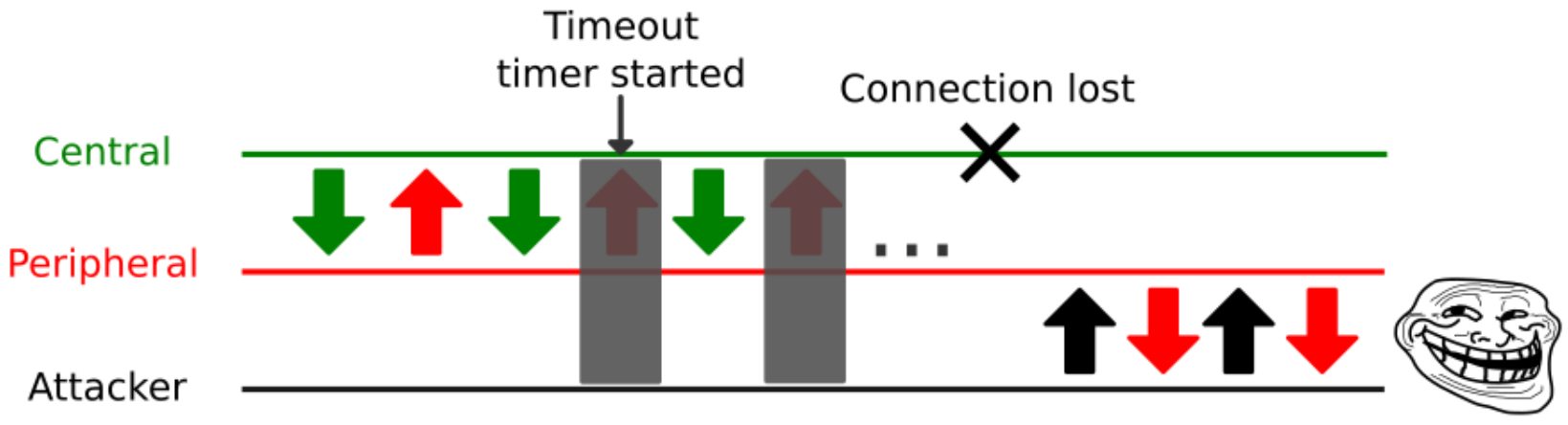
SUPERVISION TIMEOUT VS. JAMMING



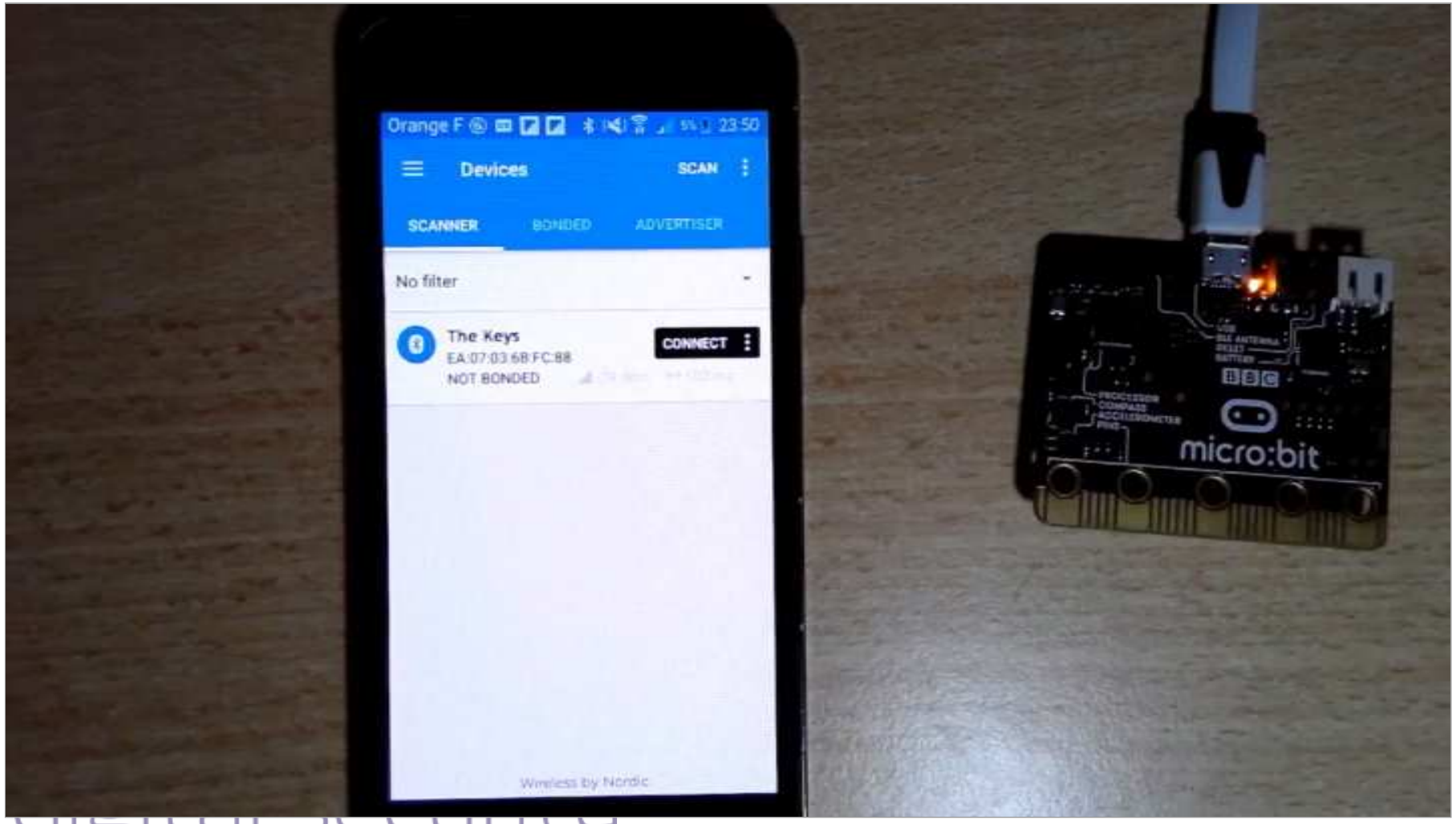
SUPERVISION TIMEOUT VS. JAMMING



SUPERVISION TIMEOUT VS. JAMMING



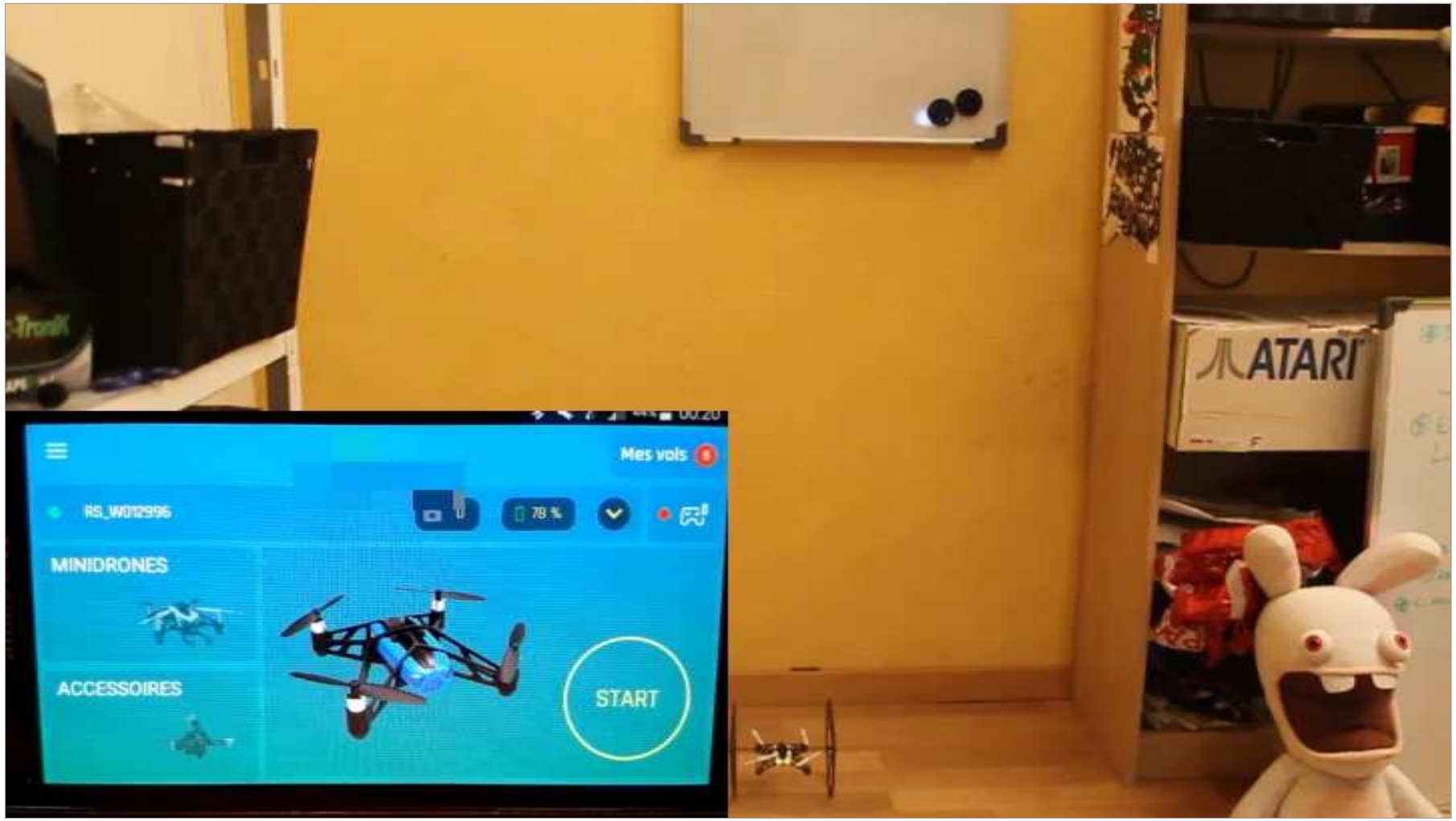
JAMMING FTW


Orange F
23:50
Devices
SCAN
SCANNER
BONDED
ADVERTISER
No filter
The Keys
EA:07:03:6B:FC:88
NOT BONDED
CONNECT
Wireless by Nordic
micro:bit

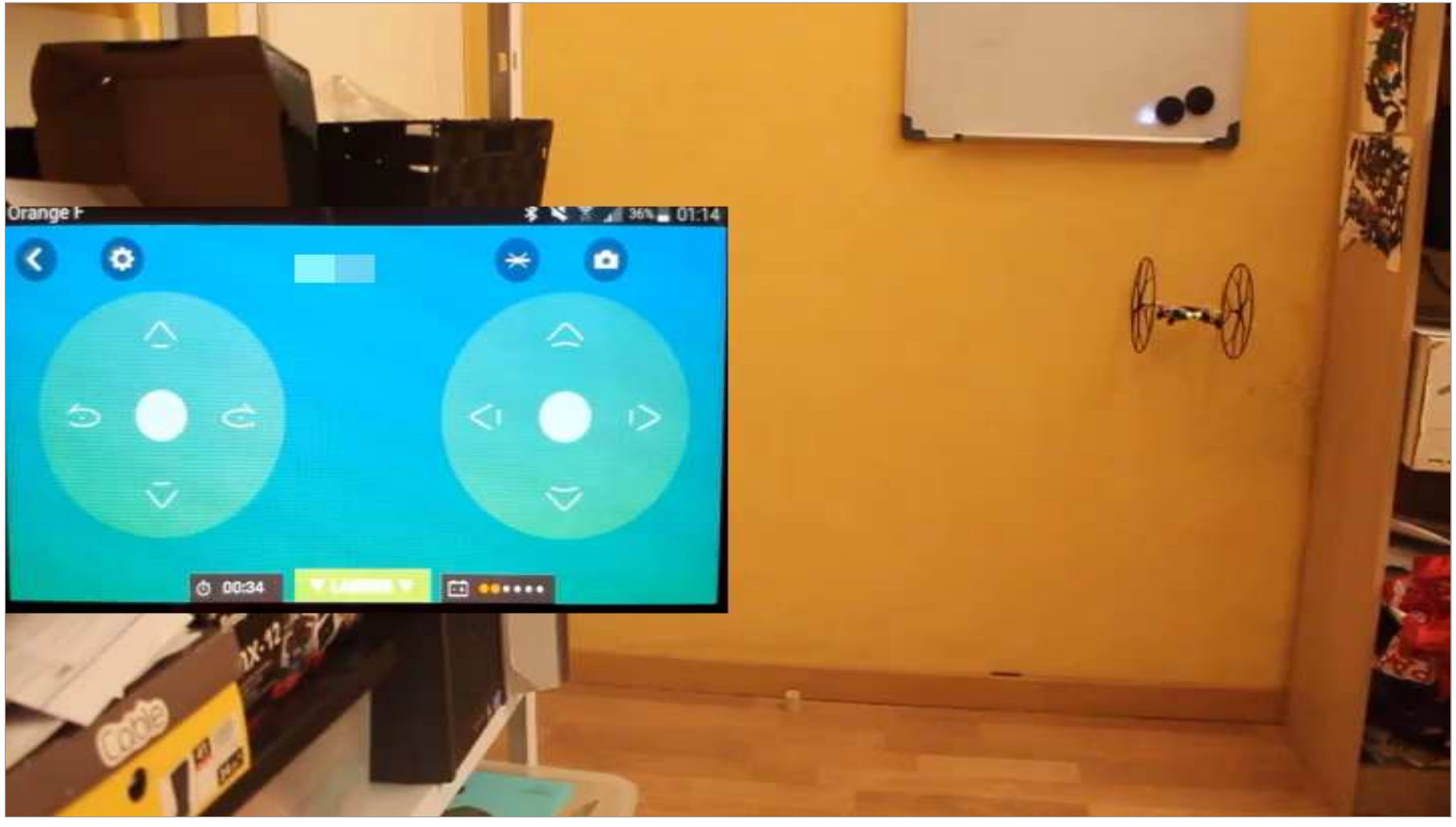
BTLEJACKING

- Abuse BLE supervision timeout to **take over a connection**
- BLE versions 4.0, 4.1, 4.2 and 5 are vulnerable
- **Requires proximity** (about 5 meters away from target)

EXAMPLE OF VULNERABLE DEVICES



digital.security



digital.security

SEXTOYS TOO !

PEN TEST PARTNERS
Penetration testing and security services

+44 20 3095 0500

So of course my colleagues suggested that my next project should be to reverse engineer it.

And here it is, the Lovense Hush:

SECURITY
Hacking Serial Networks on Ships
25 JUN 2018

REVERSE ENGINEERING
Hardware reverse engineering. A tale from the workbench
22 JUN 2018

SERVICES
Automotive and IoT Testing

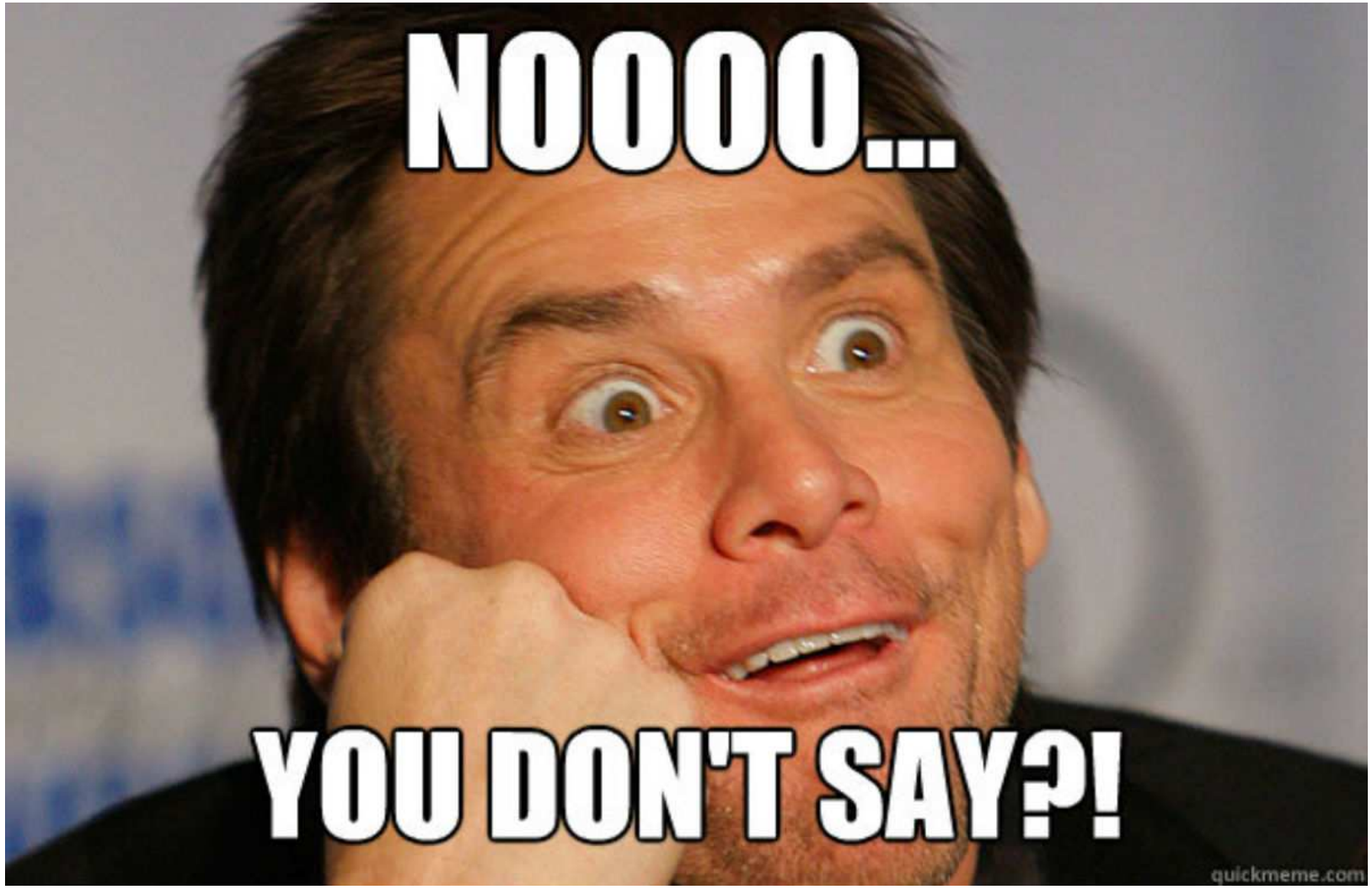
#2. IF THE TOY IS ON AND CONNECTED, YOU'RE FINE

Hackers would need to walk/drive around the city hoping someone has a teledildonic toy that is on **but NOT connected** to any phone.

It's rare to encounter this situation because if a user is wearing it out of the house it needs to be connected to the app in order to function, and that's the entire purpose of wearing it outside.

And if it's on and connected to your phone, the hacking can't happen because it **can only be controlled by one device at a time, aka the phone you're connected to.**

https://fr.lovense.com/sex-toy-blog/lovense-hack



digital.security

digital.security

IMPACT

- **Unauthorized access** to a device, even if it is already connected
- **Bypass authentication**, if authentication is performed at the start of connection
- Keep the device internal state intact: this may **leak valuable information**

COUNTER-MEASURES

- Use **BLE Secure Connections** (see specifications)
- At least authenticate data at application layer

BTLEJACK

https://github.com/virtualabs/btlejack

FEATURES

- Already established BLE connection sniffing
- New BLE connection sniffing
- Selective BLE jamming
- BLE connection take-over (btlejacking)
- PCAP export to view dumps in Wireshark
- Multiple sniffers support

CONCLUSION

- Btlejack is an **all-in-one** solution for BLE sniffing, jamming and hijacking
- BLE hijacking works on **all versions**
- Insecured BLE connections are prone to sniffing and hijacking
- It **might get worse** with further versions of BLE (greater range)
- **Secure your BLE connections FFS (really, do it)**

THANKS ! QUESTIONS ?

CONTACT

@virtualabs
damien.cauquil@digital.security

digital.security

WHY DIDN'T YOU IMPROVE *UBERTOOTH-BTLE* CODE ?

- I am **a lot more familiar** with nRF51 SoCs than LPC microcontrollers
- Buying 3 Ubertooth devices (**$360**) is not “cheap”

HOW DID YOU MAKE YOUR CLUSTER ?

From a modified **ClusterHat v2** ($30)

https://shop.pimoroni.com/products/cluster-hat