blueSPY Protocol Analyzer v24.09.09
USER GUIDE

A 2.4 GHz sniffer and protocol analyzer, with support for:

Bluetooth LE, BR/EDR, QBHSL, and mHDT;

HCI; 802.15.4; logic analysis; packet detection for WiFi;

Audio timing and analysis

Peter Ford

9 September 2024

1 CONTENTS

2 Overview

The RF Creations Protocol Analyzer allows capture and analysis of LE, BR/EDR, QBHSL, and mHDT Bluetooth traffic, 802.15.4 and logic analyser signals; detection of WiFi packets; and power measurements of other signals in the 2.4 – 2.5 GHz band ("Spectrum" data).

2.1 Getting started

If you are using blueSPY on Windows or macOS, all you need to is download the correct build (with on Windows a choice of an installer or a portable zip file containing the software), plug in your Moreph, and start capturing.

On Linux, if your system has no GUI libraries (e.g. a server) you should download the 'headless' build; otherwise download the full build including the GUI. To use blueSPY you will need to create a udev rule to communicate with the Moreph as an ordinary user. Create

```
/etc/udev/rules.d/50-minimoreph.rules
```

with the following contents:

```
SUBSYSTEM=="usb",    ATTRS{idVendor}=="2bbd",    ATTRS{idProduct}=="00f3",
MODE="0666"
```

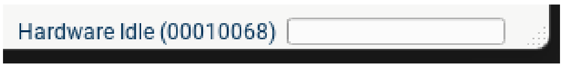
Reboot, or run "`udevadm control --reload-rules && udevadm trigger`" as root. Then you can use it as an ordinary user.

3 CAPTURING, SAVING AND LOADING FILES

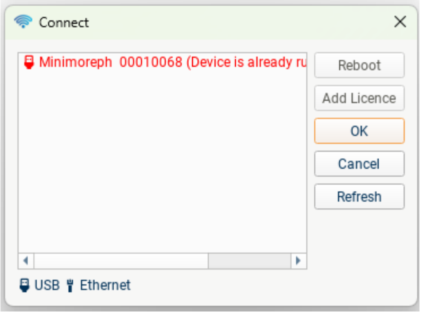
To capture traffic, you must have a Moreph 30 or a miniMoreph connected to your computer. When a Moreph is plugged in, or when the application is started, the bottom right of the status bar will display "Hardware Idle (#serial number)"; or "No Hardware Connected" if nothing is detected.

If the Moreph is not automatically detected, or you want to choose between multiple devices, use the "Connect" window to connect to a Moreph. If the Moreph is already running an application, you will need to Reboot it.

This window also allows you to add a licence to the Moreph, if you have been sent a new licence file for the device.

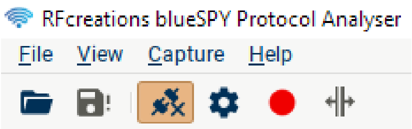


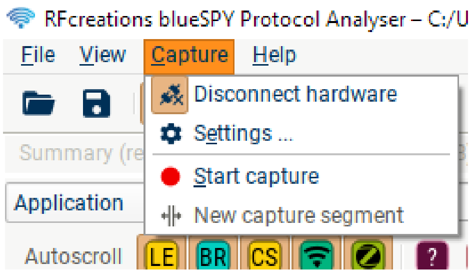
Hardware Idle (00010068)



Minimoreph 00010068 (Device is already ru

Once a Moreph is connected, the Capture Settings and Capture buttons will be enabled:





3.1.1 Connect

Open the Connect window, or disconnect the Moreph if it's already connected.

3.1.2 Capture

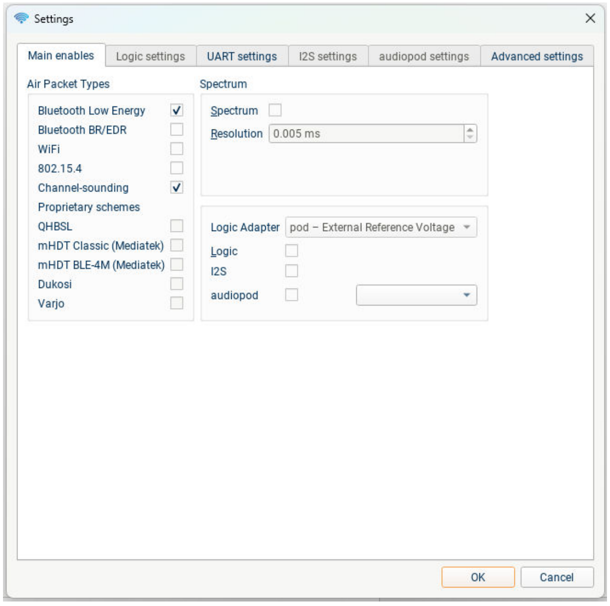
Starts and stops capture.

3.1.3 New Capture Segment

During a capture, seamlessly stops capturing and starts a new capture file; no packets are missed, so all traffic will be present in either the first or second capture. After the new capture has started a Save dialog will appear to allow you to choose a filename at your leisure. NB: Currently, connection timing is not preserved across the "new segment" operation, so only use this button at a time when none of your DUTs are connected.

3.1.4 Capture Settings

Use the Capture Settings window before starting a capture to select what data the Moreph should record. Each type of traffic can be enabled/disabled, and the time resolution of the Spectrum data can be selected; it is best to only capture the traffic you need, for the sake of capture file size if nothing else!



Air Packet Types

Bluetooth Low Energy ☑
Bluetooth BR/EDR □
WiFi □
802.15.4 □
Channel-sounding ☑
Proprietary schemes
QHBSL □
mHDT Classic (Mediatek) □
mHDT BLE-4M (Mediatek) □
Dukosi □
Varjo □

Spectrum

Spectrum □
Resolution 0.005 ms

Logic Adapter pod – External Reference Voltage
Logic □
I2S □
audiopod □

Not all combinations of packet types can be captured simultaneously due to hardware limitations; in addition, your licence may restrict which packets you can enable. If you have a licence which allows it, the following combinations of packets are possible:

1. QHS, Dukosi, Varjo, 802.15.4
2. Channel-sounding, 802.15.4
3. mHDT Classic, mHDT BLE-4M

Logic, Spectrum, WiFi, LE, and BR/EDR are supported in any of the combinations.

If that seems complicated, an alternative strategy: if a checkbox for a proprietary/unusual packet type is disabled, trying unchecking all other checkboxes and enabling it, and then see which other checkboxes are still enabled.

When using the logic analyser, it is important to specify the source of the reference voltage. If you are connecting an external reference voltage to the VREF pin (voltages between 0.8V and 3.6V are supported), select "pod – External Reference Voltage". Otherwise, select "pod – Internal Reference Voltage"; the Moreph will supply 3.3V on VREF.

Although very fine resolution (5 µs) Spectrum capture is possible, using it leads to very large file sizes (many GB) and so it is best to disable Spectrum capture or set a longer time resolution if fine timing is not needed.

3.1.4.1 Logic, UART, I2S settings

To access the Logic or I2S settings tabs, the Logic or I2S checkboxes in the "Main enables" tab must be checked.

In the logic tab you can choose which logic lines to enable (some will be unavailable if they are being used for UART or I2S), and the "Logic Rate". Selecting a lower Logic Rate will result in a smaller capture file size, but may result in some edges being missed if capturing a high-speed signal (e.g. a >1 MHz clock).

In the UART tab you can enable upto four UARTs, and configure the baud rate and other settings. The name field will be used to label the UART traffic in the Timeline, Summary etc. One or more UARTs can be used to capture HCI traffic; to capture both Controller->Host and Host->Controller traffic, a second UART will be used.

There are many possible configurations of I2S and I2S-like digital audio data. To successfully capture I2S traffic, ensure that:

a) the reference voltage has been configured appropriately in the "Main enables" tab, and that you are supplying the external VREF if needed;
b) the configuration options in the "I2S settings" match your implementation. The diagram at the top of the tab provides a pictorial explanation of the effects of the various checkboxes.

3.1.4.2 Audiopod settings

To enable the audiopod settings tab, connect the audiopod to your Moreph, and provide power to the audiopod USB-C port. Then check the "audiopod" checkbox in the "Main enables" tab; if the Moreph cannot communicate with the audiopod, an error will be returned at this stage.

For more details of the audiopod settings, see the Audiopod section

3.1.4.3 Advanced settings

These settings can be accessed during an ongoing capture, and allow extra control over the sniffer's radio, and filtering of packets captured.

The sniffer Automatic Gain Control normally performs well, so in most scenarios we don't recommend that you override the AGC; there are two main reasons to use this control:

1) Capturing traffic over a cabled link, i.e. at high powers (> 0 dBm). In this case the first packet on a link may be missed as the AGC adjusts, and the radio will perform better if you disable the AGC and specify the expected maximum power.
2) Capturing wanted distant/quiet traffic in the presence of louder interferers. Normally the AGC will adjust to ensure good reception of the loudest signals present, but if you want to force maximum sensitivity (at the risk of distortion/missed packets from the loudest devices), you can disable the AGC and set "Maximum Input Signal" to the minimum value.

3.2 Saving and Loading files

"Open" and "Save As" operate in the usual way, and are found in the File menu. A new capture always writes to a temporary file; afterwards the file can either be saved or discarded.

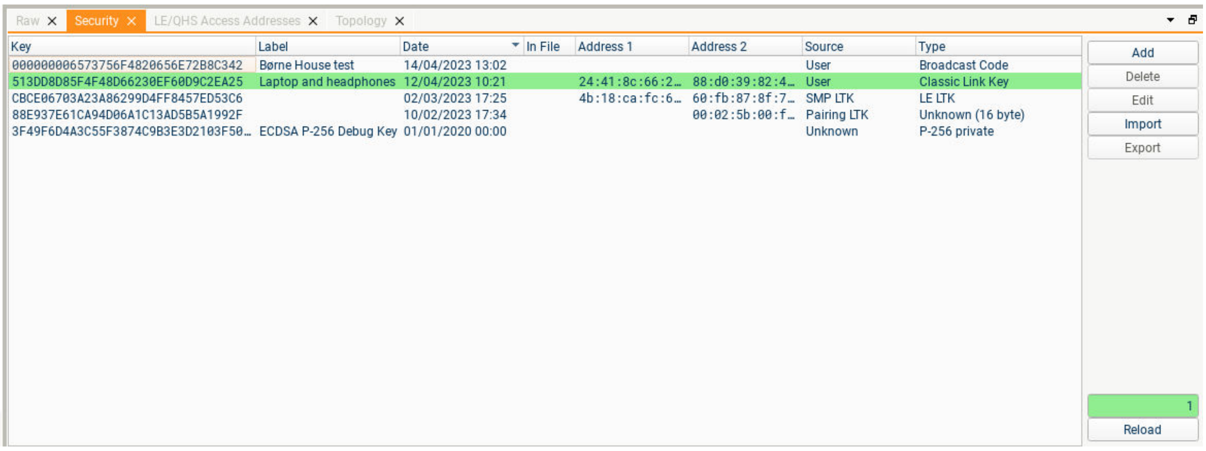
"Save Advanced" can be used to save a portion of a large file, for instance by omitting Spectrum data, selecting a shortened time-period, or saving only some of the packets. A Moreph must be connected to "license" the file. NB: Saving a subset of the packets may result in a file in which the packets can no longer be parsed, e.g. if the beginning of a Connection is missed. To minimise filesize: disable all protocol filters in the Summary; enable the device filter to show only devices you need; select the timerange you need in Save Advanced, and use "Save Packets: Shown in Summary" (with "Save Spectrum" not checked).

If you have added information to a file (e.g. adding decryption keys, adding Bookmarks, modifying device filtering) then the "Save" icon in the toolbar will change to show that you may want to save your changes:

💾 vs 💾!

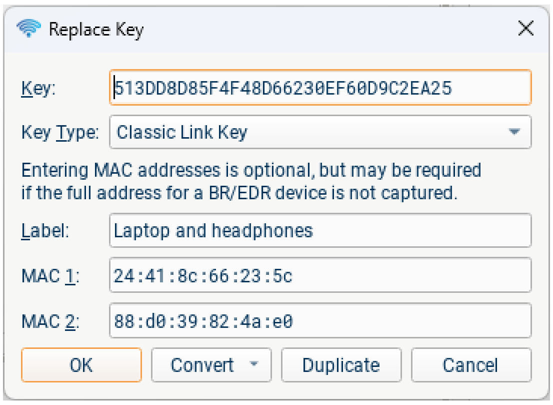
4 Decrypting

Most Bluetooth communication is encrypted, and a key is required to decrypt it. In blueSPY this is handled through the Security tab, which stores a persistent database of keys.



Key	Label	Date	In File	Address 1	Address 2	Source	Type
000000006573756F4820656E72B8C342	Børne House test	14/04/2023 13:02				User	Broadcast Code
513DD8D85F4F48D66230EF60D9C2EA25	Laptop and headphones	12/04/2023 10:21		24:41:8c:66:2…	88:d0:39:82:4…	User	Classic Link Key
CBCE06703A23A86299D4FF8457ED53C6		02/03/2023 17:25		4b:18:ca:fc:6…	60:fb:87:8f:7…	SMP LTK	LE LTK
88E937E61CA94D06A1C13AD5B5A1992F		10/02/2023 17:34			00:02:5b:00:f…	Pairing LTK	Unknown (16 byte)
3F49F6D4A3C55F3874C9B3E3D2103F50…	ECDSA P-256 Debug Key	01/01/2020 00:00				Unknown	P-256 private

The supported key types are BR/EDR link keys, LE LTKs, ECDSA pairing keys, Broadcast Codes or Broadcast GLTKs and Encryted Advertising keys. You can add or import keys, and delete, edit or export the currently selected keys using the buttons on the right. Adding or editing a key will show the following dialog:



Key: 513DD8D85F4F48D66230EF60D9C2EA25

Key Type: Classic Link Key

Entering MAC addresses is optional, but may be required
if the full address for a BR/EDR device is not captured.

Label: Laptop and headphones

MAC 1: 24:41:8c:66:23:5c

MAC 2: 88:d0:39:82:4a:e0

You only need to enter the key, the key type can be automatically detected and in most cases the MAC addresses can too. The label is also optional, but helps keep track of the keys. The 'Convert' menu allows you to easily perform some byte reordering operations on keys. When editing a key, you can press Duplicate to add a new similar key instead of replacing the old one.

After adding a key, press "Reload" to re-parse the current file using the new key. If a key is successfully used to decrypt some traffic, it will be highlighted in green in the Security table, and the number above the reload button will show the number of successful keys. Any successful keys will be stored in the file when the capture is saved.

Since there is often confusion about the endianness of keys, the software will try both the key as entered and the byte reversed version, and correct the key if it was entered reversed.

If a key is seen in an HCI, ATT or SMP message it will automatically be added and used.

Devices using LE Legacy Pairing are insecure, and the packets are decrypted without the need to add any keys as long as the pairing has been captured.

4.1 Link Keys

Link keys for BR/EDR or LE are entered as 16 hexadecimal bytes (32 characters), byte 0 first. The fields for adding MAC addresses do not need to be filled in usually, however if either of the full addresses for a BR/EDR device were not found during a capture you will need to add them manually. The addresses will be automatically filled in once a successful decryption has occurred.

4.2 ECDSA Pairing Keys

These are entered as a 24 or 32 byte (for P-192 and P-256 respectively) hexadecimal big endian integer. The corresponding public key is shown when the full key is entered and the appropriate type is selected so you can verify it has been entered correctly.

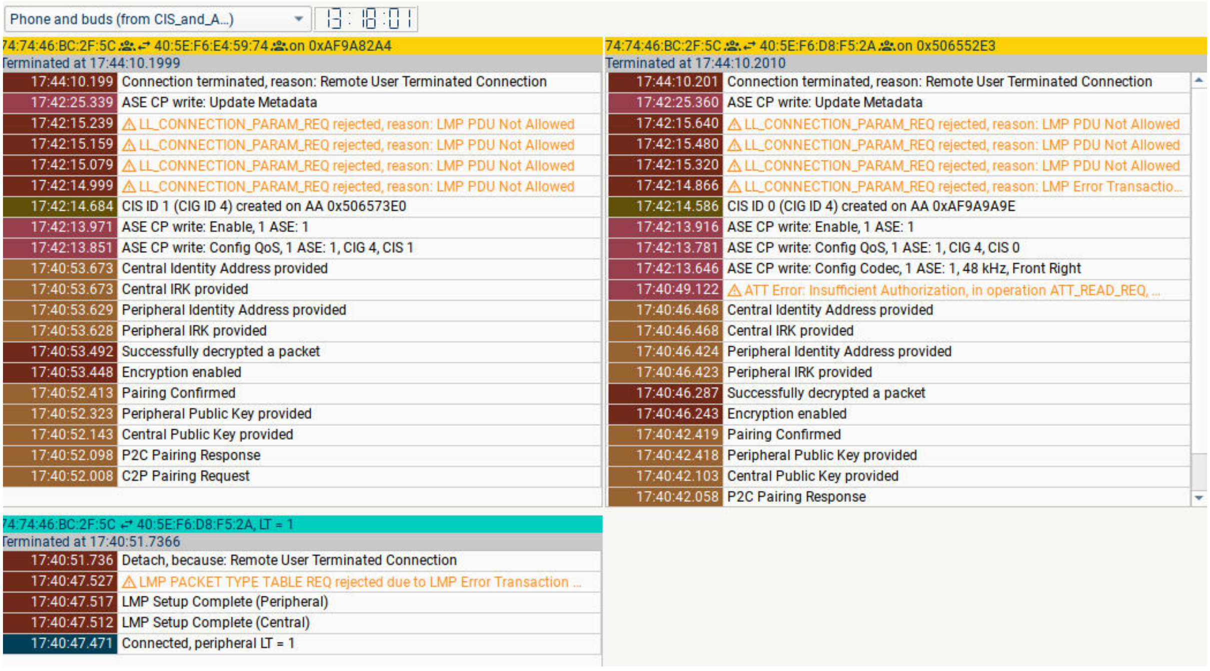
4.3 Broadcast Codes

You can enter broadcast codes as a 16 hexadecimal bytes like the other keys, or you can enter the code string directly if you select the ‘Broadcast Code’ key type. The first character of the code is the last byte of the hexadecimal version, as defined in the Bluetooth specification. You can convert between the code string and hexadecimal form using the convert menu.

4.4 Encrypted Advertising

Encrypted Advertising keys are entered as 24 hexadecimal bytes, a 16 byte key followed by an 8 byte IV.

5 DASHBOARD



Phone and buds (from CIS_and_A...) 13:18:01

```
74:74:46:BC:2F:5C ⇄ 40:5E:F6:E4:59:74 on 0xAF9A82A4
Terminated at 17:44:10.1999
17:44:10.199 Connection terminated, reason: Remote User Terminated Connection
17:42:25.339 ASE CP write: Update Metadata
17:42:15.239 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP PDU Not Allowed
17:42:15.159 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP PDU Not Allowed
17:42:15.079 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP PDU Not Allowed
17:42:14.999 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP PDU Not Allowed
17:42:14.684 CIS ID 1 (CIG ID 4) created on AA 0x506573E0
17:42:13.971 ASE CP write: Enable, 1 ASE: 1
17:42:13.851 ASE CP write: Config QoS, 1 ASE: 1, CIG 4, CIS 1
17:40:53.673 Central Identity Address provided
17:40:53.673 Central IRK provided
17:40:53.629 Peripheral Identity Address provided
17:40:53.628 Peripheral IRK provided
17:40:53.492 Successfully decrypted a packet
17:40:53.448 Encryption enabled
17:40:52.413 Pairing Confirmed
17:40:52.323 Peripheral Public Key provided
17:40:52.143 Central Public Key provided
17:40:52.098 P2C Pairing Response
17:40:52.008 C2P Pairing Request
```

```
74:74:46:BC:2F:5C ⇄ 40:5E:F6:D8:F5:2A on 0x506552E3
Terminated at 17:44:10.2010
17:44:10.201 Connection terminated, reason: Remote User Terminated Connection
17:42:25.360 ASE CP write: Update Metadata
17:42:15.640 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP PDU Not Allowed
17:42:15.480 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP PDU Not Allowed
17:42:15.320 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP PDU Not Allowed
17:42:14.866 ⚠ LL_CONNECTION_PARAM_REQ rejected, reason: LMP Error Transactio...
17:42:14.586 CIS ID 0 (CIG ID 4) created on AA 0xAF9A9A9E
17:42:13.916 ASE CP write: Enable, 1 ASE: 1
17:42:13.781 ASE CP write: Config QoS, 1 ASE: 1, CIG 4, CIS 0
17:42:13.646 ASE CP write: Config Codec, 1 ASE: 1, 48 kHz, Front Right
17:40:49.122 ⚠ ATT Error: Insufficient Authorization, in operation ATT_READ_REQ, ...
17:40:46.468 Central Identity Address provided
17:40:46.468 Central IRK provided
17:40:46.424 Peripheral Identity Address provided
17:40:46.423 Peripheral IRK provided
17:40:46.287 Successfully decrypted a packet
17:40:46.243 Encryption enabled
17:40:42.419 Pairing Confirmed
17:40:42.418 Peripheral Public Key provided
17:40:42.103 Central Public Key provided
17:40:42.058 P2C Pairing Response
```

```
74:74:46:BC:2F:5C ⇄ 40:5E:F6:D8:F5:2A, LT = 1
Terminated at 17:40:51.7366
17:40:51.736 Detach, because: Remote User Terminated Connection
17:40:47.527 ⚠ LMP PACKET TYPE TABLE REQ rejected due to LMP Error Transaction ...
17:40:47.517 LMP Setup Complete (Peripheral)
17:40:47.512 LMP Setup Complete (Central)
17:40:47.471 Connected, peripheral LT = 1
```

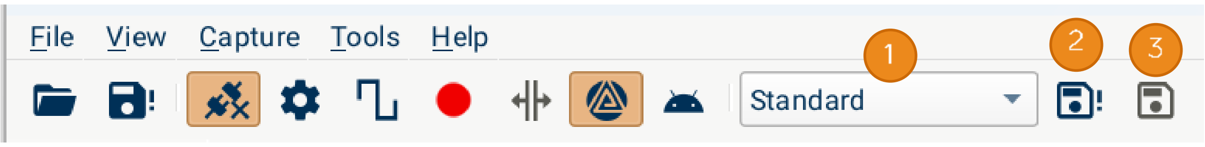
The Dashboard tab provides a brief summary of the connections selected in a device filter; a list of notable events (encryption start, audiostream creation, connection termination etc), and any warnings or errors seen. Some warnings aren't really a problem (e.g. the LMP Transaction Collision shown above), but if you are trying to answer the question "what went wrong with the Bluetooth connection/stream" these messages are likely to be a good place to start; you can double-click any of them to jump to that point in the Summary/Timeline/etc to investigate further.

We look out for known bugs and highlight them in the Dashboard, so if you would like to see some particular bug or type of message displayed, let us know.

6 Changing Layouts

In BlueSPY, the arrangement of tabs along with the configurations of each tab is called a Layout. Layouts can be saved to and loaded from BlueSPY at the user's convenience, and there is a selection of default layouts within blueSPY to choose from.

Each tab has information about it stored in a layout such as which buttons are pressed and their geometric dimensions. For example, in a Summary tab, the columns used, the filtering options, and any search queries are stored.



1 Drop menu to select different layouts.

2 Saves the current layout as a new layout.

3 Overwrites the selected layout with the current layout; only possible for user-defined layouts.

Two user-defined layouts cannot share the same name, and a user-defined layout cannot share a name with a default layout. BlueSPY does not load the layout it had when it was last closed. BlueSPY will load the most recently selected layout before it last closed. The user must remember to save their layout regularly if they want it to persist after closing the application.

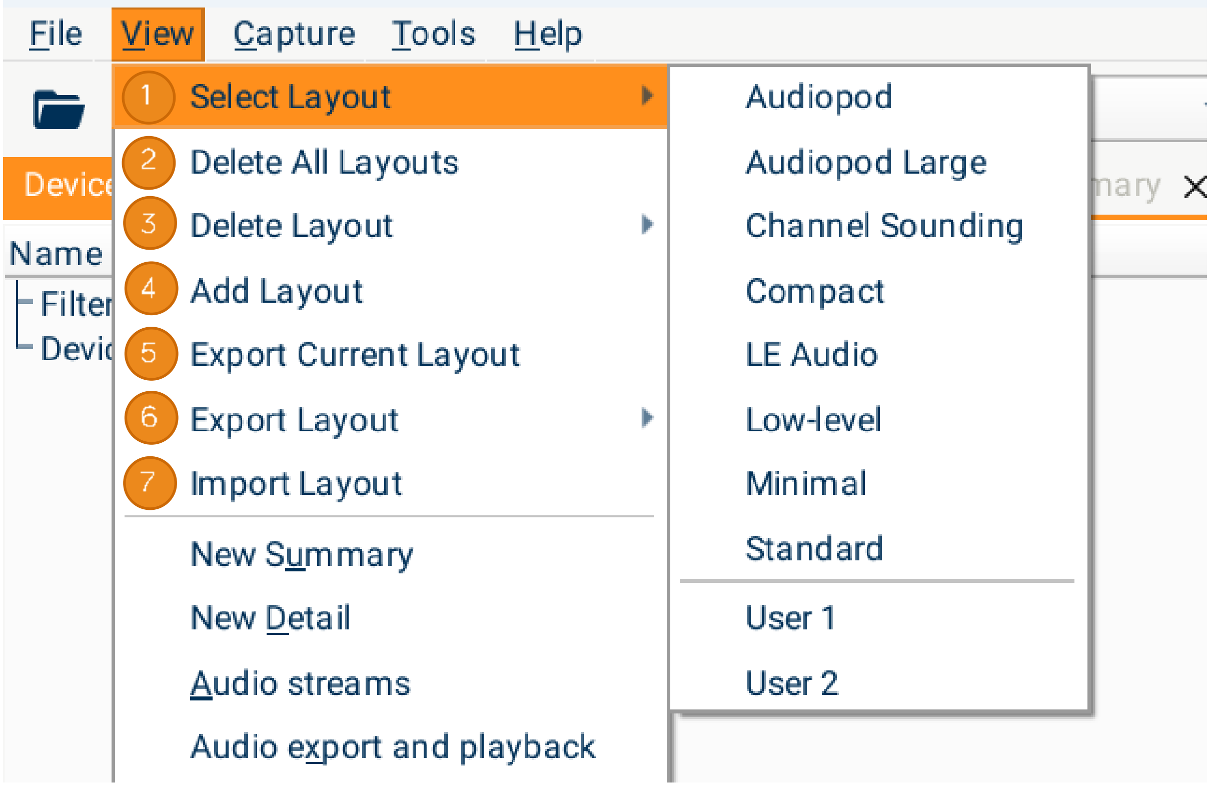
A layout can be exported to a file, e.g. for transferring to another computer.

6.1 Default Layouts

BlueSPY contains eight default layouts:

1. Standard – A general-use layout for larger screens.
2. Compact – A general-use layout for medium-sized screens.
3. Minimal – A general-use layout for small screens.
4. Audiopod – The Audio and Audiopod tabs raised.
5. Audiopod Large – The Audio, Audiopod, and Audio Export are raised.
6. Channel Sounding – Channel Sounding options are selected, and two Details tabs are present to make it easy to compare different CS procedures/events/steps.
7. LE Audio – Two Summary tabs are configured, one showing configuration traffic and one showing audio packets.
8. Low Level – Focuses on baseband packets and their timing.

6.2 VIEW MENU



1 Select a default or user-defined layout. Same function as the dropbox in the toolbar.

2 Deletes all user-defined layouts.

3 Deletes a single user-defined layout.

4 Saves the current layout.

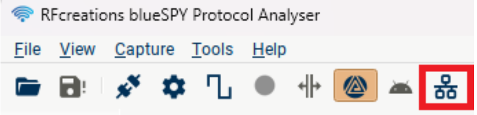
5 Exports the current layout to a file.

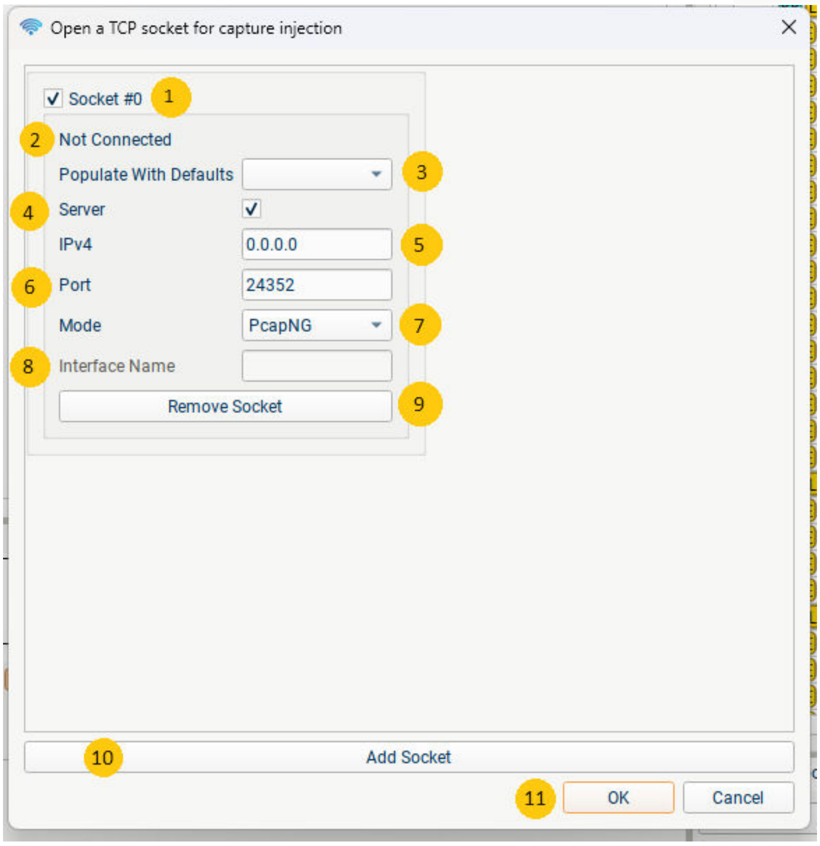
6 Exports a default or user-defined layout to a file.

7 Imports a layout from a file.

7 INJECTION INTERFACE

TCP sockets can be opened in blueSPY, allowing you to inject packets into an ongoing capture. blueSPY can act as either server or client, but currently there is a limit of one connection per socket. The dialog to configure sockets can be opened from the blueSPY toolbar or the Capture menu.




Open a TCP socket for capture injection

☑ Socket #0

Not Connected

Populate With Defaults:

Server: ☑

IPv4: 0.0.0.0

Port: 24352

Mode: PcapNG

Interface Name:

1. Checkbox for disconnecting the socket if connected. Only sockets with this checkbox ticked will be set up upon confirmation.
2. Connection status label
3. Shortcuts for setting fields to certain preset values
   - BTVS
     - Client
     - IPv4: 127.0.0.1
     - port: 24352
     - Pcap mode
   - Local Host - IPv4: 127.0.0.1
   - Remote Host - IPv4: 0.0.0.0
4. Server checkbox. If ticked the socket will be a TCP/IP server otherwise a TCP/IP client will be set up.
5. IPv4 address field. Specifies which IPv4 address to connect to or listen for connection attempts from in client and server modes respectively. 0.0.0.0 in server mode is interpreted as any IP address. Leaving the field blank has the same effect as setting it to 0.0.0.0.
6. TCP port field
7. Operating modes. Specifies how the socket should interpret incoming data.

8. Interface name allows the user to set a description for the PcapNG interface that will be created. It is used in blueSPY for labelling the HCI "device" in the Summary and Timeline. If left blank a default name using TCP/IP name will be used. Since the PcapNG mode requires the Interface Description blocks to be sent over the socket, this field is disabled in that mode.
9. Remove this socket
10. Add a new socket
11. OK - confirm changes, Cancel - discard changes

After closing the dialog, blueSPY starts listening for connections or making connection attempts, depending on the selected mode. However, packets are ignored until a capture has been started. In the server mode, blueSPY listens until a successful connection is established. Upon client disconnection, the socket starts listening for a new connection. In the client mode connection attempts to the specified server are being made until it succeeds or times out. Information about the status, and debug messages if invalid bytes are received, are printed in the log. If the timeout occurs or the connection has been lost, the socket is closed and needs to be set up again.

The toolbar button is highlighted to indicate that at least one socket is currently connected to a peer. The dialog can be opened again to view the status of each socket. Sockets can be removed or added at any point; however any data with the exception of the Interface Description PcapNG blocks will be discarded if blueSPY is not actively capturing. Changing socket settings is disabled once a connection has been established. Changing the interface name of a connected socket is allowed only prior to starting a blueSPY capture.

7.1.1 Raw HCI H4

In this mode blueSPY accepts raw bytes and tries to interpret them in accordance with the HCI H4 format.

7.1.2 Pcap

Follows the specification at: https://www.ietf.org/archive/id/draft-ietf-opsawg-pcap-03.html
A file header is required in this mode as the packet records themselves do not contain all the information needed for successful decoding. Further restrictions:

- little endian format
- LINKTYPE_BLUETOOTH_HCI_H4_WITH_PHDR (0xC9) link type

The socket is closed upon reception of an invalid header.

7.1.3 BTSnoop

A file header is required in this mode as the packet records themselves do not contain all the information needed for successful decoding. Further restrictions:

- format version 1
- Un-encapsulated HCI (H1) (0x3E9) or HCI UART (H4) (0x3EA) link types

The socket is closed upon reception of an invalid header.

7.1.4 PcapNG

Follows the specification at: https://www.ietf.org/archive/id/draft-ietf-opsawg-pcapng-01.html
All blocks are required to be in the little endian format.
blueSPY accepts the following standard block types:

- section header
- interface description
- enhanced packet

The following custom blocks are defined and accepted.

7.1.4.1 Link Key

```
                         1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 0 |                     block type = 0x80000200                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4 |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 8 |                              key                              |
12 |                                                               |
16 |                                                               |
20 |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
24 |                             addr0                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
28 |            addr0              |              addr1            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
32 |                             addr1                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
36 |  label length |               label (variable length)         |
   +-+-+-+-+-+-+-+-+                                               /
   /                                                               /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

7.1.4.2 *Encrypted Advertising key*

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 0 |                     block type = 0x80000206                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4 |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 8 |                              key                              |
12 |                                                               |
16 |                                                               |
20 |                                                               |
24 |                                                               |
28 |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
36 |  label length |               label (variable length)         |
   +-+-+-+-+-+-+-+-+                                               /
   /                                                               /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

7.1.4.3 Broadcast Code

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 0 |                     block type = 0x80000207                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4 |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 8 |                              key                              |
12 |                                                               |
16 |                                                               |
20 |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
36 |  label length |               label (variable length)         |
   +-+-+-+-+-+-+-+-+                                               /
   /                                                               /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

7.1.4.4 *Log message*

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 0 |                     block type = 0x80000109                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4 |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 8 |                         timestamp MSB                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
12 |                         timestamp LSB                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
20 |                         message length                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
36 |s_subtyp |s_typ|             message (variable length)         |
   +-+-+-+-+-+-+-+-+                                               /
   /                                                               /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            length                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

`timestamp` is a UNIX Timestamp in nanoseconds.

`s_subtyp` is severity subtype; this can be set to any value, to distinguish user-defined error types.

`s_typ` is severity type; this is used to colour-code the log messages in the Summary.

defined severity types:

- Pass = 0x0,
- Warning = 0x1,
- Info = 0x2,
- Debug = 0x3,
- Error = 0x4,

8 AUDIOPOD

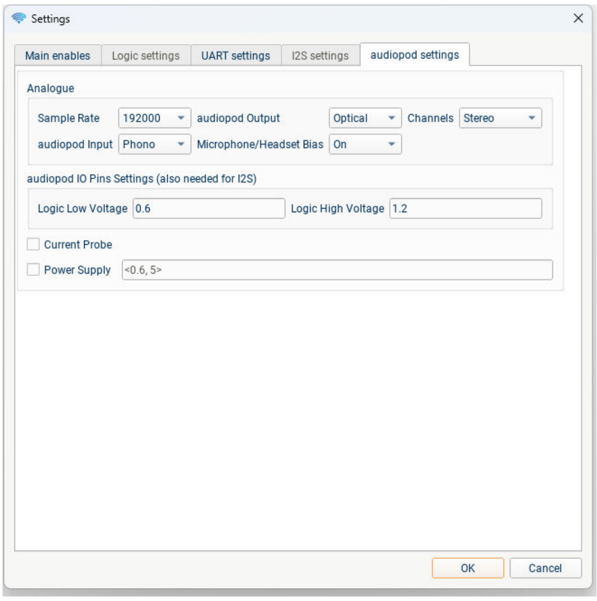
Using the audiopod accessory, you can record and play back analogue or digital audio (or use the provided microphones), timestamped using the same clock as the sniffer uses to timestamp the Bluetooth packets on-air. This is particularly useful for measuring conformance with the Presentation Delay requirements of LE Audio, and can also be used to measuring timing, quality, and the effects of missed or retransmitted packets on all types of Bluetooth audio streams.

8.1 CAPTURING AND PLAYING AUDIO

To capture traffic using audiopod: connect the audiopod to the Moreph, use the USB-C port to provide power, and connect audio cables to whichever of the ports you want to use.

To configure the audiopod, in the "Main Enables" tab select whether you will be using Analogue audio, Digital audio, or some combination.

Then in the "audiopod settings" tab you can configure the inputs and outputs to audiopod; either logic levels for input and output on the logic pins, or one or more of the audio connectors (Phono, 3.5 mm Jack, Coax S/PDIF, Optical S/PDIF, or the combined input+output "Headset" connector). When using the external earcanal microphones, or other microphones which require a bias voltage, make sure to enable the bias.



Analogue

Sample Rate: 192000
audiopod Output: Optical
Channels: Stereo
audiopod Input: Phono
Microphone/Headset Bias: On

audiopod IO Pins Settings (also needed for I2S)

Logic Low Voltage: 0.6
Logic High Voltage: 1.2

□ Current Probe
□ Power Supply: <0.6, 5>

Settings (other than volume controls, and enables for the AGC and DRC on input/output) must be selected before starting the capture.

For audio output, you can either provide a file for blueSPY to play (.wav or .mp3), or use the simple tone/chirp generating options in the audiopod tab. NB: A constant tone or other short-period periodic signal will not work for measuring audio latency/synchronisation! White noise is a good choice.

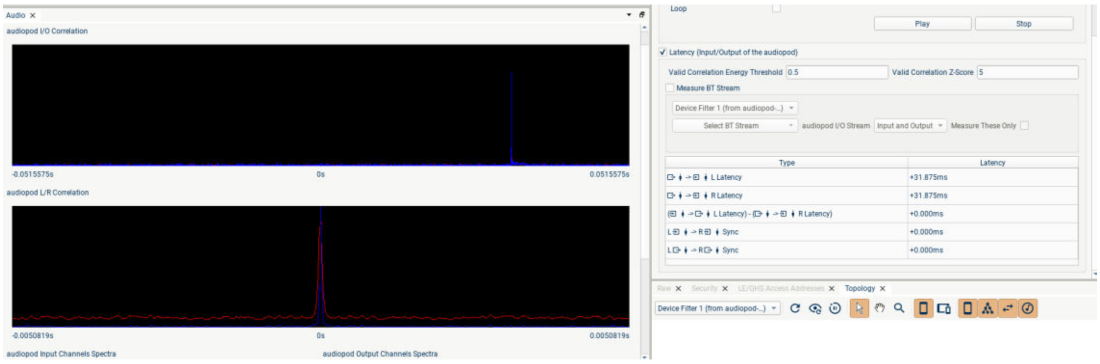
8.2 ANALYSING LATENCY

Audiopod introduces two new tabs to blueSPY, the "audiopod" control tab and the "Audio" tab displaying graphs of correlation and audio spectrum. In the top half of the "audiopod" tab are the volume/gain controls, and controls over the signal generation or file choice for the audio output. In the bottom half of the tab, you can choose which Bluetooth audio streams you want to correlate with the analogue/digital audio inputs and outputs. Below these controls, you will see a table containing either latency/synchronisation measurements, or "N/A" if no correlation was found between the signals you are comparing.

To trigger a measurement, choose a time to measure at (the calculation uses one second of audio centred at this point) by clicking on a packet, e.g. in the Timeline. While capturing you can also use "Autoscroll" in the Timeline; this will cause the calculations to continuously update using the most recent second of audio.

The "Audio" tab shows two types of graph; how many graphs are shown depends on what signals you have chosen to measure.

The first type of graph shows the output of a cross-correlation of two of the audio signals. If the two audio streams contain the same signal but one is delayed, you should see a sharp single peak whose x-axis position represents the time-delay:



☑ Latency (Input/Output of the audiopod)

Valid Correlation Energy Threshold 0.5 Valid Correlation Z-Score 5

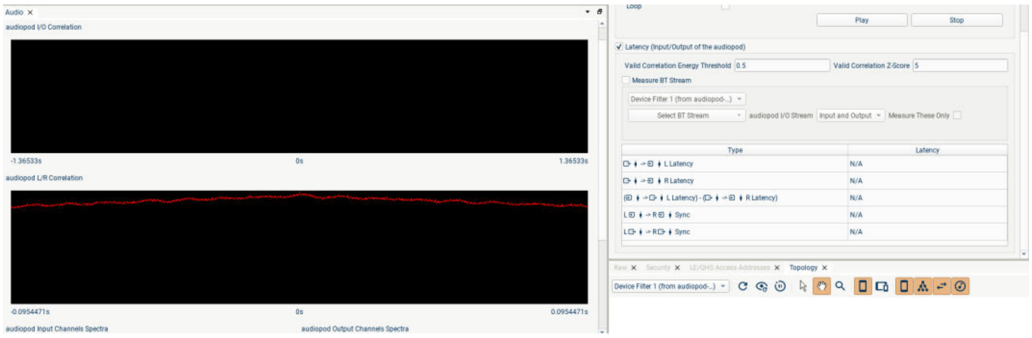
□ Measure BT Stream

Device Filter 1 (from audiopod-...) Select BT Stream audiopod I/O Stream Input and Output Measure These Only □

Type	Latency
⇨ ⧫ -> ⊟ ⧫ L Latency	+31.875ms
⇨ ⧫ -> ⊟ ⧫ R Latency	+31.875ms
(⊟ ⧫ -> ⇨ ⧫ L Latency) - (⇨ ⧫ -> ⊟ ⧫ R Latency)	+0.000ms
L⊟ ⧫ -> R⊟ ⧫ Sync	+0.000ms
L⇨ ⧫ -> R⇨ ⧫ Sync	+0.000ms

Device Filter 1 (from audiopod-...)

If the signals are unrelated you will see no graph, or just noise:



☑ Latency (Input/Output of the audiopod)

Valid Correlation Energy Threshold 0.5 Valid Correlation Z-Score 5

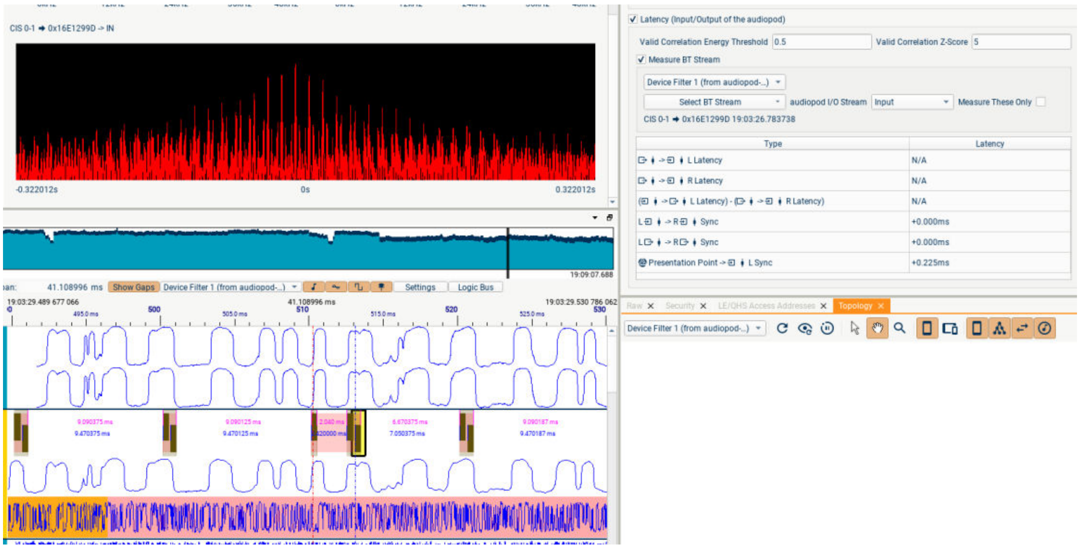
□ Measure BT Stream

Device Filter 1 (from audiopod-...) Select BT Stream audiopod I/O Stream Input and Output Measure These Only □

Type	Latency
⇨ ⧫ -> ⊟ ⧫ L Latency	N/A
⇨ ⧫ -> ⊟ ⧫ R Latency	N/A
(⊟ ⧫ -> ⇨ ⧫ L Latency) - (⇨ ⧫ -> ⊟ ⧫ R Latency)	N/A
L⊟ ⧫ -> R⊟ ⧫ Sync	N/A
L⇨ ⧫ -> R⇨ ⧫ Sync	N/A

Device Filter 1 (from audiopod-...)

If instead of one peak you see a number of peaks, this is likely to indicate that the audio signal you are using is too similar to a periodic signal, e.g. a single musical note:



☑ Latency (Input/Output of the audiopod)

Valid Correlation Energy Threshold 0.5 Valid Correlation Z-Score 5

☑ Measure BT Stream

Device Filter 1 (from audiopod-...) Select BT Stream audiopod I/O Stream Input Measure These Only □

CIS 0-1 ➜ 0x16E1299D 19:03:26.783738

Type	Latency
⇨ ⧫ -> ⊟ ⧫ L Latency	N/A
⇨ ⧫ -> ⊟ ⧫ R Latency	N/A
(⊟ ⧫ -> ⇨ ⧫ L Latency) - (⇨ ⧫ -> ⊟ ⧫ R Latency)	N/A
L⊟ ⧫ -> R⊟ ⧫ Sync	+0.000ms
L⇨ ⧫ -> R⇨ ⧫ Sync	+0.000ms
Presentation Point -> ⊟ ⧫ L Sync	+0.225ms

Device Filter 1 (from audiopod-...)

Ideally you should change the audio input to something with a wider bandwidth. If this isn't possible, you may still be able to get a latency measurement, but you may need to reduce either/both of the thresholds in the audiopod tab to lower the criteria for what is considered a valid correlation.

NB: You don't need to try to read time values from the graph! The position of the highest peak (if it meets the specified thresholds) will be reported in the table of latencies.

The second type of graph displays the spectra of the audio signals at the audio input and output. Currently this is purely an indication of what audio is currently being heard; in a subsequent update we will measure the difference between the input and output spectra to measure the frequency response of the channel.

8.3 Using audiopod to measure LE Audio latency

The example capture “audiopod_LE_Audio_CIG.pcapng” provides an example of using audiopod to measure all of the relevant subsections of latency of an LE Audio stream. The physical setup in this capture was:

1) Audio was played by audiopod into the input headphone jack of a development board.
2) The development board transmitted this audio over two CISes (left and right) to two other development boards.
3) The headphone jack outputs from these boards were combined and sent into the stereo jack input of audiopod.

Audiopod provides accurate timestamps for the audio at points (1) and (3), and the Moreph provides accurate timestamps for the on-air packets of (2). Using these timestamps and the detected CIS and audio parameters, we are able to calculate the “SDU Synchronization Reference” as detailed in this diagram from the TMAP test specification:

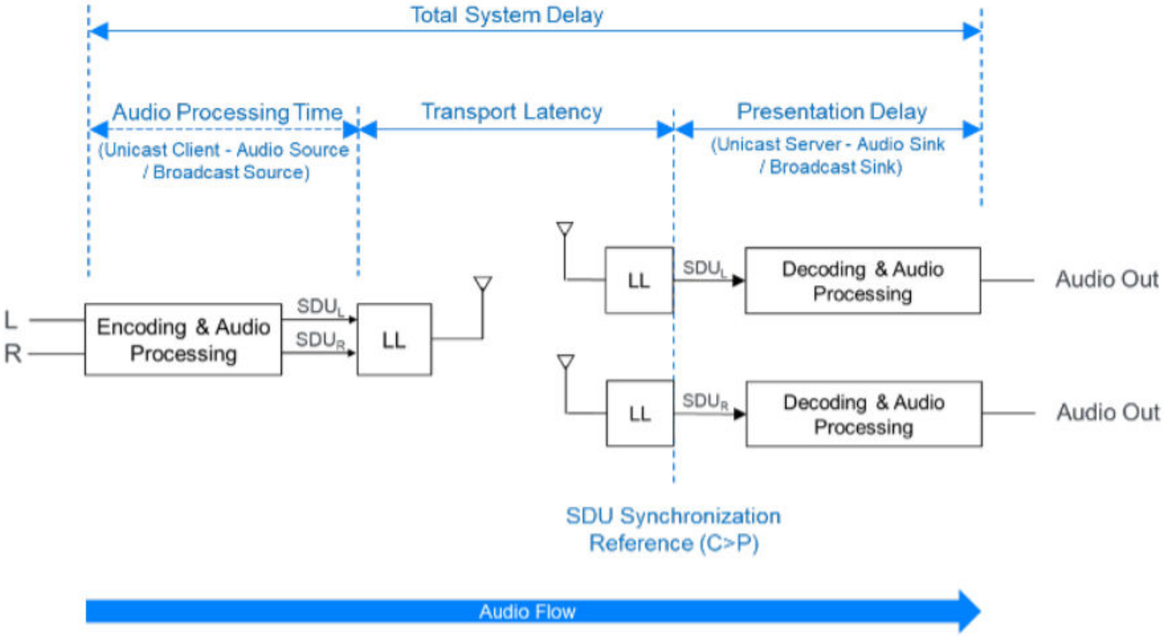
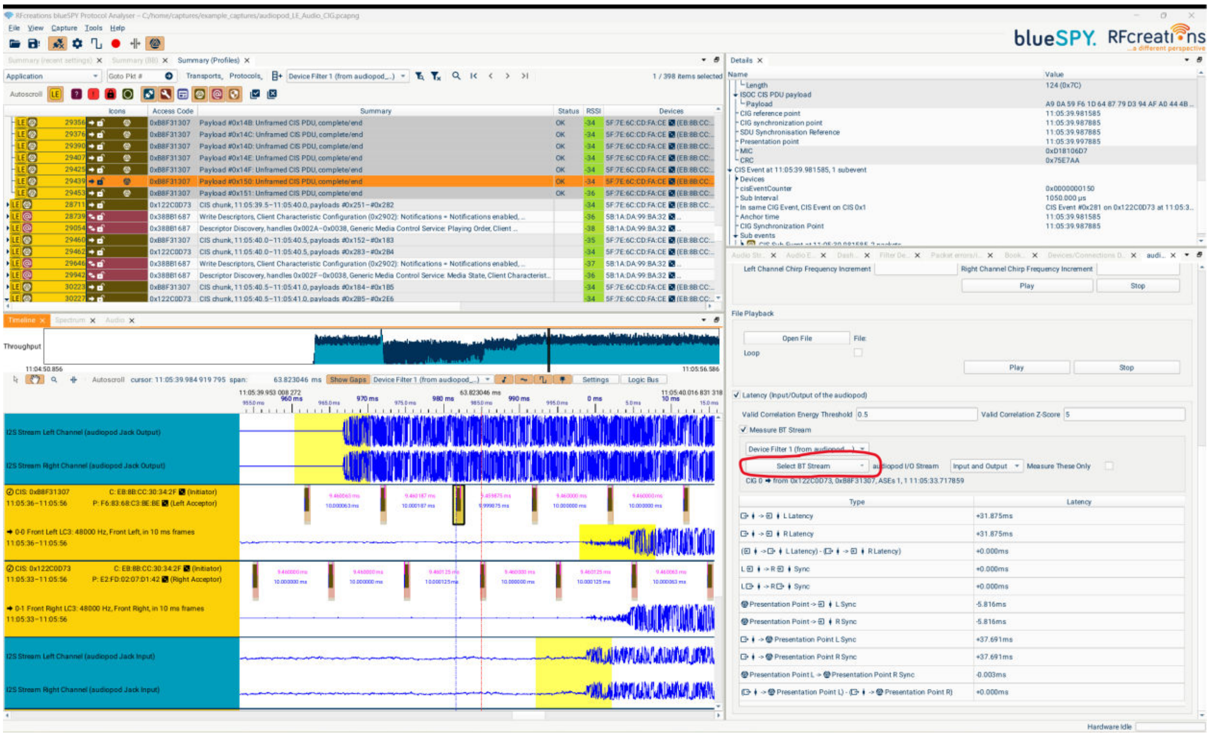

Total System Delay
Audio Processing Time
(Unicast Client - Audio Source / Broadcast Source)
Transport Latency
Presentation Delay
(Unicast Server - Audio Sink / Broadcast Sink)
L
R
Encoding & Audio Processing
SDU_L
SDU_R
LL
LL
SDU_L
Decoding & Audio Processing
Audio Out
LL
SDU_R
Decoding & Audio Processing
Audio Out
SDU Synchronization Reference (C>P)
Audio Flow

Figure 3.10: Unicast total system delay

If you select a baseband packet and go to the Details pane, you will find the SDU Synchronisation Reference for that packet, along with some of the other timing references related to the packet and the CIS Event.

To select a Bluetooth stream to compare with the audiopod input/output:

- First of all, ensure that the relevant devices are selected in the Device Filter
- Click on a suitable time in the Timeline, when the audio is playing.
- The audiostreams involving the filtered devices which are playing at that timepoint will now be available to select in the circled dropdown; here we have selected the CIG to measure both channels of the stereo audio.



RFcreations blueSPY Protocol Analyser – C:/home/captures/example_captures/audiopod_LE_Audio_CIG.pcapng

Summary (Profiles) — Device Filter 1 (from audiopod_...) — 1 / 398 items selected

Icons		Access Code	Summary	Status	RSSI	Devices
LE	29356	0xB8F31307	Payload #0x14B: Unframed CIS PDU, complete/end	OK	-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29376	0xB8F31307	Payload #0x14C: Unframed CIS PDU, complete/end	OK	-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29390	0xB8F31307	Payload #0x14D: Unframed CIS PDU, complete/end	OK	-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29407	0xB8F31307	Payload #0x14E: Unframed CIS PDU, complete/end	OK	-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29425	0xB8F31307	Payload #0x14F: Unframed CIS PDU, complete/end	OK	-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29439	0xB8F31307	Payload #0x150: Unframed CIS PDU, complete/end	OK	-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29453	0xB8F31307	Payload #0x151: Unframed CIS PDU, complete/end	OK	-36	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	28711	0x122C0D73	CIS chunk, 11:05:39.5–11:05:40.0, payloads #0x251–#0x282		-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	28739	0x38BB1687	Write Descriptors, Client Characteristic Configuration (0x2902): Notifications = Notifications enabled, ...		-36	5B:1A:DA:99:BA:32 ...
LE	29054	0x38BB1687	Descriptor Discovery, handles 0x002A–0x0038, Generic Media Control Service: Playing Order, Client ...		-38	5B:1A:DA:99:BA:32 ...
LE	29460	0xB8F31307	CIS chunk, 11:05:40.0–11:05:40.5, payloads #0x152–#0x183		-35	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29462	0x122C0D73	CIS chunk, 11:05:40.0–11:05:40.5, payloads #0x283–#0x2B4		-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	29646	0x38BB1687	Write Descriptors, Client Characteristic Configuration (0x2902): Notifications = Notifications enabled, ...		-37	5B:1A:DA:99:BA:32 ...
LE	29942	0x38BB1687	Descriptor Discovery, handles 0x002F–0x0038, Generic Media Control Service: Media State, Client Characterist...		-36	5B:1A:DA:99:BA:32 ...
LE	30223	0xB8F31307	CIS chunk, 11:05:40.5–11:05:41.0, payloads #0x184–#0x1B5		-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...
LE	30227	0x122C0D73	CIS chunk, 11:05:40.5–11:05:41.0, payloads #0x285–#0x2E6		-34	5F:7E:6C:CD:FA:CE (EB:8B:CC:...

Details

Name	Value
Length	124 (0x7C)
ISOC CIS PDU payload	
Payload	A9 0A 59 F6 1D 64 87 79 D3 94 AF A0 44 4B ...
CIG reference point	11:05:39.981585
CIG synchronization point	11:05:39.987885
SDU Synchronisation Reference	11:05:39.987885
Presentation point	11:05:39.997885
MIC	0xD18106D7
CRC	0x75E7AA
CIS Event at 11:05:39.981585, 1 subevent	
Devices	
cisEventCounter	0x0000000150
Sub Interval	1050.000 µs
In same CIG Event, CIS Event on CIS 0x1	CIS Event #0x281 on 0x122C0D73 at 11:05:3...
Anchor time	11:05:39.981585
CIG Synchronization Point	11:05:39.987885
Sub events	

Timeline — Throughput 11:04:50.856 – 11:05:56.586 — cursor: 11:05:39.984 919 795 span: 63.823046 ms — 11:05:39.953 008 272 – 11:05:40.016 831 318

I2S Stream Left Channel (audiopod Jack Output)
I2S Stream Right Channel (audiopod Jack Output)
CIS: 0xB8F31307 11:05:36–11:05:56 C: EB:8B:CC:30:34:2F (Initiator) P: F6:83:68:C3:BE:BE (Left Acceptor)
0-0 Front Left LC3: 48000 Hz, Front Left, in 10 ms frames 11:05:36–11:05:56
CIS: 0x122C0D73 11:05:33–11:05:56 C: EB:8B:CC:30:34:2F (Initiator) P: E2:FD:02:07:D1:42 (Right Acceptor)
0-1 Front Right LC3: 48000 Hz, Front Right, in 10 ms frames 11:05:33–11:05:56
I2S Stream Left Channel (audiopod Jack Input)
I2S Stream Right Channel (audiopod Jack Input)

☑ Latency (Input/Output of the audiopod)
Valid Correlation Energy Threshold: 0.5 Valid Correlation Z-Score: 5
☑ Measure BT Stream
Device Filter 1 (from audiopod_...) — Select BT Stream — audiopod I/O Stream: Input and Output — Measure These Only □
CIG 0 from 0x122C0D73, 0xB8F31307, ASEs 1, 1 11:05:33.717859

Type	Latency
→ L Latency	+31.875ms
→ R Latency	+31.875ms
(→ L Latency) - (→ R Latency)	+0.000ms
L → R Sync	+0.000ms
L → R Sync	+0.000ms
Presentation Point → L Sync	-5.816ms
Presentation Point → R Sync	-5.816ms
→ Presentation Point L Sync	+37.691ms
→ Presentation Point R Sync	+37.691ms
Presentation Point L → Presentation Point R Sync	-0.003ms
(→ Presentation Point L) - (→ Presentation Point R)	+0.000ms

The four rows of audio displayed here correspond to the three points in the audio chain mentioned above. The first row is the stereo output from audiopod, point (1). The next two rows are the two CISes (left and right), point (2). The final row is the output from the two development boards, returned to the stereo input of audiopod, point (3).

The yellow highlighted regions represent the same 10 ms frame of audio, at the different points:

1) The 10ms of audio played into the development board starting at 11:05:39.960194 is captured by the Central development board
2) The left channel of this audio is encoded using LC3 and transmitted in packet #29439, sent at 11:05:39.981585. Using the CIS Offset, CIG & CIS Sync Delays, Presentation Delay and other CIS parameters, it's possible to calculate the correct start time or "Presentation Point" for the frame of audio: 11:05:39.997885. This Presentation Point is the start of the yellow highlighted region on the second row.
3) The audio from packet #29439 is decoded by the development board and played out of its headphone jack starting at 11:05:39.992069, i.e. 5.816 ms early. This is displayed in the table of latencies as "Presentation Point -> audiopod L channel input Sync : -5.816 ms" instead of the ideal value of 0.000 ms

So although the total latency from audiopod output to audiopod input is a commendable 31.875 ms, this has unfortunately been achieved via a miscalculation of the correct presentation delay to apply.

On the most important measure, the Left-Right Sync, this implementation has done rather better. The audio played out of audiopod here is identical on the two channels, and so can be directly compared in the OTA packet and at the analogue output; the synchronisation here is almost perfect.

This example showed a setup where cabled audio input and output were possible; this allows measuring the end-to-end latency, and all of the sub-components of the latency defined in the TMAP and GMAP specifications (e.g. the audio input -> SDU sync reference timing is tested for Peripheral -> Central "uplink" audio).

When cabled audio is not available, the microphones can be used to record a DUT's speaker's output (e.g. by placing the earbuds you are testing into the earcanal microphones), and wired earbuds or other speakers plugged into audiopod's outputs can be used to play into a DUT's microphone.

When audio input to the system is not available (e.g. when testing using a smartphone or laptop as the Central), the second portion of the system (SDU Sync Reference -> audio output) can still be measured, provided a reasonable source of audio (music with a reasonably wide bandwidth) is available on the device.

8.4 Using audiopod to measure A2DP, HFP, and ASHA latency

In the Bluetooth audio specifications prior to LE Audio:

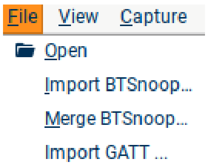
1) the correct time to render the audio is not defined;
2) the transport packets are not always sent at fixed intervals;

and so the calculations carried out by blueSPY are a little different. Instead of showing the discrepancy between a frame of audio's "Presentation Point" and the time at which it was rendered, we show the time interval between a packet containing audio (e.g. an AVDTP packet) and the time it is rendered by the receiver. As in this case the transport isn't synchronous, we show the mean, minimum, and maximum value of this time interval within the 1 second of audio used.

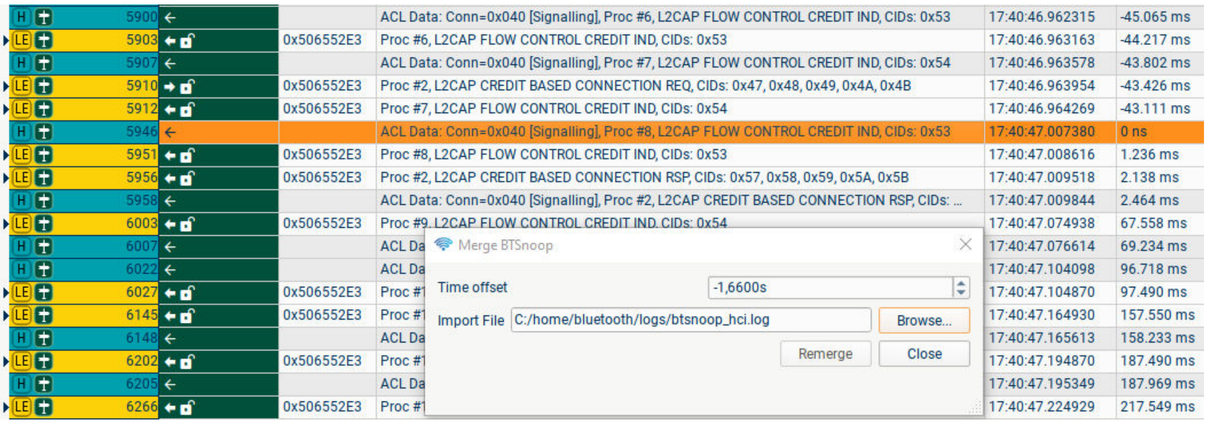
9 Importing GATT and HCI

If you have information to add to the air packets captured by the sniffer, they can be imported into the airtrace and displayed integrated with the Bluetooth traffic. You can use the injection interface described in section 6, or you can import complete files. When importing files, currently we only support import of GATT, and HCI from btsnoop logs, but various other formats/methods of import are planned. If there is a format you would particularly appreciate, let us know and we can prioritise it.

9.1 Btsnoop Import/Merge



The btsnoop file format (with slight variations) is used by (at least) Android and blueZ for logging HCI traffic. If you have files in this format, they can either be imported (creating a new capture file) or merged into an existing file, including ongoing captures. NB: When you merge into a live capture, the files will be merged and the resulting files reloaded from scratch. No packets will be missed, but the resulting reload may take some time for large captures; you can watch the progress using the orange bar in the bottom right corner.



H	5900	←		ACL Data: Conn=0x040 [Signalling], Proc #6, L2CAP FLOW CONTROL CREDIT IND, CIDs: 0x53	17:40:46.962315	-45.065 ms
LE	5903	←	0x506552E3	Proc #6, L2CAP FLOW CONTROL CREDIT IND, CIDs: 0x53	17:40:46.963163	-44.217 ms
H	5907	←		ACL Data: Conn=0x040 [Signalling], Proc #7, L2CAP FLOW CONTROL CREDIT IND, CIDs: 0x54	17:40:46.963578	-43.802 ms
LE	5910	→	0x506552E3	Proc #2, L2CAP CREDIT BASED CONNECTION REQ, CIDs: 0x47, 0x48, 0x49, 0x4A, 0x4B	17:40:46.963954	-43.426 ms
LE	5912	←	0x506552E3	Proc #7, L2CAP FLOW CONTROL CREDIT IND, CIDs: 0x54	17:40:46.964269	-43.111 ms
H	5946	←		ACL Data: Conn=0x040 [Signalling], Proc #8, L2CAP FLOW CONTROL CREDIT IND, CIDs: 0x53	17:40:47.007380	0 ns
LE	5951	←	0x506552E3	Proc #8, L2CAP FLOW CONTROL CREDIT IND, CIDs: 0x53	17:40:47.008616	1.236 ms
LE	5956	←	0x506552E3	Proc #2, L2CAP CREDIT BASED CONNECTION RSP, CIDs: 0x57, 0x58, 0x59, 0x5A, 0x5B	17:40:47.009518	2.138 ms
H	5958	←		ACL Data: Conn=0x040 [Signalling], Proc #2, L2CAP CREDIT BASED CONNECTION RSP, CIDs: ...	17:40:47.009844	2.464 ms
LE	6003	←	0x506552E3	Proc #9, L2CAP FLOW CONTROL CREDIT IND, CIDs: 0x54	17:40:47.074938	67.558 ms
H	6007	←		ACL Da	17:40:47.076614	69.234 ms
H	6022	←		ACL Da	17:40:47.104098	96.718 ms
LE	6027	←	0x506552E3	Proc #1	17:40:47.104870	97.490 ms
LE	6145	←	0x506552E3	Proc #1	17:40:47.164930	157.550 ms
H	6148	←		ACL Da	17:40:47.165613	158.233 ms
LE	6202	←	0x506552E3	Proc #1	17:40:47.194870	187.490 ms
H	6205	←		ACL Da	17:40:47.195349	187.969 ms
LE	6266	←	0x506552E3	Proc #1	17:40:47.224929	217.549 ms

Merge BTSnoop

Time offset: -1,6600s

Import File: C:/home/bluetooth/logs/btsnoop_hci.log

There is usually some timing offset between the clock used for the on-air capture (derived from your laptop's clock) and the clock of the Android phone. To get a rough time-alignment, find some packets which are present in both the airtrace and the HCI log (L2CAP Control packets are a good choice as they have a clear procedure number), eyeball the difference between the streams, and input the time difference before using "Remerge".

9.2 GATT Import/Export

In LE connections, missing GATT attribute definitions can make it very hard to understand the traffic you capture, and in particular can prevent decode of LE Audio in some cases. Attribute definitions may be missing because of a few corrupted packets, or may be entirely missing if the capture contains a reconnection of paired devices which are using GATT caching. We attempt to cache on your computer any GATT traffic that we see and use it in subsequent captures, but this will fail if:

(a) the initial pairing was not captured using blueSPY and the same computer
(b) the devices do not read the Database Hash attribute, or the read is in a CRC-fail packet or otherwise corrupted.

We currently support three methods of fixing this problem, detailed below. After adding GATT definitions using any of these methods, the Summary strings and any new Details trees will be updated immediately; but to reanalyse a CIG and decode audio using the new information, you will need to press reload in the Security tab.

9.2.1 Manual correction

Individual attributes with a missing definition can be edited by clicking on a relevant packet and right-clicking on the attribute in the Details tree, and then setting the UUID (16-bit, 32-bit or 128-bit). This is probably most useful for adding a missing ASE Control Point definition; fixing this gives most of the information required for decoding LE Audio.

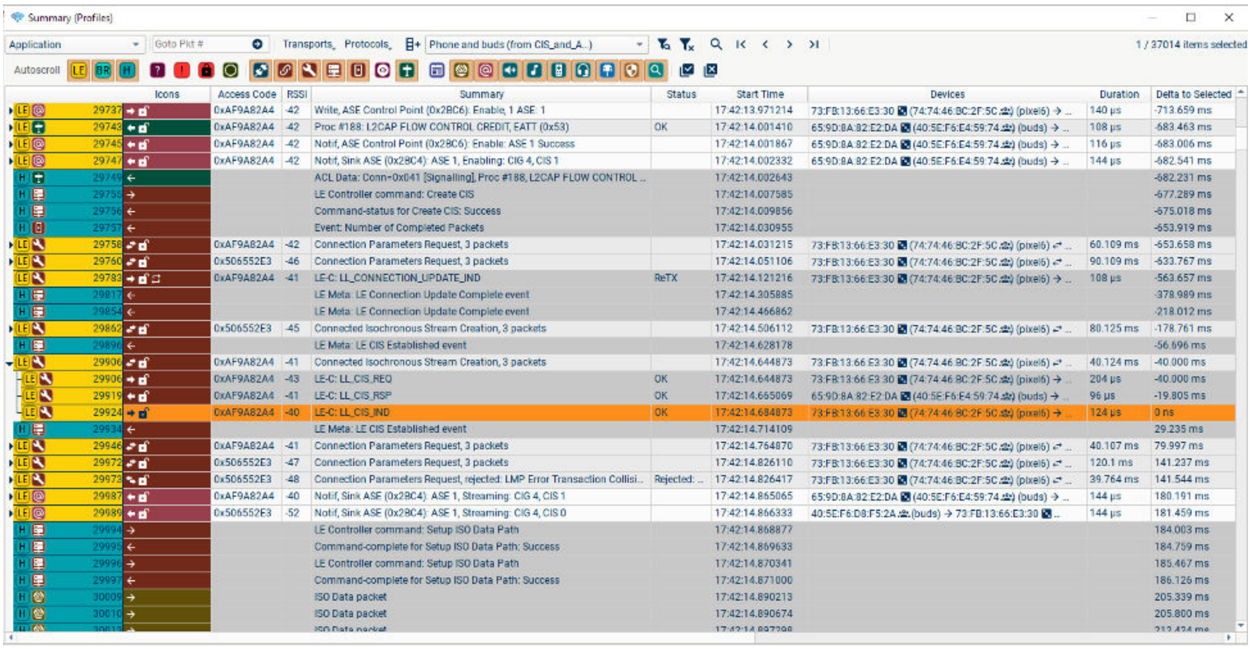
9.2.2 Android GATT database import

If one of your DUTs is an Android phone which you have root access to, you can copy the files from /data/misc/Bluetooth (`adb pull /data/misc/bluetooth`) and import the gatt_hash_* files you find there (File -> Import GATT...)

9.2.3 blueSPY GATT import and export

If the initial pairing was captured in blueSPY on another laptop, you can open the relevant capture, export the GATT database information (File -> Export -> Export GATT), and then import the file to add missing GATT to a subsequent capture. These files use a simple JSON format detailing the attribute handles and the matching UUIDs, so if you are able to generate these files from your build system or fix missing attributes in an exported file then blueSPY can import this information.

10 Summary Tab



Summary (Profiles)

Application | Goto Pkt # | Transports, Protocols, | Phone and buds (from CIS_and_A...) | 1 / 37014 items selected

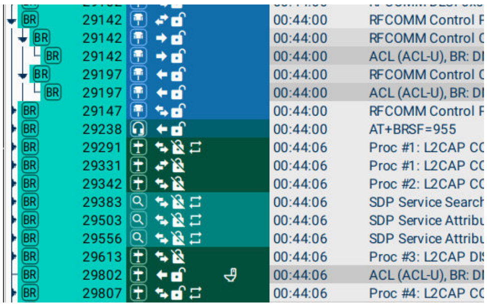
Autoscroll LE BR H

	Icons	Access Code	RSSI	Summary	Status	Start Time	Devices	Duration	Delta to Selected
LE 29737	→	0xAF9A82A4	-42	Write, ASE Control Point (0x2BC6): Enable, 1 ASE: 1		17:42:13.971214	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) → ...	140 µs	-713.659 ms
LE 29743	←	0xAF9A82A4	-42	Proc #188: L2CAP FLOW CONTROL CREDIT, EATT (0x53)	OK	17:42:14.001410	65:9D:8A:82:E2:DA ▣ (40:5E:F6:E4:59:74) (buds) → ...	108 µs	-683.463 ms
LE 29745	←	0xAF9A82A4	-42	Notif, ASE Control Point (0x2BC6): Enable: ASE 1 Success		17:42:14.001867	65:9D:8A:82:E2:DA ▣ (40:5E:F6:E4:59:74) (buds) → ...	116 µs	-683.006 ms
LE 29747	←	0xAF9A82A4	-42	Notif, Sink ASE (0x2BC4): ASE 1, Enabling: CIG 4, CIS 1		17:42:14.002332	65:9D:8A:82:E2:DA ▣ (40:5E:F6:E4:59:74) (buds) → ...	144 µs	-682.541 ms
H 29749	←			ACL Data: Conn-0x041 [Signalling], Proc #188, L2CAP FLOW CONTROL ...		17:42:14.002643			-682.231 ms
H 29755	→			LE Controller command: Create CIS		17:42:14.007585			-677.289 ms
H 29756	←			Command-status for Create CIS: Success		17:42:14.009856			-675.018 ms
H 29757	←			Event: Number of Completed Packets		17:42:14.030955			-653.919 ms
LE 29758	⇄	0xAF9A82A4	-42	Connection Parameters Request, 3 packets		17:42:14.031215	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) ⇄ ...	60.109 ms	-653.658 ms
LE 29760	⇄	0x506552E3	-46	Connection Parameters Request, 3 packets		17:42:14.051106	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) ⇄ ...	90.109 ms	-633.767 ms
LE 29783	→	0xAF9A82A4	-41	LE-C: LL_CONNECTION_UPDATE_IND	ReTX	17:42:14.121216	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) → ...	108 µs	-563.657 ms
H 29817	←			LE Meta: LE Connection Update Complete event		17:42:14.305885			-378.989 ms
H 29854	←			LE Meta: LE Connection Update Complete event		17:42:14.466862			-218.012 ms
LE 29862	⇄	0x506552E3	-45	Connected Isochronous Stream Creation, 3 packets		17:42:14.506112	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) ⇄ ...	80.125 ms	-178.761 ms
H 29896	←			LE Meta: LE CIS Established event		17:42:14.628178			-56.696 ms
LE 29906	⇄	0xAF9A82A4	-41	Connected Isochronous Stream Creation, 3 packets		17:42:14.644873	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) ⇄ ...	40.124 ms	-40.000 ms
LE 29906	→	0xAF9A82A4	-43	LE-C: LL_CIS_REQ	OK	17:42:14.644873	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) → ...	204 µs	-40.000 ms
LE 29919	←	0xAF9A82A4	-41	LE-C: LL_CIS_RSP	OK	17:42:14.665069	65:9D:8A:82:E2:DA ▣ (40:5E:F6:E4:59:74) (buds) → ...	96 µs	-19.805 ms
LE 29924	→	0xAF9A82A4	-40	LE-C: LL_CIS_IND	OK	17:42:14.684873	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) → ...	124 µs	0 ns
H 29934	←			LE Meta: LE CIS Established event		17:42:14.714109			29.235 ms
LE 29946	⇄	0xAF9A82A4	-41	Connection Parameters Request, 3 packets		17:42:14.764870	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) ⇄ ...	40.107 ms	79.997 ms
LE 29972	⇄	0x506552E3	-47	Connection Parameters Request, 3 packets		17:42:14.826110	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) ⇄ ...	120.1 ms	141.237 ms
LE 29973	⇄	0x506552E3	-48	Connection Parameters Request, rejected: LMP Error Transaction Collisi...	Rejected: ...	17:42:14.826417	73:FB:13:66:E3:30 ▣ (74:74:46:BC:2F:5C) (pixel6) ⇄ ...	39.764 ms	141.544 ms
LE 29987	←	0xAF9A82A4	-40	Notif, Sink ASE (0x2BC4): ASE 1, Streaming: CIG 4, CIS 1		17:42:14.865065	65:9D:8A:82:E2:DA ▣ (40:5E:F6:E4:59:74) (buds) → ...	144 µs	180.191 ms
LE 29989	←	0x506552E3	-52	Notif, Sink ASE (0x2BC4): ASE 1, Streaming: CIG 4, CIS 0		17:42:14.866333	40:5E:F6:D8:F5:2A (buds) → 73:FB:13:66:E3:30 ▣ ...	144 µs	181.459 ms
H 29994	→			LE Controller command: Setup ISO Data Path		17:42:14.868877			184.003 ms
H 29995	←			Command-complete for Setup ISO Data Path: Success		17:42:14.869633			184.759 ms
H 29996	→			LE Controller command: Setup ISO Data Path		17:42:14.870341			185.467 ms
H 29997	←			Command-complete for Setup ISO Data Path: Success		17:42:14.871000			186.126 ms
H 30009	→			ISO Data packet		17:42:14.890213			205.339 ms
H 30010	→			ISO Data packet		17:42:14.890674			205.800 ms
H 30012	→			ISO Data packet		17:42:14.897298			212.424 ms

The Summary tabs are the main interface and control window for the application. By default they display all of the captured packets, aggregated into e.g. GATT Procedures, LE Advertising Events, etc.

Selecting a packet or higher-layer event in the Summary will:

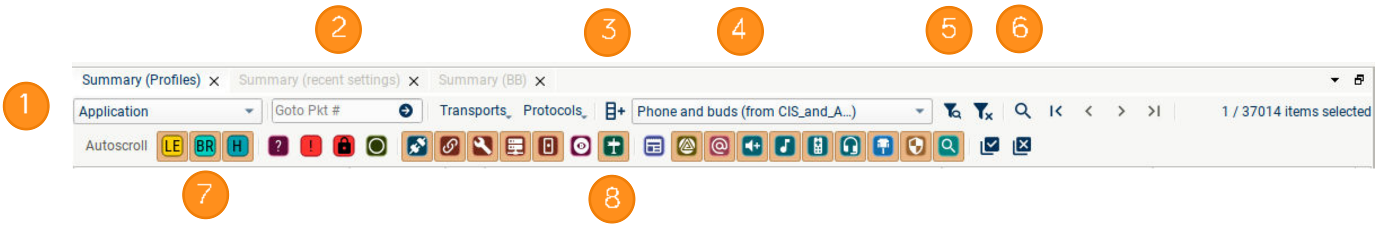
- display more information about the packet in the Details tab;
- load the raw bytes into the Raw tab;
- scroll the Spectrum and Timeline tabs to the correct time, and select the event there unless it is hidden in that view.



BR 29142	00:44:00	RFCOMM Control F
BR 29142	00:44:00	RFCOMM Control C
BR 29142	00:44:00	ACL (ACL-U), BR: DI
BR 29197	00:44:00	RFCOMM Control C
BR 29197	00:44:00	ACL (ACL-U), BR: DI
BR 29147	00:44:00	RFCOMM Control F
BR 29238	00:44:00	AT+BRSF=955
BR 29291	00:44:06	Proc #1: L2CAP CC
BR 29331	00:44:06	Proc #1: L2CAP CC
BR 29342	00:44:06	Proc #2: L2CAP CC
BR 29383	00:44:06	SDP Service Search
BR 29503	00:44:06	SDP Service Attribu
BR 29556	00:44:06	SDP Service Attribu
BR 29613	00:44:06	Proc #3: L2CAP DI
BR 29802	00:44:06	ACL (ACL-U), BR: DI
BR 29807	00:44:06	Proc #4: L2CAP CC

Expanding a higher layer event (using the arrow at the left-hand edge) will display the constituent packets from the lower protocol layer.

During Capture, the "Autoscroll" button in the top left of the window causes the view to automatically scroll, displaying the most recently captured packets. "Autoscroll" will be disabled when a packet is selected. Equivalent "Autoscroll" buttons are in the top left of the Spectrum and Timeline tabs, to provide independent control of this feature.



1 View selector

2 GoTo box

3 Add/modify columns, and add coloured columns matching some filter.

4 Choose which Device Filter to use, or choose no device filtering.

5 Complex filter popup

6 Search

7 Transport filter buttons

8 Protocol and Profile filter buttons

10.1 Finding and Filtering packets

To find and display packets of interest there are three main tools. If you are unsure which filters are currently enabled, hover over the "Clear All Filters" button to see a summary. If you can't see packets you are expecting to receive, check whether they appear if you "Clear All Filters"!

10.1.1 Protocol and Profile filters

The simplest filtering tool consists of three sets of buttons. The buttons enable/disable the following types of packet:

1. Transports:

BR BR/EDR "Classic" traffic

LE LE traffic

WiFi packets

Z 802.15.4 packets, e.g. Zigbee

H HCI packets

Q QBHSL traffic

CS Channel-sounding tones and sync packets

... and various other proprietary transports.

2. Empty/degraded packets:

! Unintelligible packets: CRC failures, packets not dewhitened, incomplete L2CAP PDUs

🔒 Encrypted packets

? Unknown packets

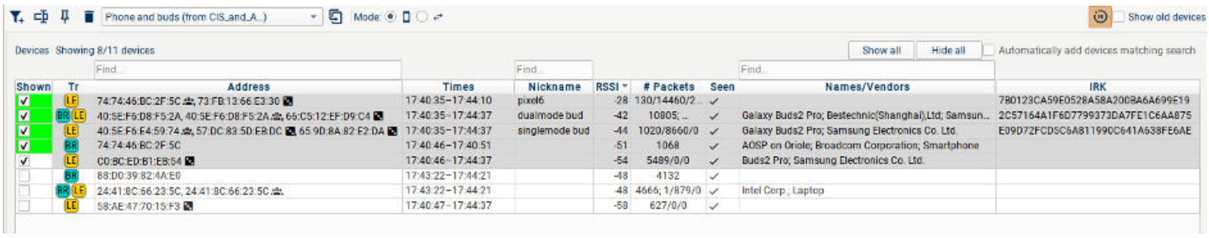
○ Empty packets

3. Protocols/other categories of packet (LE advertising, FHS, etc). E.g.

The "All" and "None" buttons are shortcuts to enable/disable all toggle buttons in this row. ☑ ☒

Descriptions of these icons, and all other icons and symbols used in the software, can be found in the "Glossary of Symbols" in the Help menu (shortcut F1).

10.1.2 Filter Devices



Phone and buds (from CIS_and_A...) ▾ Mode: ● ▯ ○ ⇄ □ Show old devices

Devices Showing 8/11 devices Show all Hide all □ Automatically add devices matching search

Shown	Tr	Address	Times	Nickname	RSSI	# Packets	Seen	Names/Vendors	IRK
☑	LE	74:74:46:BC:2F:5C, 73:FB:13:66:E3:30	17:40:35–17:44:10	pixel6	-28	130/14460/2...	✓		7B0123CA59E0528A58A200BA6A699E19
☑	BR LE	40:5E:F6:D8:F5:2A, 40:5E:F6:D8:F5:2A, 66:C5:12:EF:D9:C4	17:40:35–17:44:37	dualmode bud	-42	10805; ...	✓	Galaxy Buds2 Pro; Bestechnic(Shanghai),Ltd; Samsun...	2C57164A1F6D7799373DA7FE1C6AA875
☑	LE	40:5E:F6:E4:59:74, 57:DC:83:5D:EB:DC, 65:9D:8A:82:E2:DA	17:40:35–17:44:37	singlemode bud	-44	1020/8660/0	✓	Galaxy Buds2 Pro; Samsung Electronics Co. Ltd.	E09D72FCD5C6A811990C641A638FE6AE
☑	BR	74:74:46:BC:2F:5C	17:40:46–17:40:51		-51	1068	✓	AOSP on Oriole; Broadcom Corporation; Smartphone	
☑	LE	C0:BC:ED:B1:EB:54	17:40:46–17:44:37		-54	5489/0/0	✓	Buds2 Pro; Samsung Electronics Co. Ltd.	
□	BR	88:D0:39:82:4A:E0	17:43:22–17:44:21		-48	4132	✓		
□	BR LE	24:41:8C:66:23:5C, 24:41:8C:66:23:5C	17:43:22–17:44:21		-48	4666; 1/879/0	✓	Intel Corp.; Laptop	
□	LE	58:AE:47:70:15:F3	17:40:47–17:44:37		-58	627/0/0	✓		

The second filtering tool allows LE and BR/EDR devices to be hidden or displayed. It is possible to create multiple device filters, which can be used independently in different tabs; so for instance you can create a "Broadcasters" filter to display in one Summary tab, and a "Phone and earbuds" filter to display in a second Summary tab.

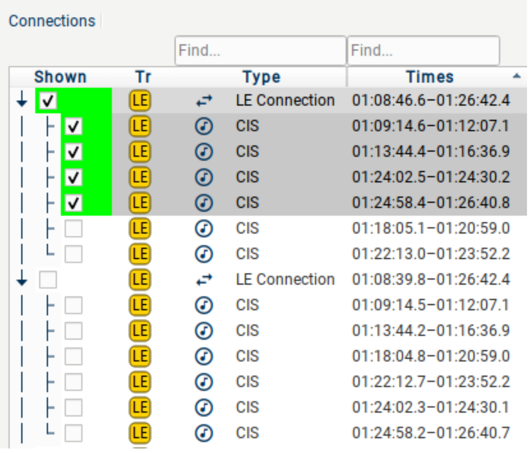
Use the dropdown menu to select which filter you are editing, and the New Filter, Rename Filter and Delete Filter buttons to modify the list of filters. In each of the other relevant tabs (Summaries, Timeline, Spectrum, Topology, Dashboard) there is a dropdown menu to select one of the filters you have created, or to choose No Device Filtering. If you want to use one filter throughout blueSPY, press the "Use this filter in all tabs" button in the Filter Device tab.

Any device filter you interact with (create or modify) during capture will be stored in the capture file on Save; to make a device filter available for future captures, press the Pin button and it will be stored on your computer until you "unpin" it.

The filter has two modes, selected using the "Devices" and "Connections" radiobuttons:

Mode: ○ ▯ ● ⇄

In Devices mode, the filter displays all packets involving a selected device, and all packets on any connection involving the device. In Connections mode, you can filter these packets further; the connections and audio streams involving the selected devices are shown in the lower pane of the tab, and you can choose which connections, and which streams (CIS, A2DP etc) on a connection to show.

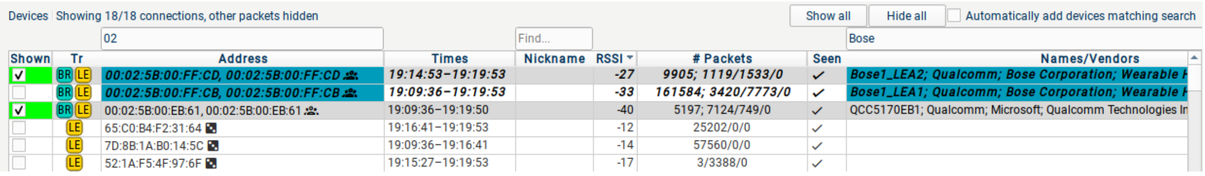


Connections

Shown	Tr	Type	Times
☑	LE	LE Connection	01:08:46.6–01:26:42.4
☑	LE	CIS	01:09:14.6–01:12:07.1
☑	LE	CIS	01:13:44.4–01:16:36.9
☑	LE	CIS	01:24:02.5–01:24:30.2
☑	LE	CIS	01:24:58.4–01:26:40.8
□	LE	CIS	01:18:05.1–01:20:59.0
□	LE	CIS	01:22:13.0–01:23:52.2
□	LE	LE Connection	01:08:39.8–01:26:42.4
□	LE	CIS	01:09:14.5–01:12:07.1
□	LE	CIS	01:13:44.2–01:16:36.9
□	LE	CIS	01:18:04.8–01:20:59.0
□	LE	CIS	01:22:12.7–01:23:52.2
□	LE	CIS	01:24:02.3–01:24:30.1
□	LE	CIS	01:24:58.2–01:26:40.7

When you use Connections mode, the Shown column of the Devices pane changes colour to highlight which connections involving a device are enabled in the filter; solid green for showing all connections, pale green for showing some connections, and clear for showing no connections involving that device.

In either pane, you can search for devices using any of the "Find..." boxes at the top of a column. Devices matching your search will be shown in bold, with a blue background in the relevant columns, and brought to the top of the pane. NB: If you enter text in more than one "Find..." box, the search only finds devices matching both search strings; so if you type in the Names/Vendors field and nothing appears, check that you haven't left a partial address in the Address box.



Devices Showing 18/18 connections, other packets hidden Show all Hide all □ Automatically add devices matching search

Address: 02 Names/Vendors: Bose

Shown	Tr	Address	Times	Nickname	RSSI	# Packets	Seen	Names/Vendors
☑	BR LE	**00:02:5B:00:FF:CD, 00:02:5B:00:FF:CD**	**19:14:53–19:19:53**		**-27**	**9905; 1119/1533/0**	✓	**Bose1_LEA2; Qualcomm; Bose Corporation; Wearable H**
□	BR LE	**00:02:5B:00:FF:CB, 00:02:5B:00:FF:CB**	**19:09:36–19:19:53**		**-33**	**161584; 3420/7773/0**	✓	**Bose1_LEA1; Qualcomm; Bose Corporation; Wearable H**
☑	BR LE	00:02:5B:00:EB:61, 00:02:5B:00:EB:61	19:09:36–19:19:50		-40	5197; 7124/749/0	✓	QCC5170EB1; Qualcomm; Microsoft; Qualcomm Technologies In
□	LE	65:C0:B4:F2:31:64	19:16:41–19:19:53		-12	25202/0/0	✓	
□	LE	7D:8B:1A:B0:14:5C	19:09:36–19:16:41		-14	57560/0/0	✓	
□	LE	52:1A:F5:4F:97:6F	19:15:27–19:19:53		-17	3/3388/0	✓	

To find the devices you want to display, there are two tools to help:

1. You can sort the panes by any of the columns.
2. You can filter based on substring matching on either Address (ignoring colons), Nickname, or Names/Vendors (case-sensitive).

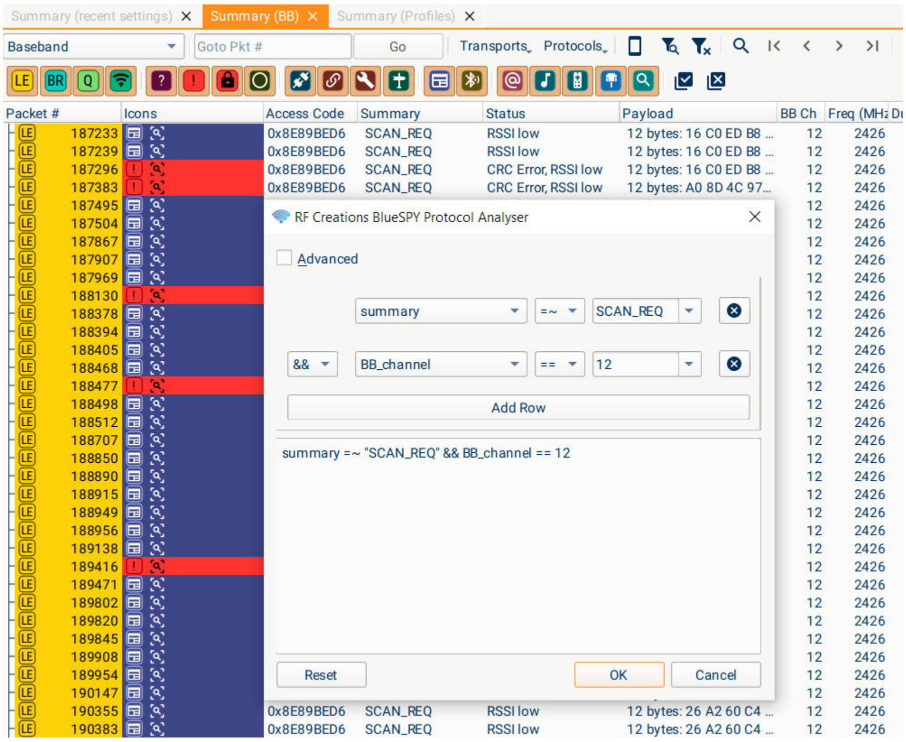
In the Devices pane of the filter, you can check "Automatically add devices matching search" to have devices added for you; e.g. if the device you are testing is frequently changing RPA (and no IRK has been seen), you can add its name or vendor to the search box and have all copies of the device added to the filter.

Any known IRKs are shown in the rightmost column of the tab (you may need to expand the tab or scroll right on lower resolution displays to see this column). If you know an IRK that has not been seen by the sniffer, you can add the IRK in this column. You can also set a nickname for any of the devices, and this nickname will then be used in place of/as well as the address in other parts of blueSPY.

10.1.3 Search interface

The final, more complex search tool is the Search and Filter interface. This allows you to search on a variety of queries associated with the packets. The available queries are listed below. It supports matching strings by regular expressions using the =~ operation. The supported regular expression syntax is documented here. It also supports == and != for exact comparison, < > <= >= for comparing numerical values, and && and || for combining expressions with logical AND or OR respectively.

For example, the search below finds all SCAN_REQs sent on channel 12. We use the =~ operation to find any summary containing the string SCAN_REQ.



Packet #	Icons	Access Code	Summary	Status	Payload	BB Ch	Freq (MHz
LE 187233		0x8E89BED6	SCAN_REQ	RSSI low	12 bytes: 16 C0 ED B8 ...	12	2426
LE 187239		0x8E89BED6	SCAN_REQ	RSSI low	12 bytes: 16 C0 ED B8 ...	12	2426
LE 187296	!	0x8E89BED6	SCAN_REQ	CRC Error, RSSI low	12 bytes: 16 C0 ED B8 ...	12	2426
LE 187383	!	0x8E89BED6	SCAN_REQ	CRC Error, RSSI low	12 bytes: A0 8D 4C 97...	12	2426
LE 187495						12	2426
LE 187504						12	2426
LE 187867						12	2426
LE 187907						12	2426
LE 187969						12	2426
LE 188130	!					12	2426
LE 188378						12	2426
LE 188394						12	2426
LE 188405						12	2426
LE 188468						12	2426
LE 188477	!					12	2426
LE 188498						12	2426
LE 188512						12	2426
LE 188707						12	2426
LE 188850						12	2426
LE 188890						12	2426
LE 188915						12	2426
LE 188949						12	2426
LE 188956						12	2426
LE 189138						12	2426
LE 189416	!					12	2426
LE 189471						12	2426
LE 189802						12	2426
LE 189820						12	2426
LE 189845						12	2426
LE 189908						12	2426
LE 189954						12	2426
LE 190147						12	2426
LE 190355		0x8E89BED6	SCAN_REQ	RSSI low	12 bytes: 26 A2 60 C4 ...	12	2426
LE 190383		0x8E89BED6	SCAN_REQ	RSSI low	12 bytes: 26 A2 60 C4 ...	12	2426

RF Creations BlueSPY Protocol Analyser

□ Advanced

summary =~ SCAN_REQ

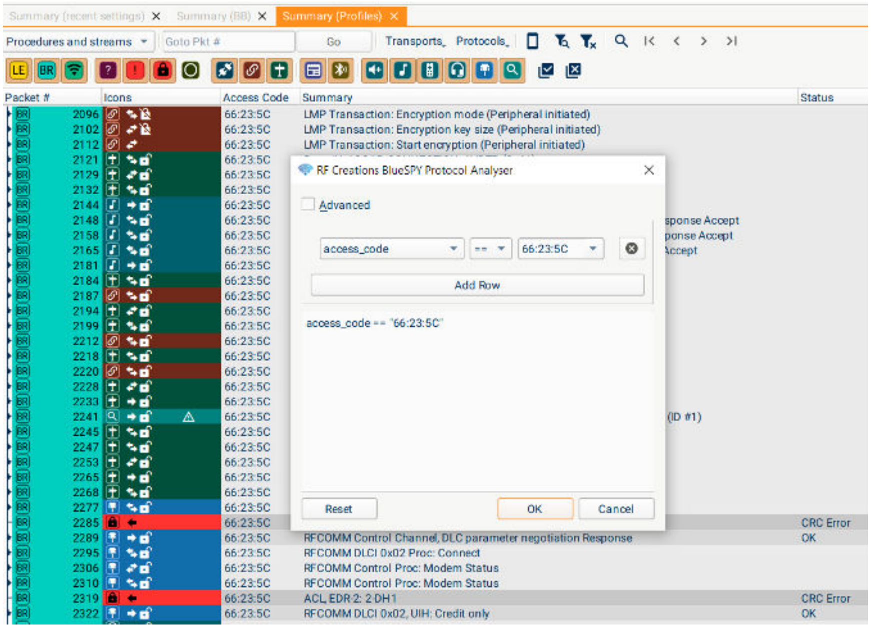
&& BB_channel == 12

Add Row

summary =~ "SCAN_REQ" && BB_channel == 12

Reset OK Cancel

The following search illustrates filtering a particular LAP.



Packet #	Access Code	Summary	Status
2096	66:23:5C	LMP Transaction: Encryption mode (Peripheral initiated)	
2102	66:23:5C	LMP Transaction: Encryption key size (Peripheral initiated)	
2112	66:23:5C	LMP Transaction: Start encryption (Peripheral initiated)	
2121	66:23:5C		
2129	66:23:5C		
2132	66:23:5C		
2144	66:23:5C		
2148	66:23:5C		
2158	66:23:5C	sponse Accept	
2165	66:23:5C	ponse Accept	
2181	66:23:5C	Accept	
2184	66:23:5C		
2187	66:23:5C		
2194	66:23:5C		
2199	66:23:5C		
2212	66:23:5C		
2218	66:23:5C		
2220	66:23:5C		
2228	66:23:5C		
2233	66:23:5C		
2241	66:23:5C	(ID #1)	
2245	66:23:5C		
2247	66:23:5C		
2253	66:23:5C		
2265	66:23:5C		
2268	66:23:5C		
2277	66:23:5C		
2285	66:23:5C		CRC Error
2289	66:23:5C	RFCOMM Control Channel, DLC parameter negotiation Response	OK
2295	66:23:5C	RFCOMM DLCI 0x02 Proc: Connect	
2306	66:23:5C	RFCOMM Control Proc: Modem Status	
2310	66:23:5C	RFCOMM Control Proc: Modem Status	
2319	66:23:5C	ACL, EDR 2: 2 DH1	CRC Error
2322	66:23:5C	RFCOMM DLCI 0x02, UIH: Credit only	OK

RF Creations BlueSPY Protocol Analyser

☐ Advanced

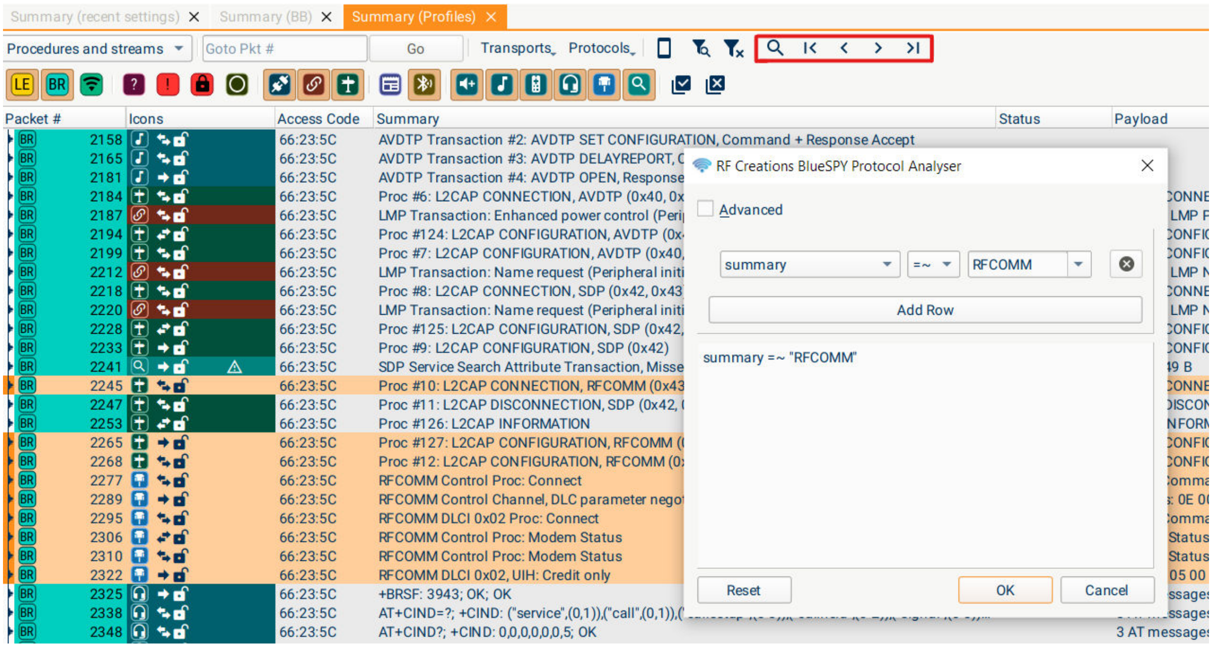
access_code == 66:23:5C

Add Row

access_code == "66:23:5C"

Reset OK Cancel

A similar interface can also be used to select packets matching a given search; the arrows to the right of the Search icon can then be used to step between the selected events.



Packet #	Access Code	Summary	Status	Payload
2158	66:23:5C	AVDTP Transaction #2: AVDTP SET CONFIGURATION, Command + Response Accept		
2165	66:23:5C	AVDTP Transaction #3: AVDTP DELAYREPORT, C		
2181	66:23:5C	AVDTP Transaction #4: AVDTP OPEN, Response		
2184	66:23:5C	Proc #6: L2CAP CONNECTION, AVDTP (0x40, 0x		CONNE
2187	66:23:5C	LMP Transaction: Enhanced power control (Peri		LMP P
2194	66:23:5C	Proc #124: L2CAP CONFIGURATION, AVDTP (0x		CONFIG
2199	66:23:5C	Proc #7: L2CAP CONFIGURATION, AVDTP (0x40,		CONFIG
2212	66:23:5C	LMP Transaction: Name request (Peripheral initi		LMP N
2218	66:23:5C	Proc #8: L2CAP CONNECTION, SDP (0x42, 0x43		CONNE
2220	66:23:5C	LMP Transaction: Name request (Peripheral initi		LMP N
2228	66:23:5C	Proc #125: L2CAP CONFIGURATION, SDP (0x42,		CONFIG
2233	66:23:5C	Proc #9: L2CAP CONFIGURATION, SDP (0x42)		CONFIG
2241	66:23:5C	SDP Service Search Attribute Transaction, Misse		49 B
2245	66:23:5C	Proc #10: L2CAP CONNECTION, RFCOMM (0x43		CONNE
2247	66:23:5C	Proc #11: L2CAP DISCONNECTION, SDP (0x42, (		DISCON
2253	66:23:5C	Proc #126: L2CAP INFORMATION		NFORM
2265	66:23:5C	Proc #127: L2CAP CONFIGURATION, RFCOMM (		CONFIG
2268	66:23:5C	Proc #12: L2CAP CONFIGURATION, RFCOMM (0x		CONFIG
2277	66:23:5C	RFCOMM Control Proc: Connect		Comma
2289	66:23:5C	RFCOMM Control Channel, DLC parameter nego		s: 0E 00
2295	66:23:5C	RFCOMM DLCI 0x02 Proc: Connect		Comma
2306	66:23:5C	RFCOMM Control Proc: Modem Status		Status
2310	66:23:5C	RFCOMM Control Proc: Modem Status		Status
2322	66:23:5C	RFCOMM DLCI 0x02, UIH: Credit only		05 00
2325	66:23:5C	+BRSF: 3943; OK; OK		ssages
2338	66:23:5C	AT+CIND=?; +CIND: ("service",(0,1)),("call",(0,1)),(		ssages
2348	66:23:5C	AT+CIND?; +CIND: 0,0,0,0,0,0,5; OK		3 AT messages

RF Creations BlueSPY Protocol Analyser

☐ Advanced

summary =~ RFCOMM

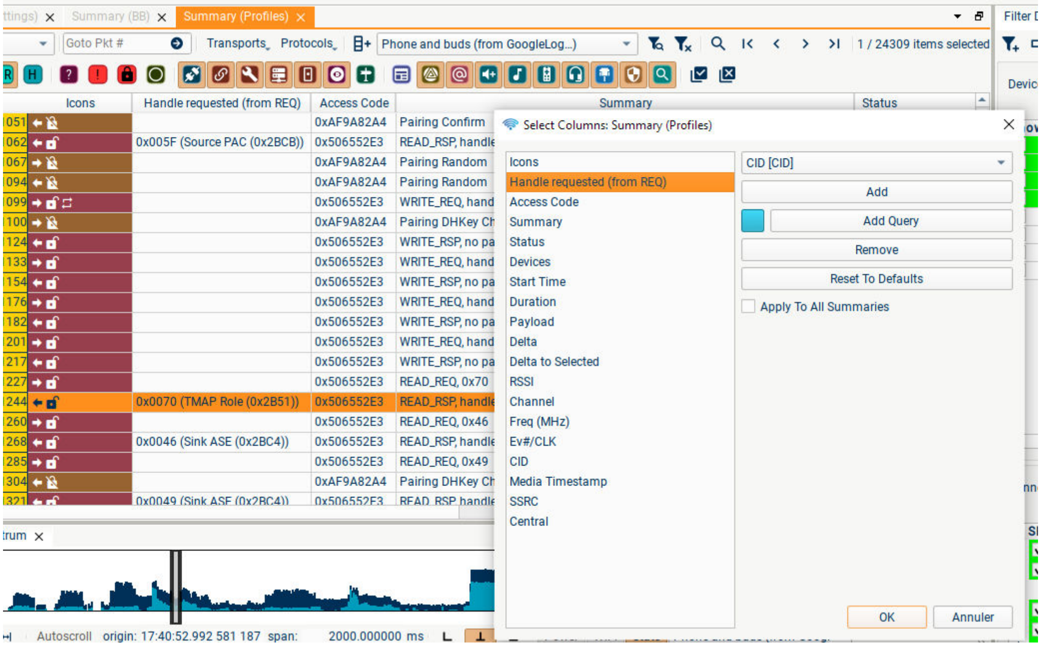
Add Row

summary =~ "RFCOMM"

Reset OK Cancel

10.2 Choosing and Creating columns

In each Summary tab, you can choose which columns to show, and add new ones, using the Select Columns dialog (launched using this button ▤+).



	Icons	Handle requested (from REQ)	Access Code	Summary	Status
051	←		0xAF9A82A4	Pairing Confirm	
062	←	0x005F (Source PAC (0x2BCB))	0x506552E3	READ_RSP, handle	
067	→		0xAF9A82A4	Pairing Random	
094	←		0xAF9A82A4	Pairing Random	
099	→		0x506552E3	WRITE_REQ, hand	
100	→		0xAF9A82A4	Pairing DHKey Ch	
124	←		0x506552E3	WRITE_RSP, no pa	
133	→		0x506552E3	WRITE_REQ, hand	
154	←		0x506552E3	WRITE_RSP, no pa	
176	→		0x506552E3	WRITE_REQ, hand	
182	←		0x506552E3	WRITE_RSP, no pa	
201	→		0x506552E3	WRITE_REQ, hand	
217	←		0x506552E3	WRITE_RSP, no pa	
227	→		0x506552E3	READ_REQ, 0x70	
244	←	0x0070 (TMAP Role (0x2B51))	0x506552E3	READ_RSP, handle	
260	→		0x506552E3	READ_REQ, 0x46	
268	←	0x0046 (Sink ASE (0x2BC4))	0x506552E3	READ_RSP, handle	
285	→		0x506552E3	READ_REQ, 0x49	
304	←		0xAF9A82A4	Pairing DHKey Ch	
321	←	0x0049 (Sink ASE (0x2BC4))	0x506552E3	READ_RSP, handle	

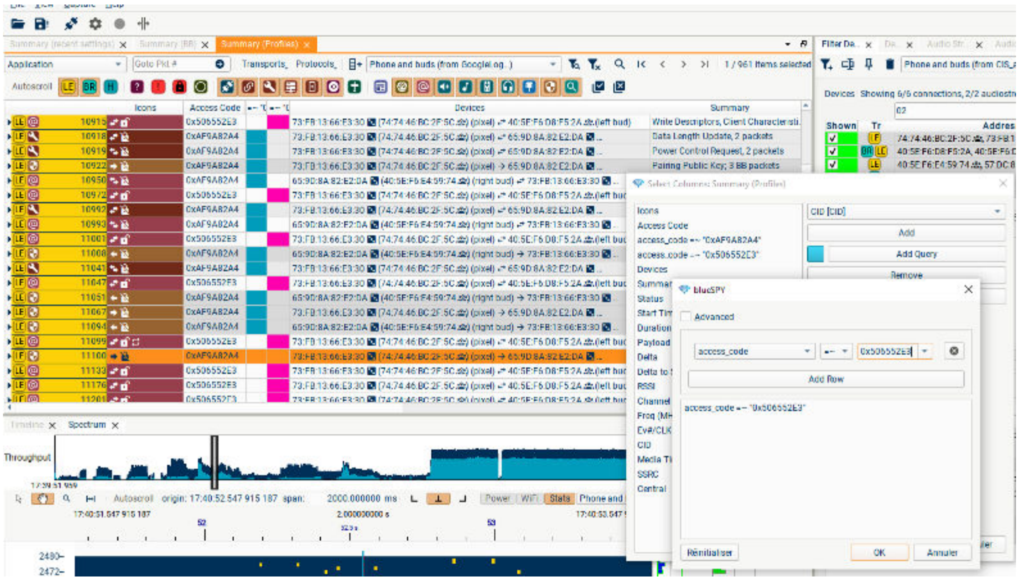
Select Columns: Summary (Profiles)

- Icons
- Handle requested (from REQ)
- Access Code
- Summary
- Status
- Devices
- Start Time
- Duration
- Payload
- Delta
- Delta to Selected
- RSSI
- Channel
- Freq (MHz)
- Ev#/CLK
- CID
- Media Timestamp
- SSRC
- Central

CID [CID]

There are three types of columns:

1. Columns using one of the documented queries
2. Columns created by clicking on a non-expandable field in a Details pane and dragging it into a Summary tab. These columns will mostly only show data when an identical field is present in another packet; so in the screenshot a column was created by dragging the "Handle requested" field from a ATT_READ_RSP details pane, and it is only showing information from other READ_RSP packets.
3. Columns showing a chosen colour when a particular query is true. For instance, when you have two similar connections to two earbuds, you can add a column highlighting which packets/transactions are on which Access Address:



Icons
Access Code
access_code == "0xAF9A82A4"
access_code == "0x506552E3"
Devices
Summary
Status
Start Tim
Duration
Payload
Delta
Delta to
RSSI
Channel
Freq (MH
Ev#/CLK
CID
Media Ti
SSRC
Central

CID [CID]

access_code == "0x506552E3"

10.3 GoTo

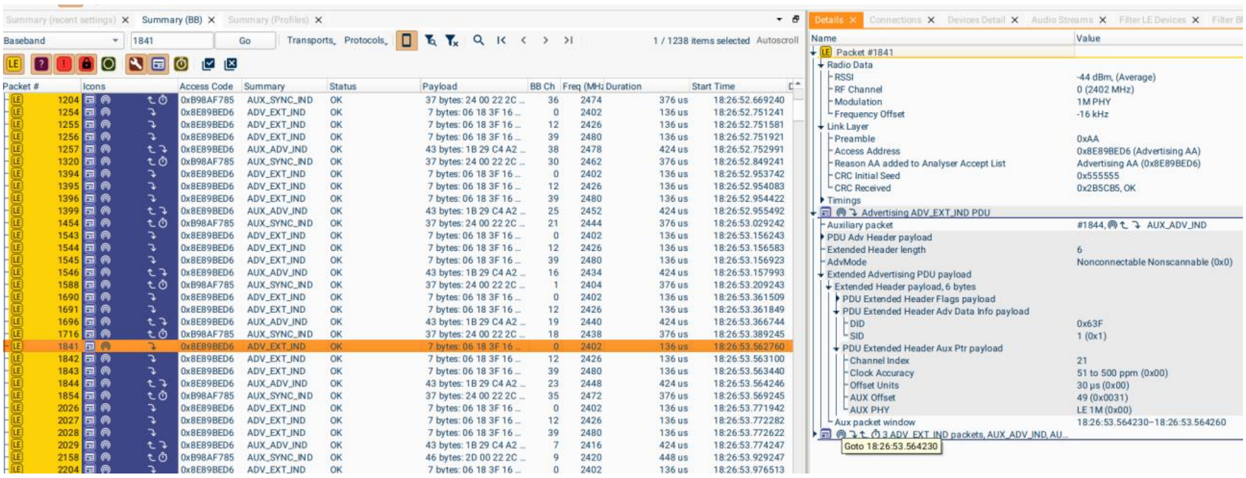
If you have previously found a packet of particular interest, you can use the GoTo box to navigate quickly to that packet using its packet number.

10.4 Multiple Summary tabs

Using the View menu, you can open extra Summary tabs; the filtering and aggregation options can be selected independently in these tabs, so you can e.g. keep one tab open showing baseband packets with no filtering, and another showing just the protocols you are interested in. These tabs can also be renamed by right-clicking in the tab.

10.5 Linked packets

If a baseband packet or a timestamp is referenced in the Details, Connections, or Devices tabs, double-clicking on the text will navigate to that packet (or to the first of that set of packets) in the Summary tab. For example, double-clicking on the "Aux packet window" field in this extended advertising packet will Goto 18:26:53.564230, and select packet #1841 which falls in that window.

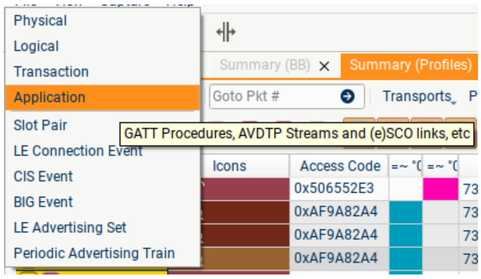


Packet #	Icons	Access Code	Summary	Status	Payload	BB Ch	Freq (MHz)	Duration	Start Time
LE 1204	↰ ⏱	0xB98AF785	AUX_SYNC_IND	OK	37 bytes: 24 00 22 2C ...	36	2474	376 us	18:26:52.669240
LE 1254	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	0	2402	136 us	18:26:52.751241
LE 1255	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	12	2426	136 us	18:26:52.751581
LE 1256	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	39	2480	136 us	18:26:52.751921
LE 1257	↰ ↴	0x8E89BED6	AUX_ADV_IND	OK	43 bytes: 1B 29 C4 A2 ...	38	2478	424 us	18:26:52.752991
LE 1320	↰ ⏱	0xB98AF785	AUX_SYNC_IND	OK	37 bytes: 24 00 22 2C ...	30	2462	376 us	18:26:52.849241
LE 1394	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	0	2402	136 us	18:26:52.953742
LE 1395	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	12	2426	136 us	18:26:52.954083
LE 1396	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	39	2480	136 us	18:26:52.954422
LE 1399	↰ ↴	0x8E89BED6	AUX_ADV_IND	OK	43 bytes: 1B 29 C4 A2 ...	25	2452	424 us	18:26:52.955492
LE 1454	↰ ⏱	0xB98AF785	AUX_SYNC_IND	OK	37 bytes: 24 00 22 2C ...	21	2444	376 us	18:26:53.029242
LE 1543	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	0	2402	136 us	18:26:53.156243
LE 1544	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	12	2426	136 us	18:26:53.156583
LE 1545	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	39	2480	136 us	18:26:53.156923
LE 1546	↰ ↴	0x8E89BED6	AUX_ADV_IND	OK	43 bytes: 1B 29 C4 A2 ...	16	2434	424 us	18:26:53.157993
LE 1588	↰ ⏱	0xB98AF785	AUX_SYNC_IND	OK	37 bytes: 24 00 22 2C ...	1	2404	376 us	18:26:53.209243
LE 1690	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	0	2402	136 us	18:26:53.361509
LE 1691	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	12	2426	136 us	18:26:53.361849
LE 1696	↰ ↴	0x8E89BED6	AUX_ADV_IND	OK	43 bytes: 1B 29 C4 A2 ...	19	2440	424 us	18:26:53.366744
LE 1716	↰ ⏱	0xB98AF785	AUX_SYNC_IND	OK	37 bytes: 24 00 22 2C ...	18	2438	376 us	18:26:53.389245
LE 1841	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	0	2402	136 us	18:26:53.562760
LE 1842	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	12	2426	136 us	18:26:53.563100
LE 1843	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	39	2480	136 us	18:26:53.563440
LE 1844	↰ ↴	0x8E89BED6	AUX_ADV_IND	OK	43 bytes: 1B 29 C4 A2 ...	23	2448	424 us	18:26:53.564246
LE 1854	↰ ⏱	0xB98AF785	AUX_SYNC_IND	OK	37 bytes: 24 00 22 2C ...	35	2472	376 us	18:26:53.569245
LE 2026	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	0	2402	136 us	18:26:53.771942
LE 2027	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	12	2426	136 us	18:26:53.772282
LE 2028	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	39	2480	136 us	18:26:53.772622
LE 2029	↰ ↴	0x8E89BED6	AUX_ADV_IND	OK	43 bytes: 1B 29 C4 A2 ...	7	2416	424 us	18:26:53.774247
LE 2158	↰ ⏱	0xB98AF785	AUX_SYNC_IND	OK	46 bytes: 2D 00 22 2C ...	9	2420	448 us	18:26:53.929247
LE 2204	↴	0x8E89BED6	ADV_EXT_IND	OK	7 bytes: 06 18 3F 16 ...	0	2402	136 us	18:26:53.976513

Name	Value
Packet #1841	
Radio Data	
RSSI	-44 dBm, (Average)
RF Channel	0 (2402 MHz)
Modulation	1M PHY
Frequency Offset	-16 kHz
Link Layer	
Preamble	0xAA
Access Address	0x8E89BED6 (Advertising AA)
Reason AA added to Analyser Accept List	Advertising AA (0x8E89BED6)
CRC Initial Seed	0x555555
CRC Received	0x2B5CB5, OK
Timings	
Advertising ADV_EXT_IND PDU	
Auxiliary packet	#1844, ↰ ↴ AUX_ADV_IND
PDU Adv Header payload	
Extended Header length	6
AdvMode	Nonconnectable Nonscannable (0x0)
Extended Advertising PDU payload	
Extended Header payload, 6 bytes	
PDU Extended Header Flags payload	
PDU Extended Header Adv Data Info payload	
DID	0x63F
SID	1 (0x1)
PDU Extended Header Aux Ptr payload	
Channel Index	21
Clock Accuracy	51 to 500 ppm (0x00)
Offset Units	30 µs (0x00)
AUX Offset	49 (0x0031)
AUX PHY	LE 1M (0x00)
Aux packet window	18:26:53.564230–18:26:53.564260
↴ ↰ ⏱ 3 ADV_EXT_IND packets, AUX_ADV_IND, AU...	

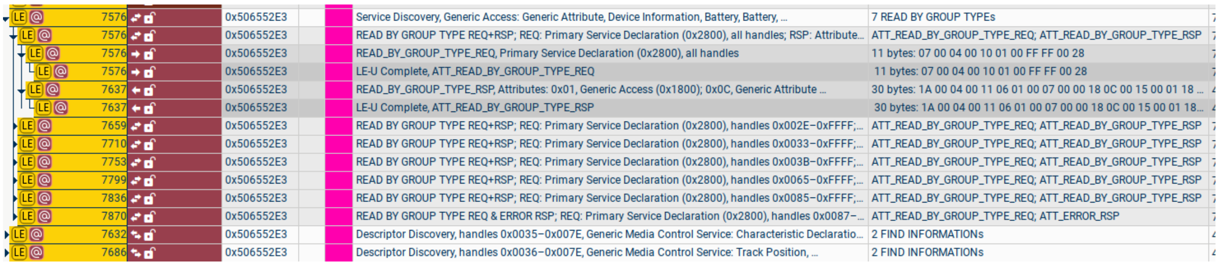
Goto 18:26:53.564230

10.6 Specific Views for different Protocol layers



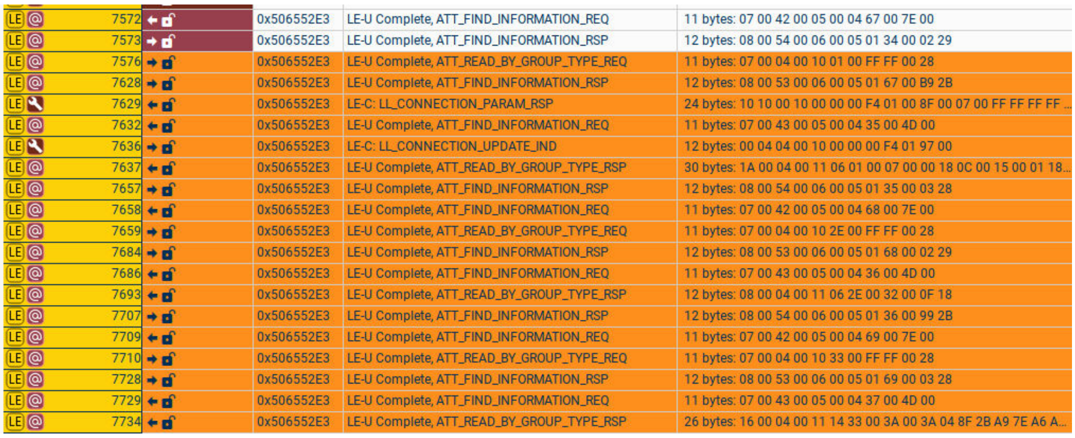
Several different Views are available in the Summary window, displaying different protocols and different types of event. Some of the Views are most useful for studying timing and NACKs/ReTXes in the lower layers, whereas others are better for following the flow of data in higher-level protocols.

The first four Views organise packets at increasing levels of abstraction/aggregation, collecting together packets that are part of e.g. a single L2CAP packet, a single Transaction involving multiple L2CAP packets, or a procedure consisting of a sequence of Transactions.



LE @	7576	⇄	0x506552E3	Service Discovery, Generic Access: Generic Attribute, Device Information, Battery, Battery, ...	7 READ BY GROUP TYPEs	7
LE @	7576	⇄	0x506552E3	READ BY GROUP TYPE REQ+RSP; REQ: Primary Service Declaration (0x2800), all handles; RSP: Attribute...	ATT_READ_BY_GROUP_TYPE_REQ; ATT_READ_BY_GROUP_TYPE_RSP	7
LE @	7576	→	0x506552E3	READ_BY_GROUP_TYPE_REQ, Primary Service Declaration (0x2800), all handles	11 bytes: 07 00 04 00 10 01 00 FF FF 00 28	7
LE @	7576	→	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_REQ	11 bytes: 07 00 04 00 10 01 00 FF FF 00 28	7
LE @	7637	←	0x506552E3	READ_BY_GROUP_TYPE_RSP, Attributes: 0x01, Generic Access (0x1800); 0x0C, Generic Attribute ...	30 bytes: 1A 00 04 00 11 06 01 00 07 00 00 18 0C 00 15 00 01 18 ...	4
LE @	7637	←	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_RSP	30 bytes: 1A 00 04 00 11 06 01 00 07 00 00 18 0C 00 15 00 01 18...	4
LE @	7659	⇄	0x506552E3	READ BY GROUP TYPE REQ+RSP; REQ: Primary Service Declaration (0x2800), handles 0x002E–0xFFFF;...	ATT_READ_BY_GROUP_TYPE_REQ; ATT_READ_BY_GROUP_TYPE_RSP	7
LE @	7710	⇄	0x506552E3	READ BY GROUP TYPE REQ+RSP; REQ: Primary Service Declaration (0x2800), handles 0x0033–0xFFFF;...	ATT_READ_BY_GROUP_TYPE_REQ; ATT_READ_BY_GROUP_TYPE_RSP	7
LE @	7753	⇄	0x506552E3	READ BY GROUP TYPE REQ+RSP; REQ: Primary Service Declaration (0x2800), handles 0x003B–0xFFFF;...	ATT_READ_BY_GROUP_TYPE_REQ; ATT_READ_BY_GROUP_TYPE_RSP	7
LE @	7799	⇄	0x506552E3	READ BY GROUP TYPE REQ+RSP; REQ: Primary Service Declaration (0x2800), handles 0x0065–0xFFFF;...	ATT_READ_BY_GROUP_TYPE_REQ; ATT_READ_BY_GROUP_TYPE_RSP	7
LE @	7836	⇄	0x506552E3	READ BY GROUP TYPE REQ+RSP; REQ: Primary Service Declaration (0x2800), handles 0x0085–0xFFFF;...	ATT_READ_BY_GROUP_TYPE_REQ; ATT_READ_BY_GROUP_TYPE_RSP	7
LE @	7870	⇄	0x506552E3	READ BY GROUP TYPE REQ & ERROR RSP; REQ: Primary Service Declaration (0x2800), handles 0x0087–...	ATT_READ_BY_GROUP_TYPE_REQ; ATT_ERROR_RSP	7
LE @	7632	⇆	0x506552E3	Descriptor Discovery, handles 0x0035–0x007E, Generic Media Control Service: Characteristic Declaratio...	2 FIND INFORMATIONs	4
LE @	7686	⇆	0x506552E3	Descriptor Discovery, handles 0x0036–0x007E, Generic Media Control Service: Track Position, ...	2 FIND INFORMATIONs	4

The GATT Procedure above is shown as displayed using the "Application" View, in which upto 3 levels of aggregation are used. The Procedure consists of seven READ_BY_GROUP_TYPE ATT Transactions, each of which contains two L2CAP packets, each of which is sent in a single LL packet.



LE @	7572	←	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_REQ	11 bytes: 07 00 42 00 05 00 04 67 00 7E 00
LE @	7573	→	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_RSP	12 bytes: 08 00 54 00 06 00 05 01 34 00 02 29
LE @	7576	→	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_REQ	11 bytes: 07 00 04 00 10 01 00 FF FF 00 28
LE @	7628	→	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_RSP	12 bytes: 08 00 53 00 06 00 05 01 67 00 B9 2B
LE 🔧	7629	←	0x506552E3	LE-C: LL_CONNECTION_PARAM_RSP	24 bytes: 10 10 00 10 00 00 00 F4 01 00 8F 00 07 00 FF FF FF FF ...
LE @	7632	←	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_REQ	11 bytes: 07 00 43 00 05 00 04 35 00 4D 00
LE 🔧	7636	→	0x506552E3	LE-C: LL_CONNECTION_UPDATE_IND	12 bytes: 00 04 04 00 10 00 00 00 F4 01 97 00
LE @	7637	←	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_RSP	30 bytes: 1A 00 04 00 11 06 01 00 07 00 00 18 0C 00 15 00 01 18...
LE @	7657	→	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_RSP	12 bytes: 08 00 54 00 06 00 05 01 35 00 03 28
LE @	7658	←	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_REQ	11 bytes: 07 00 42 00 05 00 04 68 00 7E 00
LE @	7659	→	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_REQ	11 bytes: 07 00 04 00 10 2E 00 FF FF 00 28
LE @	7684	→	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_RSP	12 bytes: 08 00 53 00 06 00 05 01 68 00 02 29
LE @	7686	←	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_REQ	11 bytes: 07 00 43 00 05 00 04 36 00 4D 00
LE @	7693	←	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_RSP	12 bytes: 08 00 04 00 11 06 2E 00 32 00 0F 18
LE @	7707	→	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_RSP	12 bytes: 08 00 54 00 06 00 05 01 36 00 99 2B
LE @	7709	←	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_REQ	11 bytes: 07 00 42 00 05 00 04 69 00 7E 00
LE @	7710	→	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_REQ	11 bytes: 07 00 04 00 10 33 00 FF FF 00 28
LE @	7728	→	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_RSP	12 bytes: 08 00 53 00 06 00 05 01 69 00 03 28
LE @	7729	←	0x506552E3	LE-U Complete, ATT_FIND_INFORMATION_REQ	11 bytes: 07 00 43 00 05 00 04 37 00 4D 00
LE @	7734	←	0x506552E3	LE-U Complete, ATT_READ_BY_GROUP_TYPE_RSP	26 bytes: 16 00 04 00 11 14 33 00 3A 00 3A 04 8F 2B A9 7E A6 A...

Viewing the same procedure in the "Physical" View shows the LE LL layer packets in sequence (Empty packets have been hidden here); here we can see that another GATT procedure (using the ATT_FIND_INFORMATION… packets) and some LL Control packets are mixed in with this procedure.

The "Logical" and "Transaction" Views use intermediate levels of aggregation; the first collects together segmented packets (e.g. a complete L2CAP PDU), and the second collects together sets of these packets.

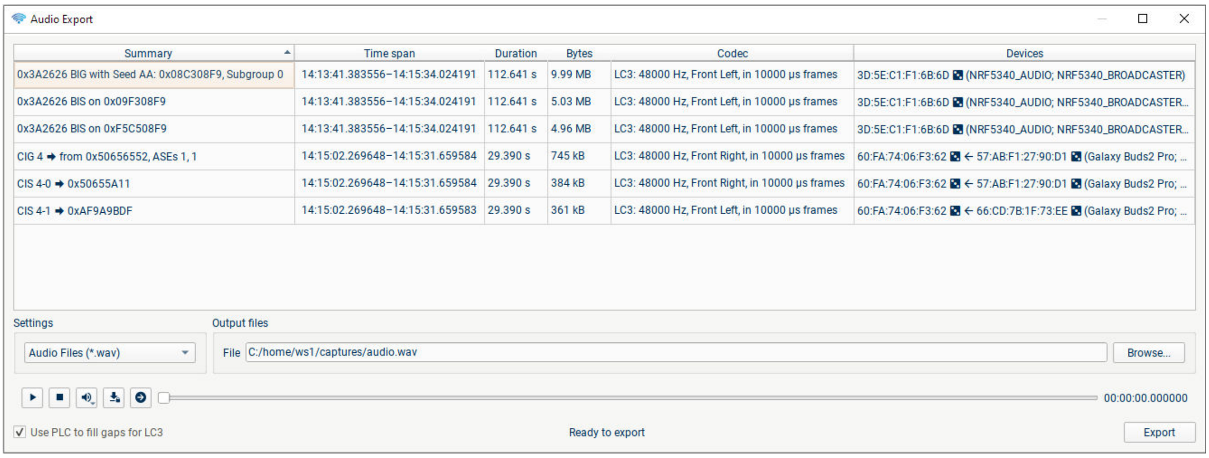
The four Views showing BR/EDR Slot Pairs, LE Connection Events, CIS and BIG Events display limited information about the content of the packets sent, and instead focus on the timings of events, and any missing packets/NACKs/ReTXes.

The LE Advertising Sets View collects together Advertising Events sent by one device at regular intervals. The lower events in this View are usually substantially out of order, as Advertising Sets will often overlap and can be very long.

The Periodic Advertising Train View shows only periodic advertising packets, and not the legacy/extended packets pointing to them. This is particularly useful for analysing Periodic Advertising with Responses, where the View shows the PAwR Events and Subevents.

If no packets are shown when a View is selected, check that there are no Filters/Searches hiding the relevant packets. For example, displaying only BR/EDR packets while on the LE Connection Events View will result in a blank screen!

11 AUDIO EXPORT



Summary	Time span	Duration	Bytes	Codec	Devices
0x3A2626 BIG with Seed AA: 0x08C308F9, Subgroup 0	14:13:41.383556–14:15:34.024191	112.641 s	9.99 MB	LC3: 48000 Hz, Front Left, in 10000 µs frames	3D:5E:C1:F1:6B:6D (NRF5340_AUDIO; NRF5340_BROADCASTER)
0x3A2626 BIS on 0x09F308F9	14:13:41.383556–14:15:34.024191	112.641 s	5.03 MB	LC3: 48000 Hz, Front Left, in 10000 µs frames	3D:5E:C1:F1:6B:6D (NRF5340_AUDIO; NRF5340_BROADCASTER...
0x3A2626 BIS on 0xF5C508F9	14:13:41.383556–14:15:34.024191	112.641 s	4.96 MB	LC3: 48000 Hz, Front Left, in 10000 µs frames	3D:5E:C1:F1:6B:6D (NRF5340_AUDIO; NRF5340_BROADCASTER...
CIG 4 → from 0x50656552, ASEs 1, 1	14:15:02.269648–14:15:31.659584	29.390 s	745 kB	LC3: 48000 Hz, Front Right, in 10000 µs frames	60:FA:74:06:F3:62 ← 57:AB:F1:27:90:D1 (Galaxy Buds2 Pro; ...
CIS 4-0 → 0x50655A11	14:15:02.269648–14:15:31.659584	29.390 s	384 kB	LC3: 48000 Hz, Front Right, in 10000 µs frames	60:FA:74:06:F3:62 ← 57:AB:F1:27:90:D1 (Galaxy Buds2 Pro; ...
CIS 4-1 → 0xAF9A9BDF	14:15:02.269648–14:15:31.659583	29.390 s	361 kB	LC3: 48000 Hz, Front Left, in 10000 µs frames	60:FA:74:06:F3:62 ← 66:CD:7B:1F:73:EE (Galaxy Buds2 Pro; ...

Settings: Audio Files (*.wav)

Output files: File C:/home/ws1/captures/audio.wav

☑ Use PLC to fill gaps for LC3

Ready to export

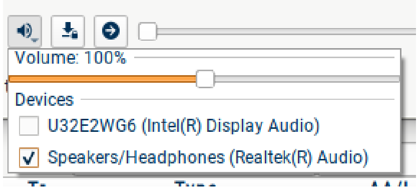
The Audio Export tab allows you to export captured audio to a file, and to play it back live or at a later point. All audio streams that are in the capture and we have some chance of decoding (i.e. we have decrypted them, or they were sent unencrypted) are present, including those for which we are missing some audio parameters (e.g. sample rate) or those using an unsupported or unknown codec; so not all of the rows can necessarily be decoded for playback. For streams using LC3 which have unknown, invalid or incorrect audio parameters, you can right-click on the stream to open the "Configure LC3 Codec" dialog. Here you can either override the parameters manually, or use the "auto-detect" functionality to determine likely parameters.

The table can be sorted by any of the columns; the first number printed in the Summary column for BIGes and BISes is the Broadcast Code, to make it easier to find a broadcast with a known Code.

For playback or export of LC3 audio, you have the option to either use the PLC algorithm to fill gaps where packets were missed, or to have an appropriate number of zeroes inserted in the stream.

11.1 PLAYBACK

To play some decoded audio, select a row, select an output (headphones, laptop speakers etc) and press play. The audio will start playing from the start of the stream; to hear it (approximately) live, press play and then drag the play position slider all the way to the right.


Volume: 100%

Devices

□ U32E2WG6 (Intel(R) Display Audio)

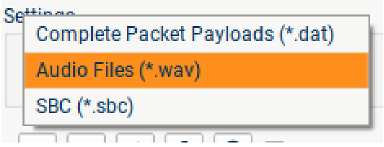
☑ Speakers/Headphones (Realtek(R) Audio)

Multichannel streams sent over separate transports (e.g. multiple CISes carrying stereo, or multiple BISes) can be played either individually by selecting the CIS/BIS row, or with both/all channels by selecting the CIG or BIG row.

You can choose to have the Timeline and Summary scroll to remain synchronised with the audio you are hearing, using the Autoscroll button . This works smoothly when the selected Summary is in "Physical" mode and all of the audio packets are visible; otherwise, the autoscroll will be jerky as it jumps between larger chunks of audio in the Summary.

You can also click to go to the packet matching the current audio position, e.g. when the playback is paused.

11.2 Export


Settings
Complete Packet Payloads (*.dat)
Audio Files (*.wav)
SBC (*.sbc)

Audio streams can be exported either to a WAV file as decoded audio, or to binary files containing the original (decrypted) bytes from the air packets for analysis in other tools (e.g. reference decoders). In addition, for A2DP streams the binary export can either include the AVDTP header ("Complete Packet Payloads"), or only the audio payload; in the case of SBC, this results in a *.sbc file which can be played by standard audio players.

12 Spectrum and Timeline tabs

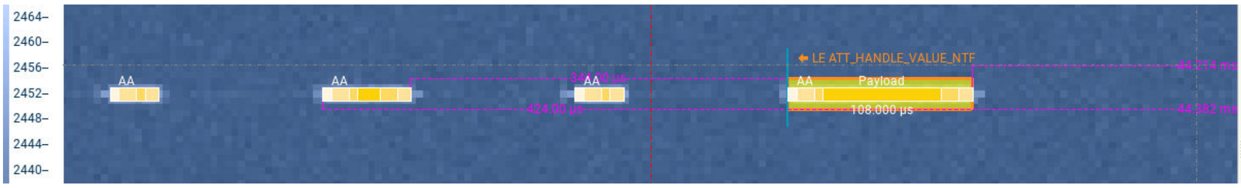
The Spectrum and Timeline tabs have much in common, and take two different approaches to a common goal: Displaying received packets sequentially in time.

In the Timeline tab the packets are grouped vertically by their sender device.

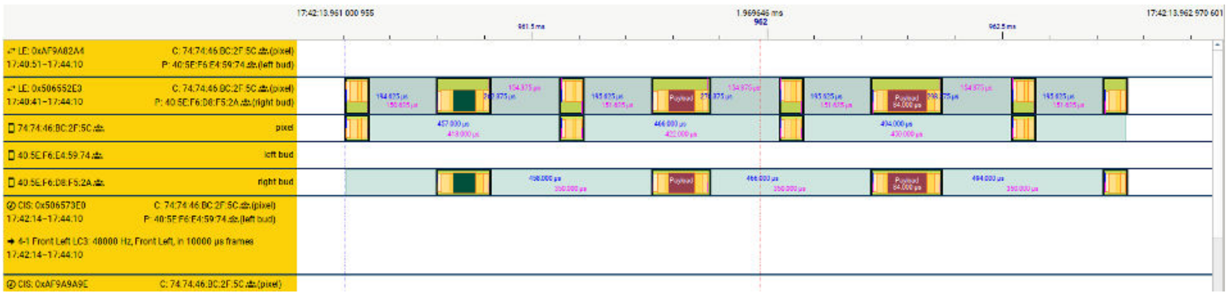
In the Spectrum tab, the packets are plotted according to frequency and time. In addition, the grey background colouring represents the power detected in the spectrum at that time and frequency.

In both tabs, time gaps to earlier/later packets and events are shown. More detail is shown in the Spectrum tab than in the Timeline, but the gaps displayed in the Timeline (if "Show Gaps" is enabled) will always represent gaps between events from the same Device/Connection.

Time deltas are drawn from packet start to packet start; when space allows, the gap between the end of a packet and the following start is also shown.

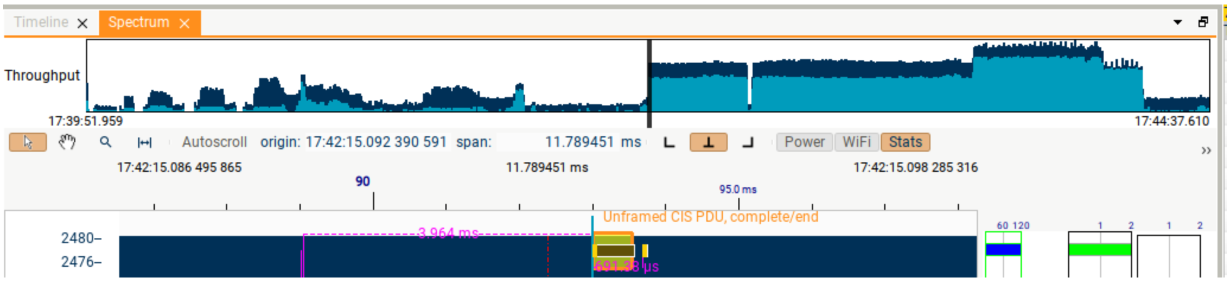


Spectrum

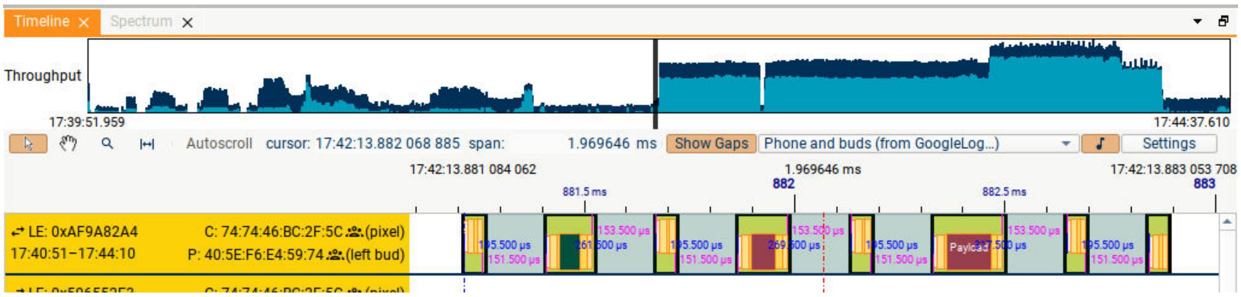

17:42:13.961 000 956
961.5 ms
1.969646 ms
962
962.5 ms
17:42:13.962 970 601
LE: 0xAF9A82A4 17:40:51–17:44:10
C: 74:74:46:BC:2F:5C (pixel)
P: 40:5E:F6:E4:59:74 (left bud)
LE: 0x506552E3 17:40:41–17:44:10
C: 74:74:46:BC:2F:5C (pixel)
P: 40:5E:F6:D8:F5:2A (right bud)
74:74:46:BC:2F:5C pixel
40:5E:F6:E4:59:74 left bud
40:5E:F6:D8:F5:2A right bud
CIS: 0x506573E0 17:42:14–17:44:10
C: 74:74:46:BC:2F:5C (pixel)
P: 40:5E:F6:E4:59:74 (left bud)
4-1 Front Left LC3: 48000 Hz, Front Left, in 10000 µs frames 17:42:14–17:44:10
CIS: 0xAF9A9A9E
C: 74:74:46:BC:2F:5C (pixel)
Payload
457.000 µs
418.000 µs
466.000 µs
422.000 µs
494.000 µs
430.000 µs

Timeline

12.1 Common Interface



Spectrum



Timeline

12.1.1 Filtering

In both the Spectrum and Timeline tabs, the filtering applied corresponds to the filtering in the currently active (last clicked-on) Summary tab, combined with a device filter of your choice, or no device filtering. So to see all packets in the Spectrum or Timeline, clear all filtering in the active Summary tab, and select No Device Filtering in the dropdown in the Spectrum/Timeline. In addition, in the Timeline you have the option to display or hide decoded audio for any suitable audiostreams which are shown.

12.1.2 Mouse control

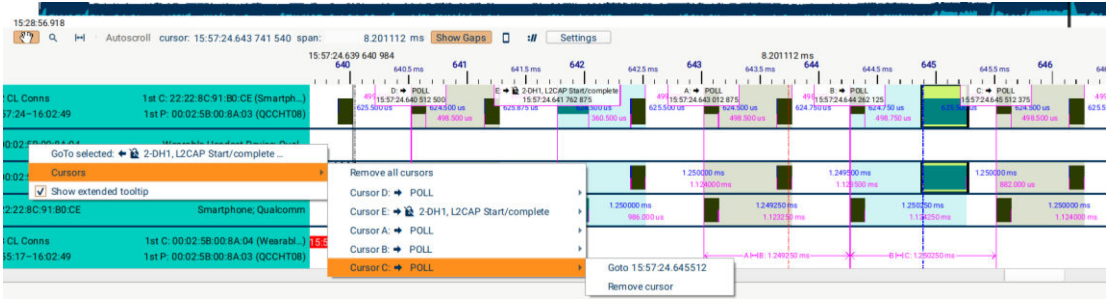
The mouse can be used in one of four modes: Pointer, Pan, Zoom and Measure. These can be selected using the buttons in the top left of the tab. The Pan, Zoom, and Measure modes can also be enabled temporarily by holding down Shift, Control or Alt respectively.

In all modes other than Zoom mode, the mouse scrollwheel can be used to:

a) scroll vertically in the Timeline view, when the mouse is over the labels at the left-hand side.
b) scroll horizontally in the rest of the Timeline view and in the Spectrum view.

Details of the four modes:

- In Pan mode, the display can be dragged left and right.
- In Zoom mode:
  - The scroll wheel zooms in and out. There are two modes for zooming, to match conflicting customer requests; zooming in and out will either zoom to the mouse pointer, or zoom while leaving the centre of the time window fixed. Choose which mode you want using the context menu in the Timeline/Spectrum.
  - A region can be selected by clicking and dragging, to zoom to that time region.
- In Pointer mode, clicking in the Spectrum/Timeline display will select the packet/event closest to the pointer (including selecting it in the Summary and Details tabs).
- In Measure mode, pairs of packets can have cursors added to them; the time difference between cursors is shown, and cursors can be removed or jumped to using the context menu.


15:28:56.918
Autoscroll
cursor: 15:57:24.643 741 540
span:
8.201112 ms
Show Gaps
Settings
15:57:24.639 640 984
8.201112 ms
CL Conns
1st C: 22:22:8C:91:BC:CE (Smartph...)
1st P: 00:02:5B:00:8A:03 (QCCHT08)
GoTo selected: 2-DH1, L2CAP Start/complete ...
Cursors
☑ Show extended tooltip
Smartphone; Qualcomm
1st C: 00:02:5B:00:8A:04 (Wearabl...)
1st P: 00:02:5B:00:8A:03 (QCCHT08)
Remove all cursors
Cursor D: POLL
Cursor E: 2-DH1, L2CAP Start/complete
Cursor A: POLL
Cursor B: POLL
Cursor C: POLL
Goto 15:57:24.645512
Remove cursor

In the Spectrum tab, when an event is selected, time gaps to the previous/next event of the same time will be displayed if the zoom level allows. If no time gaps appear, try zooming in! A selected event will also be selected in the Timeline and Summary tabs, unless you have used the filtering controls to hide it in those tabs.

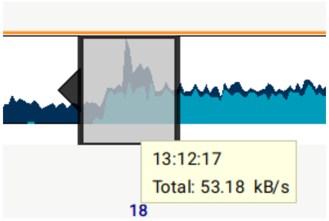
12.1.3 Keyboard control

The arrow keys can also be used to navigate; Up and Down zoom in and out, Left and Right jump earlier and later.

In the Spectrum tab, the A and Z keys increase and decrease the intensity of the Spectrum Power plot. The colourbar at the left-hand side of the tab shows how the grayscale corresponds to power in dBm.

12.1.4 Throughput plot

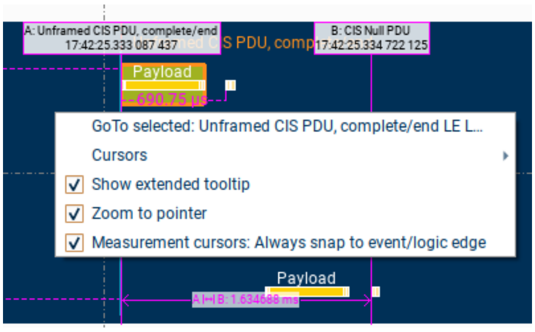
The throughput plot at the top of each tab shows the whole capture, and illustrates two estimates of Bluetooth traffic rate; one (darker blue) including all packet types, and one excluding Advertising packet and empty packets (LE Empty, NULL/POLLs etc). The black window shows the current time-range shown in the main Spectrum/Timeline area, and this window can be dragged (select the middle of the window) or extended/shrunk (select the arrows which appear to the left and right of the window when you hover).



12.1.5 Settings

Right-click in the rows of the Timeline, or anywhere in the Spectrum to see the context menu. There are several options to control how the GUI behaves:

1. "Zoom to pointer" controls the behaviour of the scrollwheel when zooming, as described above.
2. Measurement cursors: as you move your mouse, these can either snap to a packet edge only when close to a packet, or always snap to a packet/other event (e.g. logic line edges).

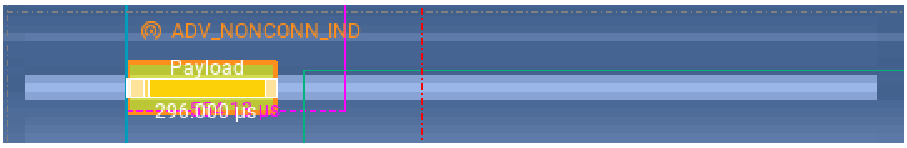


12.2 Spectrum

In addition to the Power plotted in grayscale, the text in the top-right of the tab displays the power that was measured at the time and frequency corresponding to the pointer position. This text also records the attenuation used by the Moreph's amplifier during capture; an unexpectedly large attenuation value could indicate interference.

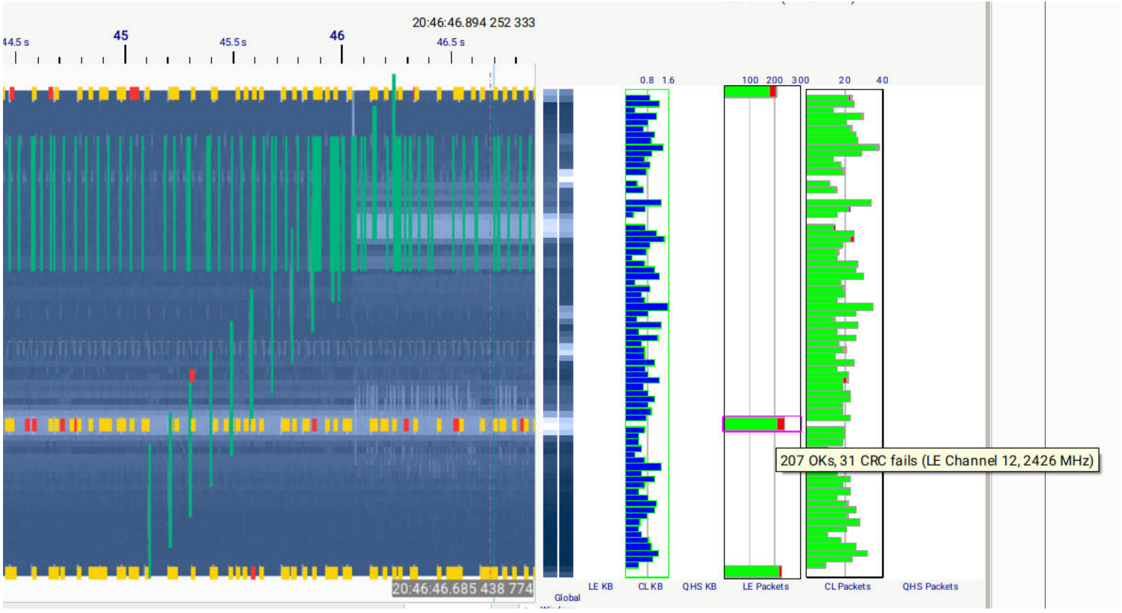
Two important notes:

1. The power displayed in a time interval represents the peak power measured during that interval. This can look slightly confusing when using low-resolution spectrum capture, as a packet can appear as a short blob in the middle of a long grey rectangle.
2. The lengths of the rectangles representing the packet header and other packet segments are:
   a. accurate at high zoom
   b. pictorial at low zoom; the rectangles are wider than the actual time interval in order to prevent them becoming invisibly narrow.



A 376 µs packet marooned in a 2 ms peak-hold window.

12.2.1 Statistics display

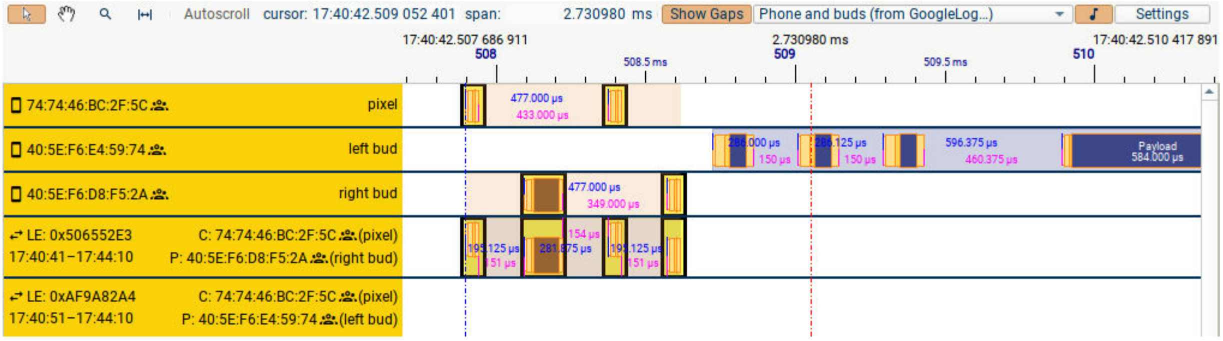


207 OKs, 31 CRC fails (LE Channel 12, 2426 MHz)

Per-channel statistics showing average power, throughput, and packet success (rates of ReTXes, CRC or dewhitening fails, and Rejections) can be enabled using the "Stats" button. To see details of the values represented by each bar in the bar chart, hover over the bar.

12.3 Timeline

The shorter rectangles in solid colours represent on-air packets. The larger translucent rectangles represent groupings of those packets (LE Advertising Events and Connection Events, BR/EDR Slot Pairs, CIS/BIS events). The selected packet or packets are highlighted with a yellow box; as in the Spectrum tab, clicking on a packet will cause it to be selected in the other tabs unless it has been hidden there.



cursor: 17:40:42.509 052 401 span: 2.730980 ms

Phone and buds (from GoogleLog...)

74:74:46:BC:2F:5C pixel

40:5E:F6:E4:59:74 left bud

40:5E:F6:D8:F5:2A right bud

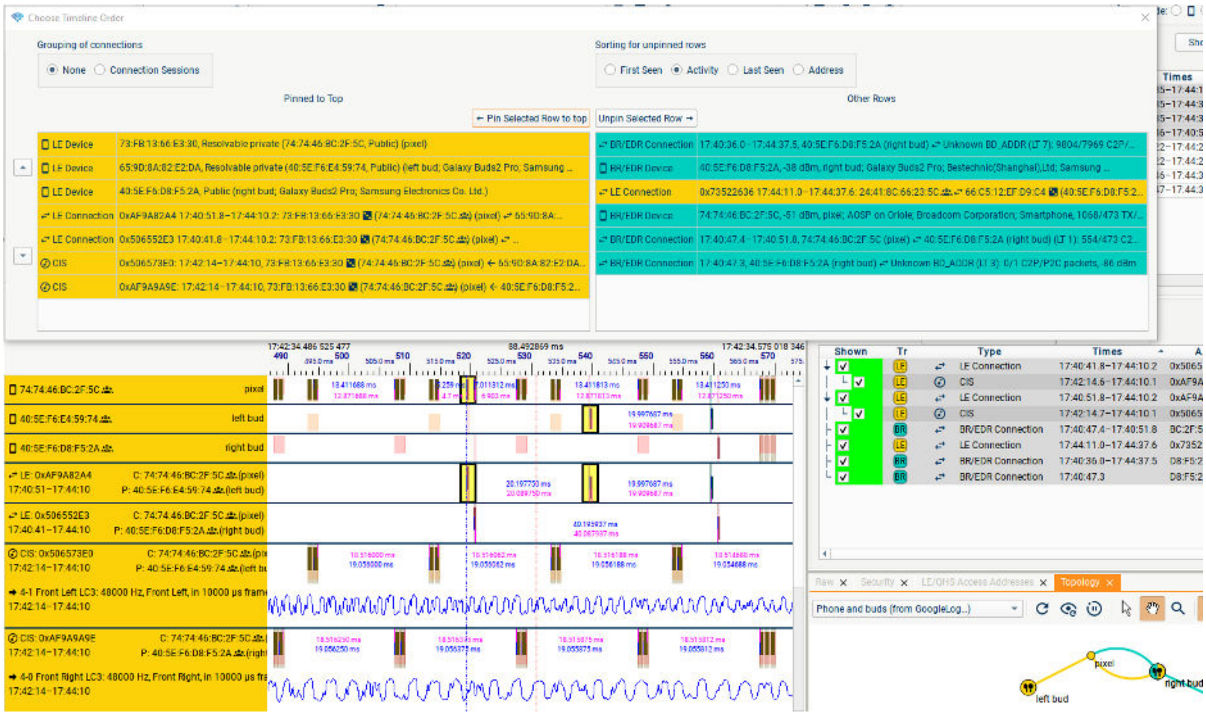
LE: 0x506552E3 17:40:41–17:44:10 C: 74:74:46:BC:2F:5C (pixel) P: 40:5E:F6:D8:F5:2A (right bud)

LE: 0xAF9A82A4 17:40:51–17:44:10 C: 74:74:46:BC:2F:5C (pixel) P: 40:5E:F6:E4:59:74 (left bud)

The rows shown in the Timeline are determined by the devices, connections, and audio streams you have enabled in the chosen device filter. If you find that the packets from your DUTs are separated out into too many rows and it's hard to see everything you need, there are two ways to reduce the number of rows:

1. Use the "Connections" mode in the device filter and deselect some of the connections or audio streams. This is helpful when your DUT has various connections to uninteresting 3rd devices and you are only interested in e.g. the phone <-> earbud connections.
2. Use the "Connection Sessions" mode (selected in the Timeline "Settings" dialog). This collapses all connections between a particular pair of devices onto one row, so is particularly helpful if your DUTs are repeatedly disconnecting and reconnecting.

You can control the order the rows are displayed in using the “Settings” popup:



Choose Timeline Order

Grouping of connections: ● None ○ Connection Sessions

Sorting for unpinned rows: ○ First Seen ● Activity ○ Last Seen ○ Address

Pinned to Top

← Pin Selected Row to top

LE Device	73:FB:13:66:E3:30, Resolvable private (74:74:46:BC:2F:5C, Public) (pixel)
LE Device	65:9D:8A:82:E2:DA, Resolvable private (40:5E:F6:E4:59:74, Public) (left bud; Galaxy Buds2 Pro; Samsung ...
LE Device	40:5E:F6:D8:F5:2A, Public (right bud; Galaxy Buds2 Pro; Samsung Electronics Co. Ltd.)
LE Connection	0xAF9A82A4 17:40:51.8–17:44:10.2: 73:FB:13:66:E3:30 (74:74:46:BC:2F:5C) (pixel) 65:9D:8A:...
LE Connection	0x506552E3 17:40:41.8–17:44:10.2: 73:FB:13:66:E3:30 (74:74:46:BC:2F:5C) (pixel) ...
CIS	0x506573E0: 17:42:14–17:44:10, 73:FB:13:66:E3:30 (74:74:46:BC:2F:5C) (pixel) ← 65:9D:8A:82:E2:DA...
CIS	0xAF9A9A9E: 17:42:14–17:44:10, 73:FB:13:66:E3:30 (74:74:46:BC:2F:5C) (pixel) ← 40:5E:F6:D8:F5:2...

Other Rows

Unpin Selected Row →

BR/EDR Connection	17:40:36.0–17:44:37.5, 40:5E:F6:D8:F5:2A (right bud) Unknown BD_ADDR (LT 7): 9804/7969 C2P/...
BR/EDR Device	40:5E:F6:D8:F5:2A, -38 dBm, right bud; Galaxy Buds2 Pro; Bestechnic(Shanghai),Ltd; Samsung ...
LE Connection	0x73522636 17:44:11.0–17:44:37.6: 24:41:8C:66:23:5C 66:C5:12:EF:D9:C4 (40:5E:F6:D8:F5:2...
BR/EDR Device	74:74:46:BC:2F:5C, -51 dBm, pixel; AOSP on Oriole; Broadcom Corporation; Smartphone, 1068/473 TX/...
BR/EDR Connection	17:40:47.4–17:40:51.8, 74:74:46:BC:2F:5C (pixel) 40:5E:F6:D8:F5:2A (right bud) (LT 1): 554/473 C2...
BR/EDR Connection	17:40:47.3, 40:5E:F6:D8:F5:2A (right bud) Unknown BD_ADDR (LT 3): 0/1 C2P/P2C packets, -86 dBm

74:74:46:BC:2F:5C pixel

40:5E:F6:E4:59:74 left bud

40:5E:F6:D8:F5:2A right bud

LE: 0xAF9A82A4 17:40:51–17:44:10 C: 74:74:46:BC:2F:5C (pixel) P: 40:5E:F6:E4:59:74 (left bud)

LE: 0x506552E3 17:40:41–17:44:10 C: 74:74:46:BC:2F:5C (pixel) P: 40:5E:F6:D8:F5:2A (right bud)

CIS: 0x506573E0 17:42:14–17:44:10 C: 74:74:46:BC:2F:5C (pi P: 40:5E:F6:E4:59:74 (left bu

→ 4-1 Front Left LC3: 48000 Hz, Front Left, in 10000 µs fram 17:42:14–17:44:10

CIS: 0xAF9A9A9E 17:42:14–17:44:10 C: 74:74:46:BC:2F:5C P: 40:5E:F6:D8:F5:2A (righ

→ 4-0 Front Right LC3: 48000 Hz, Front Right, in 10000 µs fra 17:42:14–17:44:10

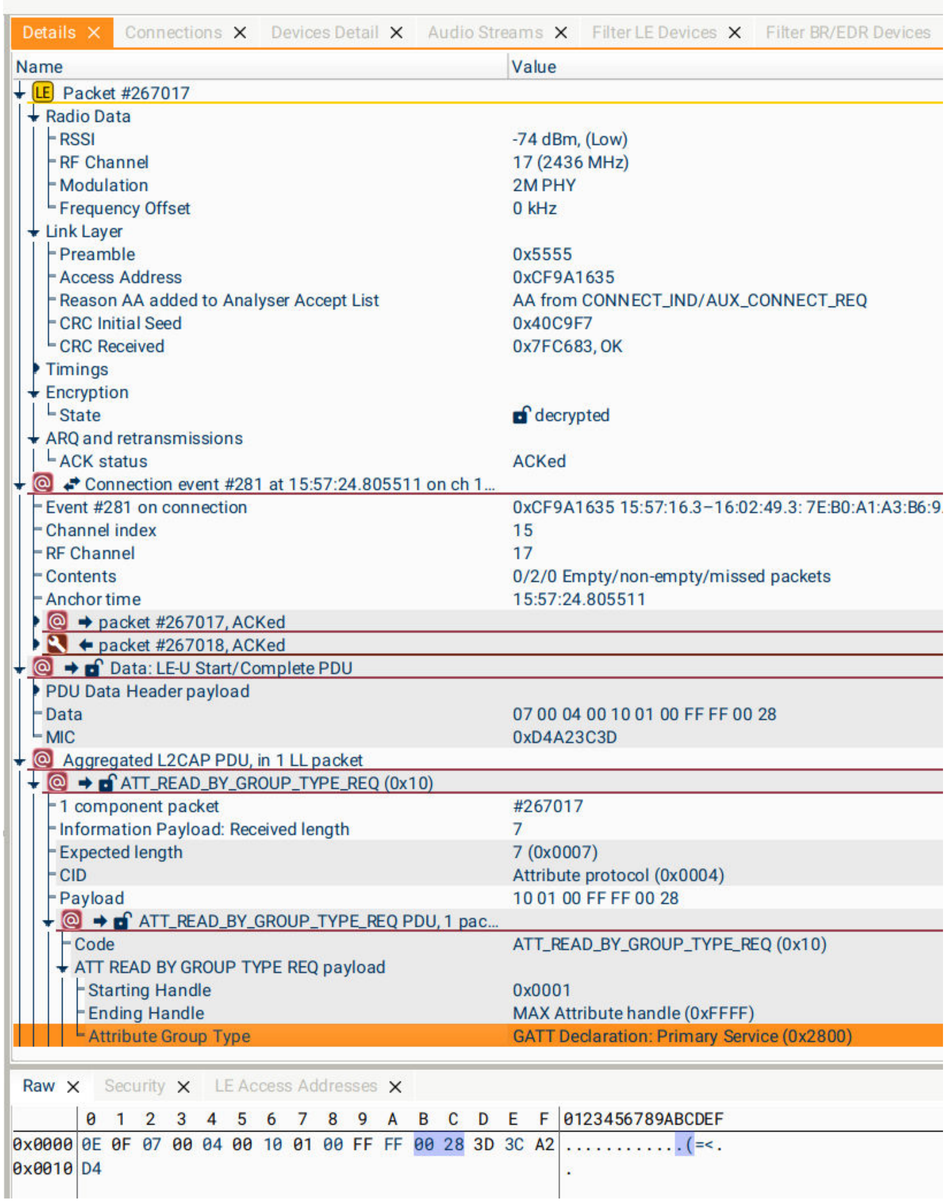
Shown	Tr	Type	Times	A
☑	LE	LE Connection	17:40:41.8–17:44:10.2	0x5065
☑	LE	CIS	17:42:14.6–17:44:10.1	0xAF9A
☑	LE	LE Connection	17:40:51.8–17:44:10.2	0xAF9A
☑	LE	CIS	17:42:14.7–17:44:10.1	0x5065
☑	BR	BR/EDR Connection	17:40:47.4–17:40:51.8	BC:2F:5
☑	LE	LE Connection	17:44:11.0–17:44:37.6	0x7352
☑	BR	BR/EDR Connection	17:40:36.0–17:44:37.5	D8:F5:2
☑	BR	BR/EDR Connection	17:40:47.3	D8:F5:2

You can “pin” the rows you’re interested in by moving them into the left-hand column (double-click on an item to move it left or right), and choose the order of the pinned rows; these will be displayed at the top of the Timeline view. All other rows, including rows created when new devices/connections are seen, will appear below these, sorted according to your choice from “First Seen”, “Activity” (e.g. number of packets), “Last Seen”, or “Address”.

13 Other tabs

13.1 Details & Raw

These tabs provide the most detailed look at a packet or higher-layer event. The Details tab is populated whenever an event is selected in the Summary/Timeline/Spectrum tabs, and the Raw tab is also populated when the event has "a payload".



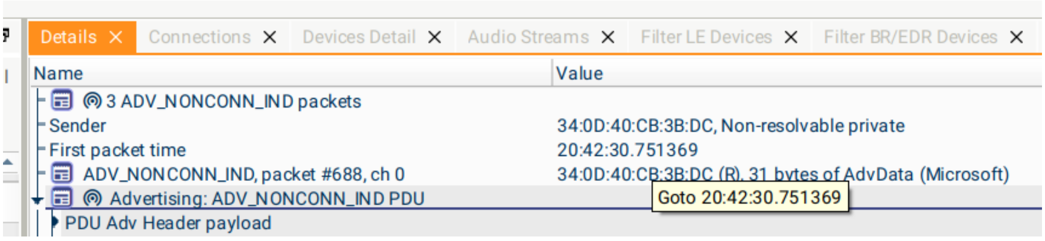
```
Name                                          Value
LE Packet #267017
  Radio Data
    RSSI                                      -74 dBm, (Low)
    RF Channel                                17 (2436 MHz)
    Modulation                                2M PHY
    Frequency Offset                          0 kHz
  Link Layer
    Preamble                                  0x5555
    Access Address                            0xCF9A1635
    Reason AA added to Analyser Accept List   AA from CONNECT_IND/AUX_CONNECT_REQ
    CRC Initial Seed                          0x40C9F7
    CRC Received                              0x7FC683, OK
  Timings
  Encryption
    State                                     decrypted
  ARQ and retransmissions
    ACK status                                ACKed
@ Connection event #281 at 15:57:24.805511 on ch 1...
  Event #281 on connection                    0xCF9A1635 15:57:16.3–16:02:49.3: 7E:B0:A1:A3:B6:9
  Channel index                               15
  RF Channel                                  17
  Contents                                    0/2/0 Empty/non-empty/missed packets
  Anchor time                                 15:57:24.805511
  @ → packet #267017, ACKed
    ← packet #267018, ACKed
@ → Data: LE-U Start/Complete PDU
  PDU Data Header payload
  Data                                        07 00 04 00 10 01 00 FF FF 00 28
  MIC                                         0xD4A23C3D
@ Aggregated L2CAP PDU, in 1 LL packet
  @ → ATT_READ_BY_GROUP_TYPE_REQ (0x10)
    1 component packet                        #267017
    Information Payload: Received length      7
    Expected length                           7 (0x0007)
    CID                                       Attribute protocol (0x0004)
    Payload                                   10 01 00 FF FF 00 28
    @ → ATT_READ_BY_GROUP_TYPE_REQ PDU, 1 pac...
      Code                                    ATT_READ_BY_GROUP_TYPE_REQ (0x10)
      ATT READ BY GROUP TYPE REQ payload
        Starting Handle                       0x0001
        Ending Handle                         MAX Attribute handle (0xFFFF)
        Attribute Group Type                  GATT Declaration: Primary Service (0x2800)
```

```
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
0x0000 0E 0F 07 00 04 00 10 01 00 FF FF 00 28 3D 3C A2  ............(=<.
0x0010 D4                                               .
```

In the Details pane, baseband packets display some details about the layer(s) above (e.g. the Connection Event and Control Procedure trees displayed for the LE Link Layer packet selected in the screenshot above), and higher-layer packets display some information about the layer(s) below.

Any highlighted fields in a payload shown in the Details tab correspond to bits or bytes within the Raw data. In the screenshot above, the "Attribute Group Type" field has been selected, and so the corresponding bytes 0x00, 0x28 in the Raw data have been highlighted.

Double-clicking on any field containing a time, or referencing another packet, will jump to that location in the Summary. These fields will display a tooltip showing what time you can jump to. Similar jump-to-time fields are present in the Connections and Devices tabs.



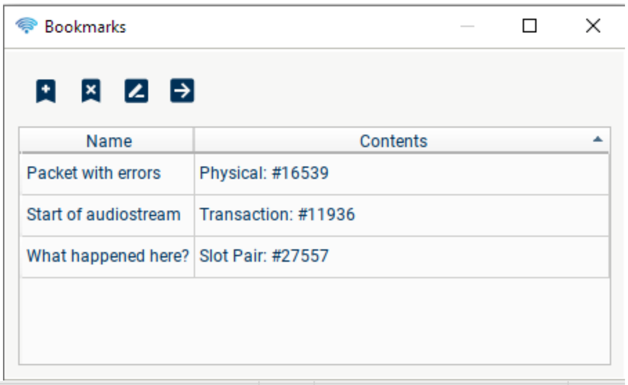
Name	Value
3 ADV_NONCONN_IND packets	
Sender	34:0D:40:CB:3B:DC, Non-resolvable private
First packet time	20:42:30.751369
ADV_NONCONN_IND, packet #688, ch 0	34:0D:40:CB:3B:DC (R), 31 bytes of AdvData (Microsoft)
Advertising: ADV_NONCONN_IND PDU	Goto 20:42:30.751369
PDU Adv Header payload	

The other fields with coloured backgrounds are section headers, marking different types of information about the packet. The colours used are consistent throughout the application; lighter colours denote baseband packets/information, and darker colours show Connection/Advertising Events, Slot Pairs, and higher protocol PDUs.

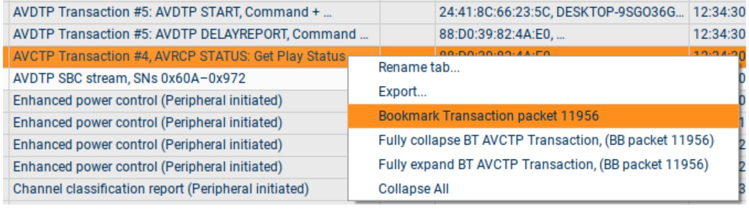
13.2 Bookmarks

Any packet or larger event can be bookmarked for later study or to send to other users; when you have added bookmarks, make sure to Save to add them to the file!

Bookmarks can either be added using the context menu in the Summary tab, or by selecting a packet and pressing 🔖 in the Bookmarks tab. Bookmarks can also be deleted and renamed in this tab; double-clicking on a bookmark, or pressing the ➡ button will jump to that packet in the Summary, if it is visible under the current filtering.



Name	Contents
Packet with errors	Physical: #16539
Start of audiostream	Transaction: #11936
What happened here?	Slot Pair: #27557


Rename tab...
Export...
Bookmark Transaction packet 11956
Fully collapse BT AVCTP Transaction, (BB packet 11956)
Fully expand BT AVCTP Transaction, (BB packet 11956)
Collapse All

AVDTP Transaction #5: AVDTP START, Command + ...	24:41:8C:66:23:5C, DESKTOP-9SGO36G...	12:34:30
AVDTP Transaction #5: AVDTP DELAYREPORT, Command ...	88:D0:39:82:4A:E0, ...	12:34:30
AVCTP Transaction #4, AVRCP STATUS: Get Play Status		
AVDTP SBC stream, SNs 0x60A–0x972		
Enhanced power control (Peripheral initiated)		
Enhanced power control (Peripheral initiated)		
Enhanced power control (Peripheral initiated)		
Enhanced power control (Peripheral initiated)		
Channel classification report (Peripheral initiated)		

13.3 Devices/Connections Details

This tab collects together extra information about the Connections and Devices selected in the filter currently being edited in the Filter Devices tab.

The connections trees show a summary of the Control Procedures/LMP packets sent, and detail of any L2CAP Channels in use. Details of any GATT Characteristics/Services and ATT Transactions seen are also found in the L2CAP Channels subtree.

For an LE Connection, in addition to the above information we collect a list of the Connection Parameters valid at each time during the Connection, including both Parameters transmitted and Parameters we have inferred from an ongoing Connection.

The device trees, in addition to device addresses, collect any other transmitted information including device names, vendors, IRKs, Class of Device, and various types of supported feature.

Each Device tree also includes summaries of the Advertising/Data/Paging/Connected packets sent and received, with links to jump to the first of each of these packet types.

13.4 Audio Streams

This tab displays details of the transport configuration (e.g CIS parameters) of any audiostreams we have seen (in contrast to the audio parameters, which are shown in the Audio Export tab.

13.5 Security Key Manager

See 4 Decrypting.

13.6 LE Access Addresses

This displays the LE Access Addresses that we have seen during the capture; each row is a link to jump to the first packet seen on that AA. Only confirmed AAs are displayed here; packets with CRC failures will not cause an AA to be added to the list.

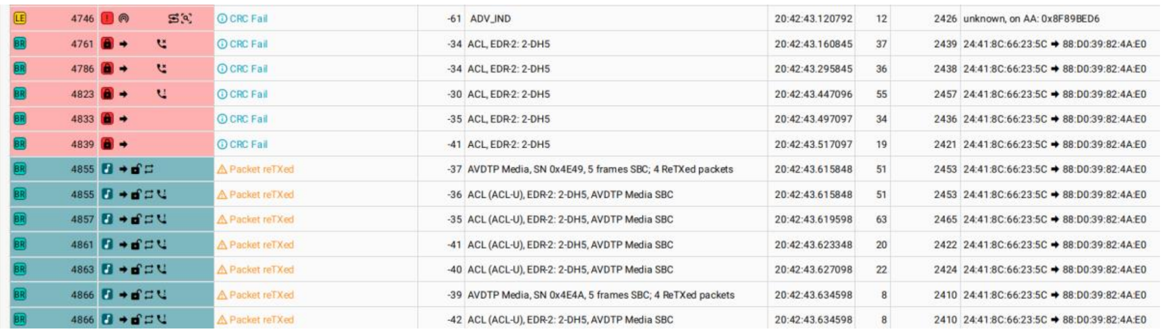
13.7 Topology

This shows the current devices, connections and audiostreams in the chosen device filter. You can enable/disable labels on the devices and the streams, and you can choose to show or hide “Inert” devices, i.e. those only advertising/inquiring and not participating in any displayed connections. The colours of the devices and links indicate whether they are communicating using LE, Classic, or both. The icons decorating the links indicate whether any audio is currently being sent, and in which direction.

13.8 Packet errors/issues

This collects together errors/warnings/informative messages seen during the capture, and allow filtering and sorting them to easily display all the instances of a particular type of issue. The issues displayed include both Bluetooth errors (invalid values, packets with invalid lengths etc), communication impairments (ReTXed packets, rejected packets), receive impairments (CRC fails, MIC fails, missing packets), and analyser difficulties (e.g. protocols not parsed, either due to use of proprietary protocols or features not yet supported).

Double-clicking on a row will navigate to that packet/event in the active Summary tab.

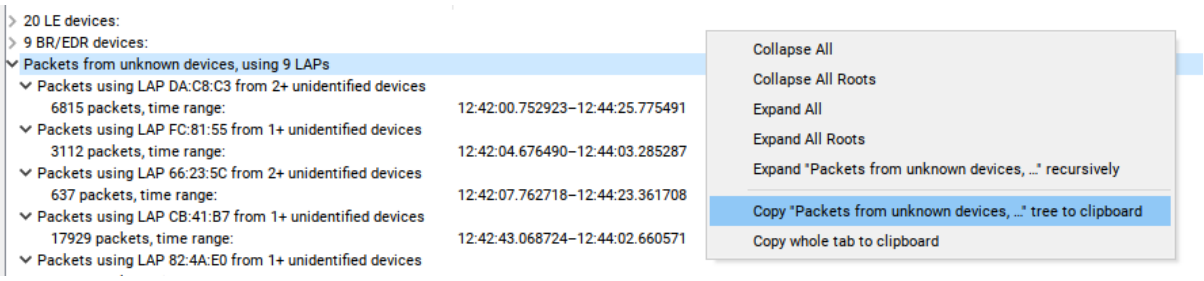


LE	4746	!	ⓘ CRC Fail	-61	ADV_IND	20:42:43.120792	12	2426	unknown, on AA: 0x8F89BED6
BR	4761	→ ☎	ⓘ CRC Fail	-34	ACL, EDR-2: 2-DH5	20:42:43.160845	37	2439	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4786	→ ☎	ⓘ CRC Fail	-34	ACL, EDR-2: 2-DH5	20:42:43.295845	36	2438	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4823	→ ☎	ⓘ CRC Fail	-30	ACL, EDR-2: 2-DH5	20:42:43.447096	55	2457	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4833	→	ⓘ CRC Fail	-35	ACL, EDR-2: 2-DH5	20:42:43.497097	34	2436	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4839	→	ⓘ CRC Fail	-41	ACL, EDR-2: 2-DH5	20:42:43.517097	19	2421	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4855	♪ →	⚠ Packet reTXed	-37	AVDTP Media, SN 0x4E49, 5 frames SBC; 4 ReTXed packets	20:42:43.615848	51	2453	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4855	♪ → ☎	⚠ Packet reTXed	-36	ACL (ACL-U), EDR-2: 2-DH5, AVDTP Media SBC	20:42:43.615848	51	2453	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4857	♪ → ☎	⚠ Packet reTXed	-35	ACL (ACL-U), EDR-2: 2-DH5, AVDTP Media SBC	20:42:43.619598	63	2465	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4861	♪ → ☎	⚠ Packet reTXed	-41	ACL (ACL-U), EDR-2: 2-DH5, AVDTP Media SBC	20:42:43.623348	20	2422	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4863	♪ → ☎	⚠ Packet reTXed	-40	ACL (ACL-U), EDR-2: 2-DH5, AVDTP Media SBC	20:42:43.627098	22	2424	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4866	♪ → ☎	⚠ Packet reTXed	-39	AVDTP Media, SN 0x4E4A, 5 frames SBC; 4 ReTXed packets	20:42:43.634598	8	2410	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0
BR	4866	♪ → ☎	⚠ Packet reTXed	-42	ACL (ACL-U), EDR-2: 2-DH5, AVDTP Media SBC	20:42:43.634598	8	2410	24:41:8C:66:23:5C → 88:D0:39:82:4A:E0

The “Filter Errors” popup can be used to select which errors to display. To select/deselect a whole error level, use the leftmost column; for error categories, the middle column, and for individual errors, the rightmost column.

13.9 Exporting Data

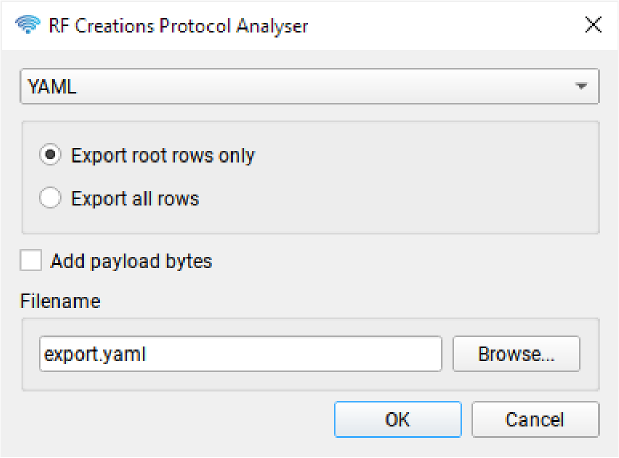
In the Details, Connections, and Devices Detail tabs, the contents of the tabs can be copied to the clipboard for pasting into a text editor or other application. The context menu allows copying either the whole tab, or individual subtrees.



```
> 20 LE devices:
> 9 BR/EDR devices:
v Packets from unknown devices, using 9 LAPs
  v Packets using LAP DA:C8:C3 from 2+ unidentified devices
      6815 packets, time range:            12:42:00.752923–12:44:25.775491
  v Packets using LAP FC:81:55 from 1+ unidentified devices
      3112 packets, time range:            12:42:04.676490–12:44:03.285287
  v Packets using LAP 66:23:5C from 2+ unidentified devices
      637 packets, time range:             12:42:07.762718–12:44:23.361708
  v Packets using LAP CB:41:B7 from 1+ unidentified devices
      17929 packets, time range:           12:42:43.068724–12:44:02.660571
  v Packets using LAP 82:4A:E0 from 1+ unidentified devices
```

Collapse All
Collapse All Roots
Expand All
Expand All Roots
Expand "Packets from unknown devices, ..." recursively
Copy "Packets from unknown devices, ..." tree to clipboard
Copy whole tab to clipboard

Additionally, the complete contents of a Summary tab can be exported as either a CSV or YAML file using the "Export..." option in the context menu. When using an aggregation level other than "Baseband", you have the option to export only the top-level, root rows. You can also choose to add the complete payload bytes (dewhitened, decrypted etc) to the export.



RF Creations Protocol Analyser

YAML

● Export root rows only

○ Export all rows

□ Add payload bytes

Filename: export.yaml

14 Queries

Bluespy supports various query strings, which can be applied to any event and are used to define the columns, for searching and in the API. The querys can all be represented as a string, but may also return an integer or a boolean. The following queries (this list can also be found in Help->Query List) are supported:

CID	L2CAP Channel Identifier
LAP	Classic Lower Address Part
T_IFS_before	Measurement of T_IFS before this packet
T_IFS_after	Measurement of T_IFS after this packet
access_address	LE Access Address
access_code	Access Address or LAP
advA	Advertising Address sent in a particular advertising packet
acked	Packet was acknowledged (bool)
aux_pkt	The auxiliary advertising packet
aux_pkt_window	The time window that should contain an auxiliary advertising packet
broadcast_code	Broadcast code for encrypted BIG
broadcast_id	Broadcast ID for BIG
broadcast_name	Broadcast Name for BIG
central.addr	Bluetooth address of the central device
central.irk	Identity Resolving Key of the central device
central.localnames	Discovered names of the central device
central.vendor_names	Vendor(s) of the central device
central_packet_count	Packet number from central in this LE connection event
channel	RF channel of packet
cie	CIE: Close Isochronous Event
connection_handle	HCI Connection Handle
cssn	CSSN: Control Subevent Sequence Number
cstf	CSTF: Control Subevent Transmission Flag
cl.ARQN	Classic ARQN
cl.SEQN	Classic SEQN
connectable	Advertising Event/Set is connectable (bool)
crc	CRC of packet
crc_valid	CRC passed (bool)
CSIPS_info	Information regarding Coordinated Sets
cte	LE packet has constant tone extension (bool)
decrypted	Packet has been successfully decrypted (bool)
delta_ref	Time delta to last selected packet
device_summary	Information on sender and receiver
directed	LE Advertising is a directed type
duration	Packet time duration
dukosi.crc	CRC
dukosi.source_id	Source ID

dukosi.message_type	Message Type
emojis	A list of symbols concisely expressing essential packet stats
encrypted	The packet was encrypted
errors	Problems detected in this packet
event_counter	LE connection event number or Classic clock
frequency	RF frequency
freq_error_ppm	RF frequency error (ppm)
flow_packet	FLOW (packet-level)
flow_payload	FLOW (payload-level)
handles	ATT handles
initA	Connection Initiator Address
initiator.addr	CS Initiator Address
is_ID	Packet was a Classic ID packet (bool)
is_adv	Packet is LE advertising (bool)
is_data	Packet is LE data (bool)
is_empty	Packet has no payload (bool)
is_isoc	Packet is isochronous (bool)
is_test	LE Test packet (bool)
llid	LLID
local_name	Local name
lt_addr	LT Address
md	MD: More Data
media_timestamp	AVDTP media timestamp
modulation	PHY layer modulation scheme
nacked	Packet was negatively acknowledged
nesn	NESN: Next Expected Sequence Number
npi	NPI: Null PDU Indicator
payload_counters	Payload Counters
payload_hex	Packet payload in hexadecimal format
payload_length	Number of payload bytes (integer)
payload_raw	Payload as raw bytes
payload_summary	Beginning of payload
peripheral.addr	Bluetooth address of the peripheral device
peripheral.irk	Identity Resolving Key of the peripheral device
peripheral.localnames	Discovered names of the peripheral device
peripheral.vendor_names	Vendor(s) of the peripheral device
peripheral_packet_count	Packet number from peripheral in this LE connection event
pkt_no	Baseband packet index in the capture file
pkt_type	Baseband packet type
read_by_type_uuid	GATT read by type UUID
receiver.addr	Bluetooth address of the receiver device
receiver.irk	Identity Resolving Key of the receiver device
receiver.localnames	Discovered names of the receiver device
receiver.vendor_names	Vendor(s) of the receiver device

rejected	Packet was rejected by receiver (bool)
resolvable_set_id	Resolvable set ID of Coordinated Set member
rssi	Received signal strength in dBm at Moreph
scanA	Scanner Address
scannable	LE Advertising is scannable (bool)
sender.addr	Bluetooth address of the sender device
sender.irk	Identity Resolving Key of the sender device
sender.localnames	Discovered names of the sender device
sender.vendor_names	Vendor(s) of the sender device
service_uuid	GATT service UUID
sn	SN: Sequence Number
ssrc	AVDTP SSRC
status	Packet status
summary	Packet summary
superior_pkts	The superior packets of this advertising packet
sync_pkt	The SYNC packet pointed to by this advertising packet
sync_pkt_window	The time window that should contain a SYNC packet
sync_pointing_pkts	The advertising packets pointing to this SYNC packet
targetA	The address this advertising packet is targetting
throughput	Byte throughput rate
time	Packet timestamp
type	Kind of packet

15 C API

The C API can be viewed in bluespy.h in the root of the installation. You will need to link against the libblueSPY shared library.

15.1 BLUESPY.H FILE REFERENCE

blueSPY C API

15.1.1 Classes

- struct bluespy_capture_options

15.1.2 Typedefs

- typedef enum bluespy_error **bluespy_error**
- typedef enum bluespy_log_level **bluespy_log_level**
- typedef struct bluespy_capture_options **bluespy_capture_options**
- typedef uint64_t **bluespy_event_id**
  Indentifier for an packet or higher layer event -1 means invalid Only use bluespy_event_ids returned by the API After bluespy_load_file or bluespy_capture all previous ids are invalid.

15.1.3 Enumerations

- enum **bluespy_error** : uint32_t {
  - **BLUESPY_NO_ERROR** = 0,
  - **BLUESPY_ERROR_NO_DEVICE**, **BLUESPY_ERROR_LICENCE**,
  - **BLUESPY_ERROR_NO_FILE**,
  - **BLUESPY_ERROR_CAPTURE_NOT_STARTED**,
  - **BLUESPY_ERROR_INVALID_PACKET** }

- enum **bluespy_log_level** : uint32_t {
  - **BLUESPY_LOG_PASS** = 0x00,
  - **BLUESPY_LOG_WARN** = 0x20,
  - **BLUESPY_LOG_INFO** = 0x40,
  - **BLUESPY_LOG_DEBUG** = 0x60,
  - **BLUESPY_LOG_ERROR** = 0x80 }

15.1.4 Functions

- BLUESPY_API const char * bluespy_error_string (bluespy_error error)
  Get message for error.

- BLUESPY_API void **bluespy_init** ()
  Initialise bluespy, run once at start of program.

- BLUESPY_API void **bluespy_deinit** ()
  Clean up before program exits.

- BLUESPY_API void bluespy_start_gui ()
  Start a GUI instance.

- BLUESPY_API bluespy_error bluespy_connect (uint32_t serial=-1)
  Connect to Moreph.

- BLUESPY_API bluespy_error bluespy_disconnect ()
  Disconnect from Moreph.

- BLUESPY_API bluespy_error bluespy_add_log_message (bluespy_log_level level, const char *message, uint64_t ts)
  Adds a log message into the file.

- BLUESPY_API bluespy_capture_options * bluespy_capture_options_alloc ()
  Create a bluespy_capture_options struct.

- BLUESPY_API void **bluespy_capture_options_delete** (bluespy_capture_options *opts)
  Delete a bluespy_capture_options struct.

- BLUESPY_API bluespy_error bluespy_capture (const char *filename, bluespy_capture_options *opts)
  Start a capture.

- BLUESPY_API bluespy_error bluespy_stop_capture ()
  Stop a capture.

- BLUESPY_API bluespy_error bluespy_load_file (const char *filename)
  Load a capture.

- BLUESPY_API bluespy_error bluespy_close_file ()
  Close current file.

- BLUESPY_API uint32_t bluespy_packet_count (void)
  Number of baseband packets loaded.

- BLUESPY_API bluespy_event_id bluespy_get_baseband (uint32_t index)
  Get a baseband packet.

- BLUESPY_API bluespy_event_id bluespy_get_parent (bluespy_event_id event)
  Get higher layer packets.

- BLUESPY_API const bluespy_event_id * bluespy_get_children (bluespy_event_id event, uint32_t *count)
  Get all lower layer packets.

- BLUESPY_API const char * bluespy_query (bluespy_event_id event, const char *query)
  Query a packet.

- BLUESPY_API int64_t bluespy_query_int (bluespy_event_id event, const char *query)
  Query a packet.

- BLUESPY_API bool bluespy_query_bool (bluespy_event_id event, const char *query)
  Query a packet.

- BLUESPY_API int bluespy_query_auto (bluespy_event_id event, const char *query, const char **s, int64_t *i, bool *b)
  Query a packet.

- BLUESPY_API bluespy_error bluespy_add_link_key (const unsigned char *key, uint64_t addr0, uint64_t addr1)
  Add a link key for decryption.

- BLUESPY_API void bluespy_add_IRK (const unsigned char *key, uint64_t *addr, uint64_t n_addresses)
  Add an IRK.

15.1.5 Detailed Description

blueSPY C API

15.1.6 Class Documentation

15.1.6.1 struct bluespy_capture_options

15.1.6.1.1 Class Members:

bool	enable_15_4	
bool	enable_CL	
bool	enable_LE	
bool	enable_MHDT_CL	
bool	enable_MHDT_LE	
bool	enable_QHS	
bool	enable_wifi	
uint16_t	spectrum_period	Valid spectrum periods in microseconds are: 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000 0 to disable spectrum_

15.1.7 Function Documentation

*15.1.7.1 BLUESPY_API void bluespy_add_IRK (const unsigned char * key, uint64_t * addr, uint64_t n_addresses)*

Add an IRK.

15.1.7.1.1 Parameters

in	*key*	- 16 bytes of binary
in	*addr*	- Array of relevant Bluetooth addresses, can be empty. Set bit 48 = 1 for Random, = 0 for Public
in	*n_addresses*	- Length of array

The key should be added before loading/capturing, or you should load again afterwards.

*15.1.7.2 BLUESPY_API bluespy_error bluespy_add_link_key (const unsigned char * key, uint64_t addr0, uint64_t addr1)*

Add a link key for decryption.

15.1.7.2.1 Parameters

in	*key*	- 16 bytes of binary
in	*addr0*	- Bluetooth address of first device, set to 0 if unknown
in	*addr1*	- Bluetooth address of second device, set to 0 if unknown

15.1.7.2.2 Returns

Error code

The link key should be added before loading/capturing, or you should load again afterwards.

*15.1.7.3 BLUESPY_API bluespy_error bluespy_add_log_message (bluespy_log_level level, const char * message, uint64_t ts)*

Adds a log message into the file.

15.1.7.3.1 Parameters

in	*level*	
in	*message*	
in	*ts*	

15.1.7.3.2 Returns

Error code

*15.1.7.4 BLUESPY_API bluespy_error bluespy_capture (const char * filename, bluespy_capture_options * opts)*

Start a capture.

15.1.7.4.1 Parameters

in	*filename*	- UTF8 filename
in	*opts*	- Capture options

15.1.7.4.2 Returns

Error code

*15.1.7.5 BLUESPY_API bluespy_capture_options * bluespy_capture_options_alloc ()*

Create a bluespy_capture_options struct.

15.1.7.5.1 Returns

New struct

15.1.7.6 BLUESPY_API bluespy_error bluespy_close_file ()

Close current file.

15.1.7.6.1 Returns

Error code

15.1.7.7 BLUESPY_API bluespy_error bluespy_connect (uint32_t serial = −1)

Connect to Moreph.

15.1.7.7.1 Parameters

in	*serial*	- Serial number of the Moreph

15.1.7.7.2 Returns

Error code

Connect by serial number, or first found on USB if serial == -1

15.1.7.8 BLUESPY_API bluespy_error bluespy_disconnect ()

Disconnect from Moreph.

15.1.7.8.1 Returns

Error code

15.1.7.9 BLUESPY_API const char * bluespy_error_string (bluespy_error error)

Get message for error.

15.1.7.9.1 Parameters

in	*error*	

15.1.7.9.2 Returns

Internal pointer to null terminated string (do not free)

15.1.7.10 BLUESPY_API bluespy_event_id bluespy_get_baseband (uint32_t index)

Get a baseband packet.

15.1.7.10.1 Parameters

in	*index*	- 0 <= index < bluespy_packet_count()

15.1.7.10.2 Returns

Event ID

15.1.7.11 BLUESPY_API const bluespy_event_id * bluespy_get_children (bluespy_event_id event, uint32_t * count)

Get all lower layer packets.

15.1.7.11.1 Parameters

in	*event*	
out	*count*	- size of returned array

15.1.7.11.2 Returns

Event ID array

Returned child array is valid until next call, so take a copy

15.1.7.12 BLUESPY_API bluespy_event_id bluespy_get_parent (bluespy_event_id event)

Get higher layer packets.

15.1.7.12.1 Parameters

in	*event*	

15.1.7.12.2 Returns

Event ID

15.1.7.13 BLUESPY_API bluespy_error bluespy_load_file (const char * filename)

Load a capture.

15.1.7.13.1 Parameters

in	*filename*	- UTF8 filename

15.1.7.13.2 Returns

Error code

15.1.7.14 BLUESPY_API uint32_t bluespy_packet_count (void)

Number of baseband packets loaded.

15.1.7.14.1 Returns

N

*15.1.7.15 BLUESPY_API const char * bluespy_query (bluespy_event_id event, const char * query)*

Query a packet.

15.1.7.15.1 Parameters

in	*event*	
in	*query*	- query string to apply

15.1.7.15.2 Returns

String result of a query

Returned value is valid until next call, so take a copy

*15.1.7.16 BLUESPY_API int bluespy_query_auto (bluespy_event_id event, const char * query, const char ** s, int64_t * i, bool * b)*

Query a packet.

15.1.7.16.1 Parameters

in	*event*	
in	*query*	- query string to apply
out	*s*	- return if string
out	*i*	- return if int
out	*b*	- return if bool

15.1.7.16.2 Returns

0 = None, 1 = str, 2 = int, 3 = bool

Returned string value is valid until next call, so take a copy

*15.1.7.17 BLUESPY_API bool bluespy_query_bool (bluespy_event_id event, const char * query)*

Query a packet.

15.1.7.17.1 Parameters

in	*event*	
in	*query*	- query string to apply

15.1.7.17.2 Returns

Bool result of a query

*15.1.7.18 BLUESPY_API int64_t bluespy_query_int (bluespy_event_id event, const char * query)*

Query a packet.

15.1.7.18.1 Parameters

in	*event*	
in	*query*	- query string to apply

15.1.7.18.2 Returns

Integer result of a query

15.1.7.19 BLUESPY_API void bluespy_start_gui ()

Start a GUI instance.

The GUI exists in a background thread, this function returns immediately

15.1.7.20 BLUESPY_API bluespy_error bluespy_stop_capture ()

Stop a capture.

15.1.7.20.1 Returns

Error code

15.1.7.21

---

16 Python API

The python API provides functions for connecting to a moreph and loading existing captures. It also provides a 'packets' object, which behaves like a list and can be used to access the packets in the current file. `len(packets)` shows the number of available packets, `packets[0]` accesses the first packet. The queries documented above can be accessed with attribute syntax, e.g. `packets[0].summary`

Examples:

To connect to a moreph with serial 00010100 and capture CL and LE:

```
import bluespy
from time import sleep

bluespy.connect(0x00010100)
bluespy.capture("example.pcapng", CL=True, LE=True)
sleep(20)
bluespy.stop_capture()
bluespy.disconnect()
print("Captured {} packets".format(len(bluespy.packets)))
bluespy.close_file()
```

To load an existing capture and print the summary strings of all packets:

```
import bluespy

bluespy.load_file("example.pcapng")
for p in bluespy.packets:
    print(p.summary)
bluespy.close_file()
```

16.1 BLUESPY.PY File Reference

16.1.1 Classes

- class bluespy.error*Return type showing why an operation failed.*
- class bluespy.log_level*Return log level.*
- class bluespy.BluespyError*Exception showing why an operation failed.*
- class bluespy.event_id*An object referencing a loaded packet.*
- class bluespy.Packets*List-like object representing the currently loaded baseband packets.*

16.1.2 bluespy.BluespyError Class Reference

Exception showing why an operation failed.

16.1.2.1 Public Member Functions

- def **get_error** (self)
  Return the underlying error object.

16.1.2.2 Detailed Description

Exception showing why an operation failed.

16.1.3 bluespy.error Class Reference

Return type showing why an operation failed.

16.1.3.1 Public Member Functions

- def **__str__** (self)
- def **__repr__** (self)
- def **__bool__** (self)

16.1.3.2 Public Attributes

- **value**

16.1.3.3 Detailed Description

Return type showing why an operation failed.

Evaluates to True if the operation succeeded, else False. str() and repr() give an error string, .value gives an error code.

16.1.4 bluespy.Packets Class Reference

List-like object representing the currently loaded baseband packets.

16.1.4.1 Public Member Functions

- def **__len__** (self)
  Current number of baseband packets captured.

- def __getitem__ (self, i)
  Get an event_id for a packet.

16.1.4.2 Detailed Description

List-like object representing the currently loaded baseband packets.

16.1.4.3 Member Function Documentation

16.1.4.3.1 def bluespy.Packets.__getitem__ (*self*, *i*)

Get an event_id for a packet.

16.1.4.3.1.1 Parameters

i	Index of packet, 0 <= i < **len** ()

16.1.4.3.1.2 Returns

: An event_id

16.1.5 bluespy.event_id Class Reference

An object referencing a loaded packet.

16.1.5.1 Public Member Functions

- def **__bool__** (self)
  Returns true if this is a valid packet.

- def parent (self)
  Get the a higher layer packet that contains this one.

- def children (self)
  Get all constituent packets of this packet.

- def query (self, name)
  Get a query from this packet, and return it in its preferred form.

- def **__getattr__** (self, name)
  Access queries as attributes.

- def query_str (self, name)
  Get a query from this packets, and return it as a string.

- def query_int (self, name)
  Get a query from this packets, and return it as an integer if possible.

- def query_bool (self, name)
  Get a query from this packets, and return it as a bool if possible.

16.1.5.2 Detailed Description

An object referencing a loaded packet.

Do not make your own, only get these from the 'packets' object. After running close_file(), do not call any methods on any existing event_id objects.

Any query (see the documentation or Help->Query List in the GUI) can be accessed as an attribute on this object

16.1.5.3 Member Function Documentation

16.1.5.3.1 def bluespy.event_id.children (*self*)

Get all constituent packets of this packet.

e.g. if this is an L2CAP packet

16.1.5.3.1.1 Returns

: List of event_ids

16.1.5.3.2 def bluespy.event_id.parent (*self*)

Get the a higher layer packet that contains this one.

e.g. if this is a baseband data packet get the L2CAP packet it is part of.

16.1.5.3.2.1 Returns

: event_id

16.1.5.3.3 def bluespy.event_id.query (*self*, *name*)

Get a query from this packet, and return it in its preferred form.

16.1.5.3.3.1 Parameters

name	A query string, e.g. "summary"

16.1.5.3.3.2 Returns

: A string, int or bool depending on the query

16.1.5.3.4 def bluespy.event_id.query_bool (*self*, *name*)

Get a query from this packets, and return it as a bool if possible.

16.1.5.3.4.1 Parameters

name	A query string, e.g. "acked"

16.1.5.3.4.2 Returns

: bool

16.1.5.3.5 def bluespy.event_id.query_int (*self*, *name*)

Get a query from this packets, and return it as an integer if possible.

16.1.5.3.5.1 Parameters

name	A query string, e.g. "summary"

16.1.5.3.5.2 Returns

: int

16.1.5.3.6 def bluespy.event_id.query_str (*self*, *name*)

Get a query from this packets, and return it as a string.

16.1.5.3.6.1 Parameters

name	A query string, e.g. "summary"

16.1.5.3.6.2 Returns

: string

16.1.6 Functions

- **bluespy._handle_error** (err)
- bluespy.connect (serial=-1)
  Connect to Moreph hardware via USB or Ethernet.

- bluespy.disconnect ()
  Disconnect from current Moreph.

- bluespy.add_log_message (level, message, ts=0)
  Adds a log message into the running capture.

- bluespy.capture (filename, CL=False, LE=False, QHS=False, _15_4=False, wifi=False, MHDT_CL=False, MHDT_LE=False, Dukosi=False, Varjo=False, CS=False, spectrum=0)
  Start a new capture in filename.

- bluespy.stop_capture ()
  Stop the current capture.

- bluespy.load_file (filename)
  Load an existing capture.

- bluespy.close_file ()
  Load an existing capture.

- bluespy.add_link_key (key, addr0=0, addr1=0)
  Add a link key for decryption.

- **bluespy.start_gui** ()
  Spawn an instance of the user interface.

16.1.7 Variables

- **bluespy.packets** = Packets()
- **bluespy.argtypes**
- **bluespy.restype**

16.1.7.1 *bluespy.add_link_key (key, addr0 = 0, addr1 = 0)*

Add a link key for decryption.

16.1.7.1.1 Parameters

key	Link key as a 16-byte bytes object
addr0	(Optional) MAC address of central as a 64-bit integer
addr1	(Optional) MAC address of peripheral as a 64-bit integer

16.1.7.2 *bluespy.add_log_message (level, message, ts = 0)*

Adds a log message into the running capture.

16.1.7.2.1 Parameters

level	The Log Level.
message	The log message content.
ts	The time of the log message. ts=0 means the time will be set to the present.

16.1.7.2.2 Exceptions

BluespyError	Exception in the bluespy library

16.1.7.3 *bluespy.capture (filename, CL = **False**, LE = **False**, QHS = **False**, _15_4 = **False**, wifi = **False**, MHDT_CL = **False**, MHDT_LE = **False**, Dukosi = **False**, Varjo = **False**, CS = **False**, spectrum = 0)*

Start a new capture in filename.

16.1.7.3.1 Parameters

filename	Path to store capture in
CL	Enable Bluetooth classic capture
LE	Enable Bluetooth LE capture
QHS	Enable Qualcomm High Speed capture
_15_4	Enable 802.15.4 capture
wifi	Enable wifi capture
MHDT_CL	Enable MediaTek mHDT Classic capture
MHDT_LE	Enable MediaTek mHDT LE capture
Dukosi	Enable Dukosi capture
Varjo	Enable Varjo capture
CS	Enable Channel-Sounding capture
spectrum	Spectrum capture interval in microseconds. 0 means disabled. Allowed values are: 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000.

16.1.7.3.2 Exceptions

BluespyError	Exception in the bluespy library

16.1.7.4 *bluespy.close_file ()*

Load an existing capture.

16.1.7.4.1 Exceptions

BluespyError	Exception in the bluespy library

16.1.7.5 bluespy.connect (serial = −1)

Connect to Moreph hardware via USB or Ethernet.

16.1.7.5.1 Parameters

serial	A serial number as an integer, or -1 to connect to the first USB device. The serial number shown in the software and on the MiniMoreph is hexadecimal, so should be entered as 0xNNNNNN. There is a serial number on the bottom of some Moreph30s is of the form AYYYY-XXXXX, the XXXXX is the required serial number in decimal.

16.1.7.5.2 Exceptions

BluespyError	Exception in the bluespy library

You should run disconnect() later if this is successful.

16.1.7.6 bluespy.disconnect ()

Disconnect from current Moreph.

16.1.7.6.1 Exceptions

BluespyError	Exception in the bluespy library

16.1.7.7 bluespy.load_file (filename)

Load an existing capture.

16.1.7.7.1 Exceptions

BluespyError	Exception in the bluespy library

16.1.7.8 bluespy.stop_capture ()

Stop the current capture.

16.1.7.8.1 Exceptions

BluespyError	Exception in the bluespy library