# LE Audio

## LE audio 是什么？

LE Audio 是蓝牙技术联盟（SIG）在 2020 年国际消费电子展上推出的新一代蓝牙低功耗音频技术，能以蓝牙低功耗状态下传递音频，有利于提升蓝牙耳机续航力和性能，同时也导入新一代蓝牙音频编码 LC3（Low Complexity Communication Codec），以及多种音频分享、广播音频模式等，并且有利于助听器产品，能够达到低延迟、高音质等功能。



- LC3 打造更好更高音质和更低功耗

LE Audio 将采用全新的低复杂性通信编解码器，以提供实现更高的音质和更低的功耗，LE Audio可以在低比特率下的状态下提供较好质量的音频，甚至在比特率降低50%的情况下仍然如此。LE Audio允许开发者自行在音频质量和功耗之间进行取舍，采样率分为8、16、24、32、44.1、48kHz，比特率在16-320kbps之间 ，帧速率可选7.5、10ms。LC3将为开发者提供巨大的灵活性，使其能够更好地在音质和功耗等重要产品参数之间进行权衡，极大地提高蓝牙音频设备的自主性。开发者将能够利用这一优势，打造出电池寿命更持久的产品，或通过使用更小尺寸的电池来优化产品设计尺寸等。

-多重串流音频

LE Audio的多重串流音频（Multi-Stream Audio）可助力打造性能更卓越的入耳式耳机。多重串流音频功能将实现在智能手机等单一音频源设备（source device）、单个或多个音频接收设备（sink device）间，同步进行多重且独立的音频串流传输。无需先将音频信号发送至一个耳机，再转发至另一个耳机的处理方式。多重串流音频功能提升蓝牙耳机的连接稳定性、降低延迟，使得不同的音频设备切换更加顺畅。

- LE Audio为助听器应用提供更强大的支持

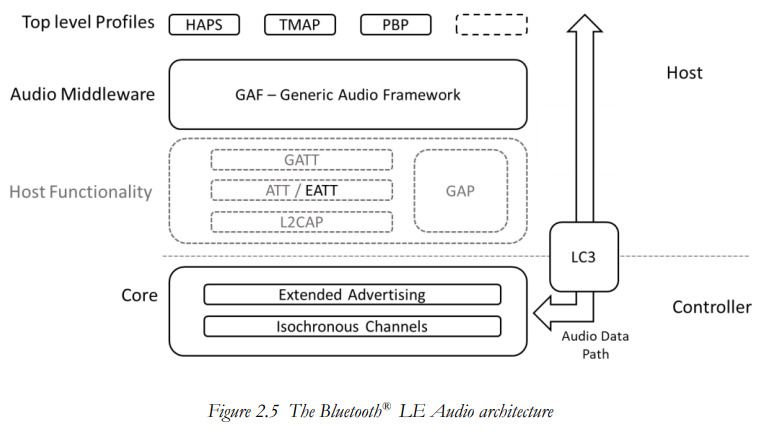
LE Audio应用在蓝牙助听器上将为听力受损人士带来福音。低功耗、高音质和多重串流功能为基础的特性提供更强大的助听器支持。目前，蓝牙音频已在无线通话、收听和观赏等方面，为全球广大的用户提供了极大的便利，同样的，蓝牙音频的所有特点将带给数量渐增的听力受损人士。

- LE Audio支持音频分享（Audio Sharing）

LE Audio还将添加广播音频功能，使单一音频源设备能够向不限数量的音频接收器设备广播一个或多个音频串流。基于LE Audio的蓝牙音频分享既可基于个人，也可基于位置。借助个人的音频分享，用户将能够与周围的人分享自己的蓝牙音频，例如，通过蓝牙功能和亲朋好友分享手机上的音乐。借助基于位置的蓝牙音频分享，在可以在车站、机场、商场、电影院和酒吧等公共场所分享蓝牙音频，从而提升这些公共场所来访者的体验。

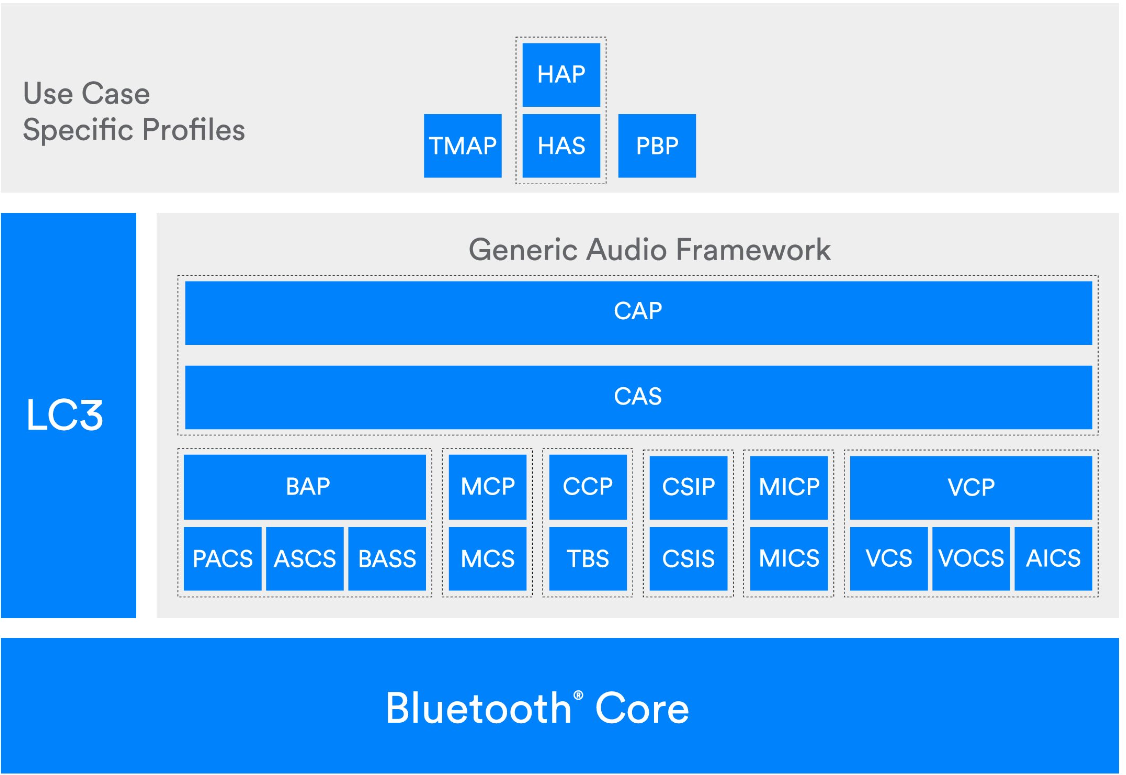
## LE Audio architecture

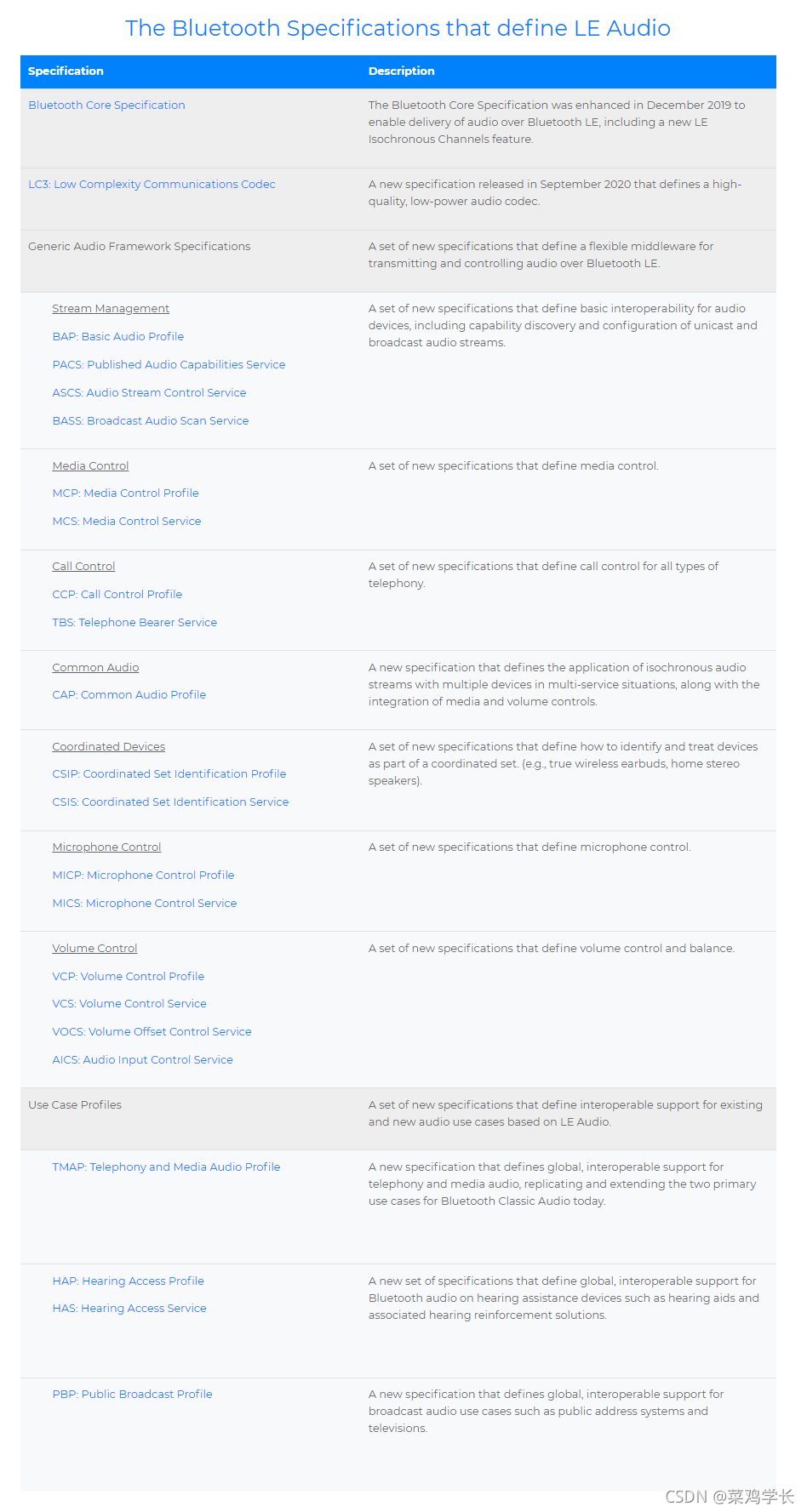
The Bluetooth LE Audio architecture has been built up in layers, as has every other Bluetooth specification before it. This is illustrated in Figure 2.5, which shows the main new specification blocks relating to Bluetooth LE Audio, (with key existing ones greyed out or dotted).



At the bottom we have the Core, which contains the radio and Link Layer, (collectively known as the Controller). It is responsible for sending Bluetooth packets over the air. On top of that is the Host, which has the task of telling the Core what to do for any specific application.

In the Host, there is a new structure called the Generic Audio Framework or GAF. This is an audio middleware, which contains all of the functions which are considered to be generic, i.e., features which are likely to be used by more than one audio application. The Core and the GAF are the heart of Bluetooth LE Audio. They provide great flexibility. Finally, at the top of the stack, we have what are termed “top level” profiles, which add application specific information to the GAF specifications.





## Top level Bluetooth LE Audio profiles

Although they’re still called profiles, in most cases they’re a lot simpler than the underlying profiles we’ve discussed, or the Bluetooth Classic Audio profiles. Instead of defining procedures, they generally confine themselves to configuration, specifying new roles which mandate a combination of optional features, and adding QoS requirements beyond those of BAP. As they rely on features which are already defined in the GAF specifications, they do not have complementary service specifications. HAP – the Hearing Access Profile is the exception, as the corresponding Hearing Access Service introduces a new Presets characteristic for Hearing Aids. TMAP – the Telephony Media and Audio Profile has a nominal TMAS service, which is included in the TMAP specification. The Public Broadcast Profile has no service. That’s an anomaly that is inherent with a broadcast application which does not expect a connection between Initiator and Acceptor. The lack of connection implies no possibility of a Client-Server relationship, hence no opportunity for a Service specification. A number of top level profiles are currently being developed in the Bluetooth working groups, but the first three scheduled for adoption are:

• HAP and HAS – the Hearing Access Profile and Service, which define requirements for products which are intended for use in the hearing aid ecosystem.

• TMAP – the Telephony and Media Audio Profile. This aims to cover the main features of the classic HFP and A2DP profiles, adding in new capabilities arising from the Bluetooth LE Audio topologies.

• PBP – the Public Broadcast Profile, which is a foundation of the Bluetooth LE Audio Sharing use case. It identifies that a broadcast Audio Stream can be received by any Bluetooth LE Audio Acceptor

# HAPS the Hearing Access Profile and Service

听力访问profile和service旨在满足助听器生态系统中使用的设备的要求，包括助听器、为助听器提供音频流的产品以及用于控制助听器的附件

<https://blog.csdn.net/u014056414/article/details/126563328>

As the whole of the Bluetooth LE Audio development was started by the hearing aid industry, it seems only fair to start with HAP and HAS.

HAPS the Hearing Access Profile and Service

The Hearing Access Profile and Service have been designed to meet the requirements of devices which are used in the Hearing Aid ecosystem, which covers hearing aids, products which supply Audio Streams to hearing aids and accessories which are used to control them. Hearing Aids are different to earbuds and other Acceptors because they are always on, capturing and processing ambient sound to assist their wearers. That means that all of the use cases driving the Hearing Access profile envisage Bluetooth LE Audio as an additional audio stream to the ambient one. This makes them unusual, as Bluetooth connectivity is not the de facto reason for buying them. They also differ in that everyone who wears them has hearing loss, so they have a different balance in requirements between audio quality (and hence QoS settings) and battery life. Taking your hearing aid out to recharge it during the day is a far more significant action than it is for an earbud wearer, as you may not be able to hear during that time. Pushing up audio quality increases the power consumption, so the Hearing Access profile imposes no additional QoS requirements over the settings mandated by BAP, using the 16\_2 LC3 codec settings for voice (16kHz sampling rate, 7 kHz bandwidth) and the 24\_2 codec settings for music (24 kHz sampling rate, 11 kHz bandwidth). As many hearing aids do not occlude the ear, ambient sounds can be heard in addition to the Bluetooth stream. For this reason, hearing aids generally prefer to use the Low Latency QoS settings to avoid echo between the ambient and transmitted sound.

The HAP specification describes the physical device configuration of products recognised as hearing aids. The profile defines four different configurations of Acceptor(s), all of which are included in the general term “hearing aid”. These are:

• A single hearing aid, which renders a single Bluetooth LE Audio Stream,

• A single hearing aid that receives separate left and right Audio Streams and combines the decoded audio into a single Audio Channel,

• A pair of hearing aids (also called a Binaural Hearing Aid Set) which are the members of a Coordinated Set, supporting individual left and right Audio Channels for each ear, and

• A hearing aid that uses a single Bluetooth link but provides separate left and right audio outputs to each ear. These are called Banded Hearing Aids, where there is a wired connection between the hearing aid device in each ear and the Bluetooth transceiver, which is typically in a neckband or an over-the-head band.

The Hearing Access Profile defines four different roles for the hearing aid ecosystem:

• HA (Hearing Aid), which is any one of the four types of hearing aid described above.

• HAUC (Hearing Aid Unicast Client), which is an Initiator which can establish a unicast Audio Stream with a hearing aid. The unicast Audio Streams can be unidirectional or bidirectional.

• HABS (Hearing Aid Broadcast Sender), which is an Initiator transmitting broadcast Audio Streams, and

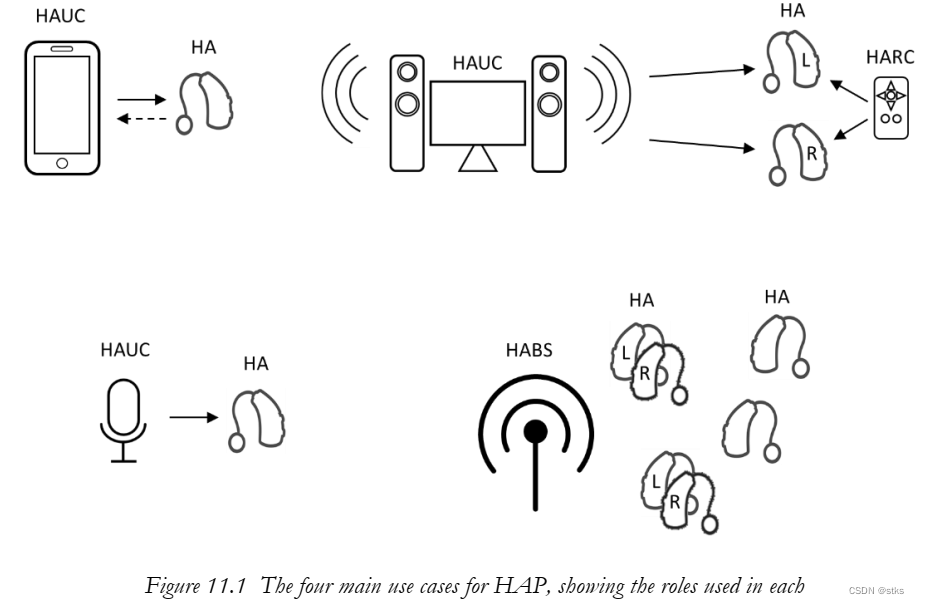
• HARC (Hearing Aid Remote Controller), which is a device that controls volume levels, mute states and hearing aid presets.

HAP and HAS introduce a new concept, which is support for Presets. Presets are proprietary audio processing configurations which hearing aid manufacturers include to optimise the sound fed to the ear. Typically, they adjust the processing to cope with different environments, such as restaurants, shops, office, home, etc. HAP and HAS don’t attempt to standardise these settings, but provide a numbering scheme which manufacturers can map to their specific implementation. Users can then select a specific preset by its number, or cycle through them. It includes the ability to add Friendly Names, so that an application can display the current presets and other available presets. It also supports dynamic presets, where the availability of a preset may change depending on the status of the hearing aid. Presets are currently a feature which is specific to hearing aids.

Within the preset feature of HAS, there is an interesting option, which is likely to expand into other profiles. This is the ability to use a non-Bluetooth radio to relay a command from one hearing aid to the other without the need for a Commander to institute a procedure to other set members based on a notification from one of them. This is the Synchronized Locally operation, which informs a Commander that any preset command sent to one hearing aid can be locally relayed to the other member of the set.

The other enhancement in HAP is the option to include the Immediate Alert Service [HAP 3.5.3]. This allows a device which does not support an Audio Stream to provide an alert to the Hearing Aid, which it can render as an alert to the user. No specific use cases are defined, but it is suggested that manufacturers might want to include this capability in products such as doorbells or microwave ovens to help inform hearing aid wearers of something they need to respond to.

The aim of the HAP requirements is to support all of the use cases envisaged for hearing aids, which are illustrated in Figure 11.1.



## HAP

The purpose of the Hearing Access Profile (HAP) is to define the requirements for an interoperable experience for users of hearing aids and complementary products in the hearing aid ecosystem that use Bluetooth Low Energy (LE) Audio technology with the Isochronous Channels feature.

To accomplish this, the HAP mandates requirements for products within the hearing aid ecosystem.

HAP specifies configurations and settings of parameters and procedures that are defined in lower-level specifications.

HAP specifies the following:

• HAP profile roles

• Procedures used with the Hearing Access Service (HAS)

• Requirements imposed on the other LE Audio profiles supported by HAP for interoperability purposes

HAP defines the following profile roles:

1. Hearing Aid (HA). The HA role is implemented in a hearing aid. An HA can be a Monaural Hearing Aid, a Banded Hearing Aid, or a member of a Binaural Hearing Aid Set.
2. Hearing Aid Unicast Client (HAUC). The HAUC role is implemented in a device capable of sending and optionally receiving unicast audio data to/from a hearing aid in the HA role. Examples of devices implementing the HAUC role are smartphones, tablets, laptops, personal microphones, music players, and voice recognition gateways.
3. Hearing Aid Remote Controller (HARC). The HARC role is implemented in a device that controls volume levels of the different audio sources (e.g., ambient sound, telecoil, audio streams), mute state of a microphone, and presets of a hearing aid in the HA role. The HARC may also implement the functionalities required to assist an HA in synchronizing to a broadcast audio stream. This role can be implemented on remote controllers but may also coexist with the HAUC role on devices such as smartphones and tablets that wish to control the functionalities of the HA. If the HARC is the only HAP role implemented on a device, that device may not support the LE Isochronous Channels features.
4. Immediate Alert Client (IAC). A device implements the IAC role to capture the attention of the HA wearer. Examples of devices implementing the IAC role are smartphones, tablets, laptops, and home appliances.

Hearing Aid role requirements：

1. An HA may optionally support the following feature： Volume balance means that different volume levels may be applied to left and right hearing aids. Volume balance and volume settings handled by VCP are generally independent of any audio logical settings, which are manufacturer specific.
2. The HA shall support the LE 2M PHY (Physical Layer), Low Energy transport.
3. Service UUIDs AD Type

If the HA is in one of the Generic Access Profile (GAP) discoverable modes, the HA shall include the Hearing Access Service Universally Unique Identifier (UUID) in the Service UUID Advertising Data (AD) Type field of the advertising data or scan response data.

If the HA is in one of the GAP connectable modes and is using a resolvable private address, the HA shall not include the Hearing Access Service UUID in the Service UUID AD type field of the advertising data or scan response.

1. Appearance AD Type

If the HA is in one of the GAP discoverable modes, the HA may include the value of the Appearance characteristic in its advertising data or scan response data for an enhanced user experience.

If the HA is in one of the GAP connectable modes and is using a resolvable private address, the HA shall not include the value of the Appearance characteristic in its advertising data or scan response data.

1. Hearing Access Service requirements:
2. HAS requirements: The HA shall instantiate one and only one instance of the HAS
3. Battery Service requirements: If the HA is equipped with one or more batteries, the HA should instantiate the Battery Service (BAS)
4. Immediate Alert Service requirements: The HA may instantiate one instance of the Immediate Alert Service (IAS)
5. CSIP Set Member role requirements

If the HA is part of a Binaural Hearing Aid Set, the HA shall satisfy the following requirements:

• The HA shall instantiate an instance of Coordinated Set Identification Service (CSIS),

• The HA shall support the Coordinated Set Size characteristic,

• The HA shall set the value of the Coordinated Set Size characteristic to 0x02.

1. BAP Unicast Server role requirements
2. The HA shall support the BAP Audio Sink role.
3. The HA may support the BAP Audio Source role.
4. The HA shall support the Audio Capability Settings
5. The HA shall support a Presentation Delay range in the Codec Configured state that includes the value of 20ms.
6. BAP Broadcast Sink role requirements

The HA shall be capable of rendering audio streams that use a mandatory LC3 configuration of BAP no later than 20ms after the SDU Synchronization reference.

1. Volume Renderer role requirements

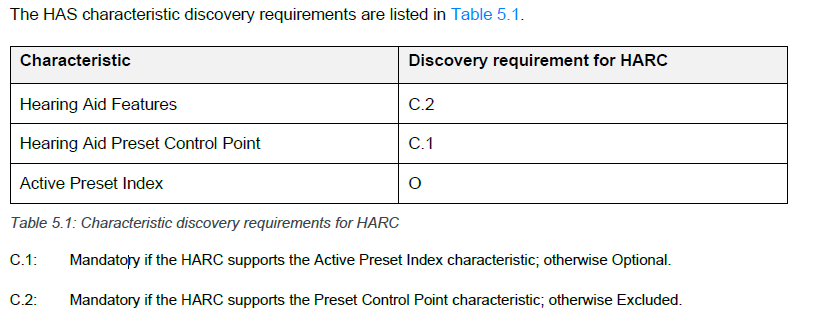
The Volume Control Service (VCS) is used to expose the volume level of the audio at the hearing aid speaker. How the Volume State characteristic affects the internal volume control of a hearing aid is implementation specific. It could act as the overall volume control or it could operate on other volume levels as well; for example, it could control only the volume level of a received audio stream.

1. MICP Device role requirements

If the HA supports the BAP Audio Source role, the HA shall expose one instance of the Microphone Input Control Service (MICS), and optionally one instance of AICS, to enable the peer device in the MICP Microphone Controller role to mute/unmute the microphone that picks up the user’s voice.

Hearing Aid Remote Controller role requirements

1. The HARC shall support the LE 2M PHY Low Energy transport.
2. The HARC shall discover the Hearing Access Service.
3. Characteristic discovery:



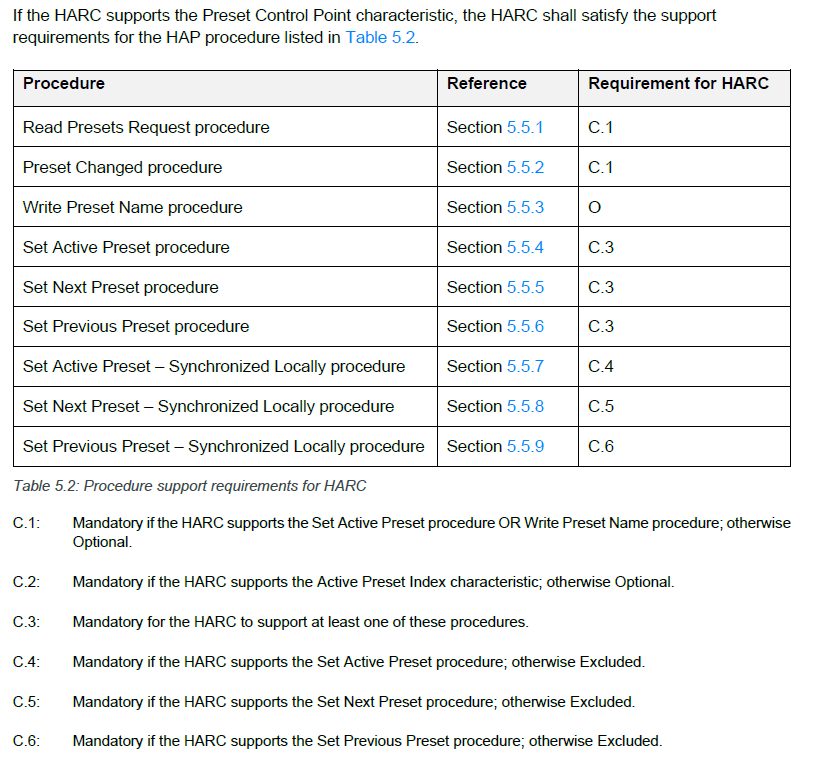
1. Hearing Aid Features characteristic requirements

If the HARC supports the Hearing Aid Features characteristic and if the HA supports notifications of this characteristic, then the HARC shall register for notifications.

1. Hearing Aid Preset Control Point characteristic requirements

If the HARC supports the Preset Control Point characteristic and the Read Preset procedure and an enhanced ATT bearer exists between the client and the server, then the HARC shall register for indications and may register for notifications of this characteristic.

If the HARC supports the Preset Control Point characteristic and the Read Preset procedure and only the unenhanced ATT bearer exists between the client and the server, then the HARC shall register for indications of this characteristic.



1. Active Preset Index characteristic requirements

If the HARC supports the Active Preset Index characteristic, then the HARC shall register for notifications of this characteristic.

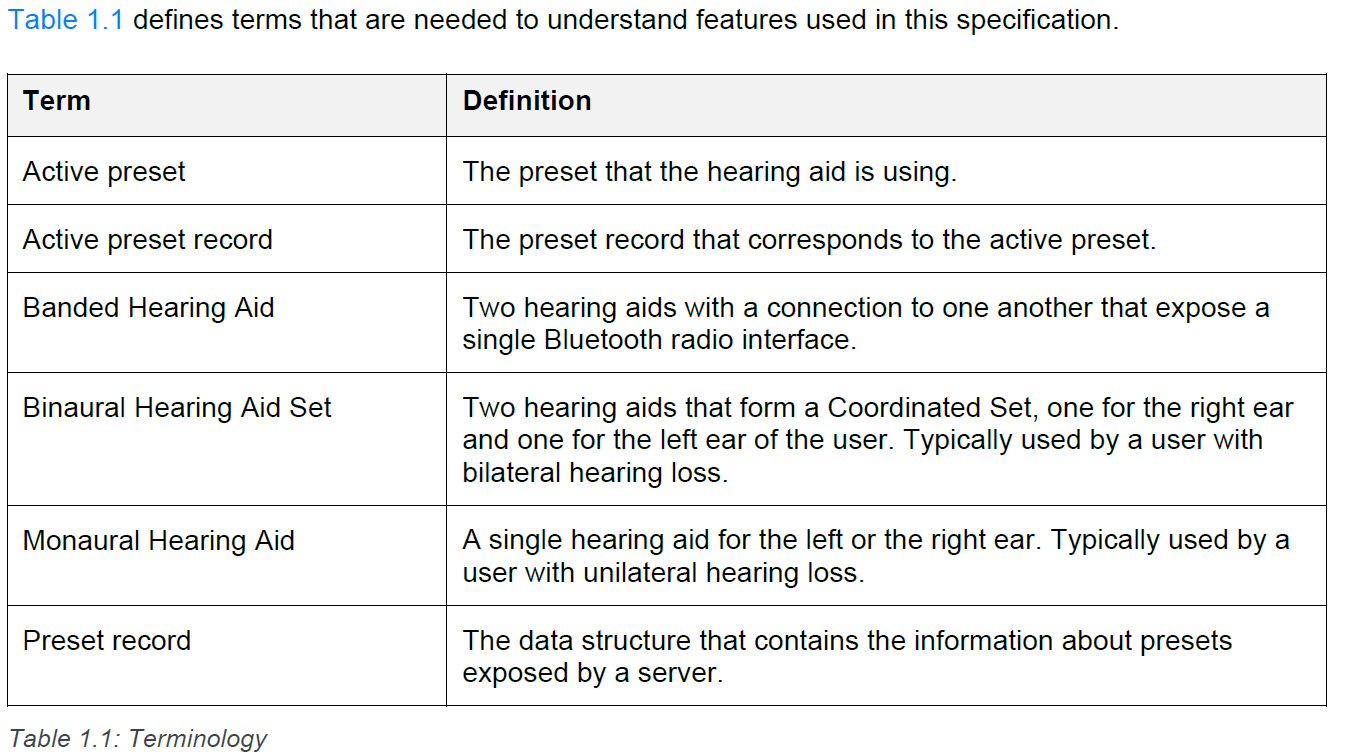
The HARC may use the Read Presets Request procedure to retrieve the name of the active preset after reading the Active Preset Index characteristic or receiving a notification of the Active Preset Index characteristic.

## HAS

The Hearing Access Service (HAS) is used to identify hearing aids and to control hearing aid presets. A hearing aid preset represents a configuration of the hearing aid signal processing parameters tailored to a specific listening situation.

The preset used by a hearing aid can be changed by user action on the hearing aid or on a client device (e.g., smartphone, remote controller), or autonomously by the hearing aid.

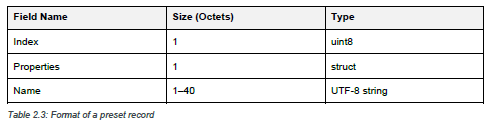
The hearing aid can modify the list of preset records by adding and deleting presets, changing their names, and changing their availability over time. As defined in this specification, client devices are informed about these changes.



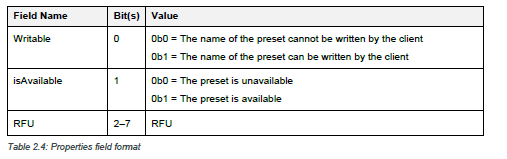
This service shall operate on the Bluetooth Low Energy (LE) transport only.

Preset record:

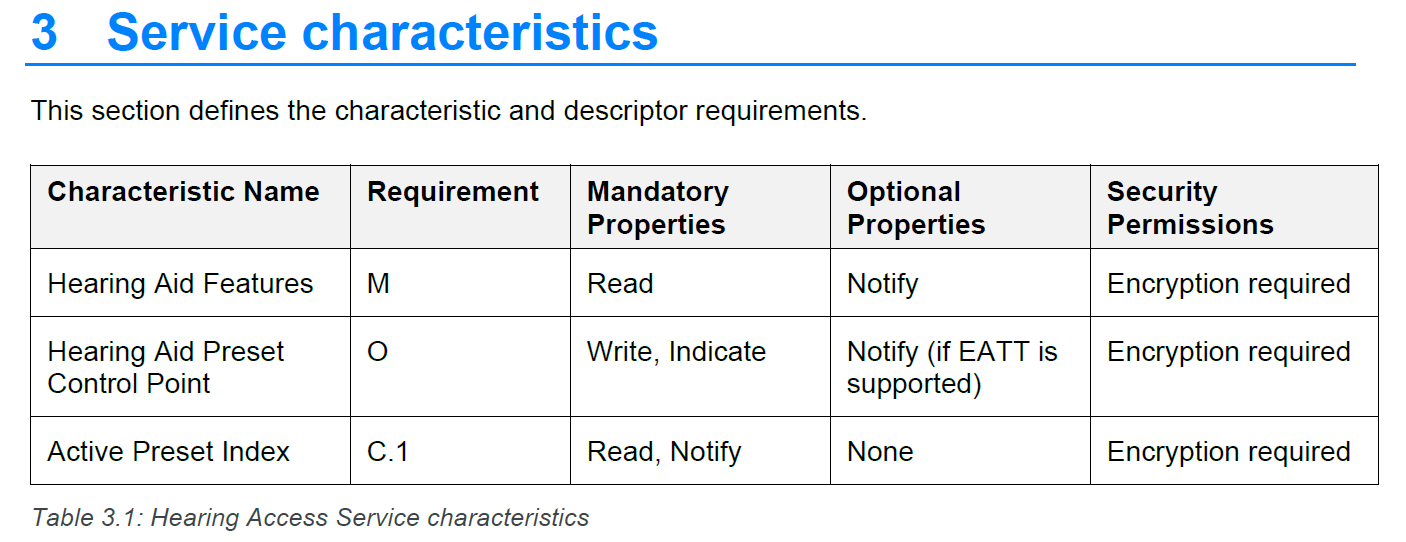
The preset record contains all the information about a hearing aid preset. The server shall maintain a list of preset records.



The Index field uniquely identifies a preset record in the list of preset records



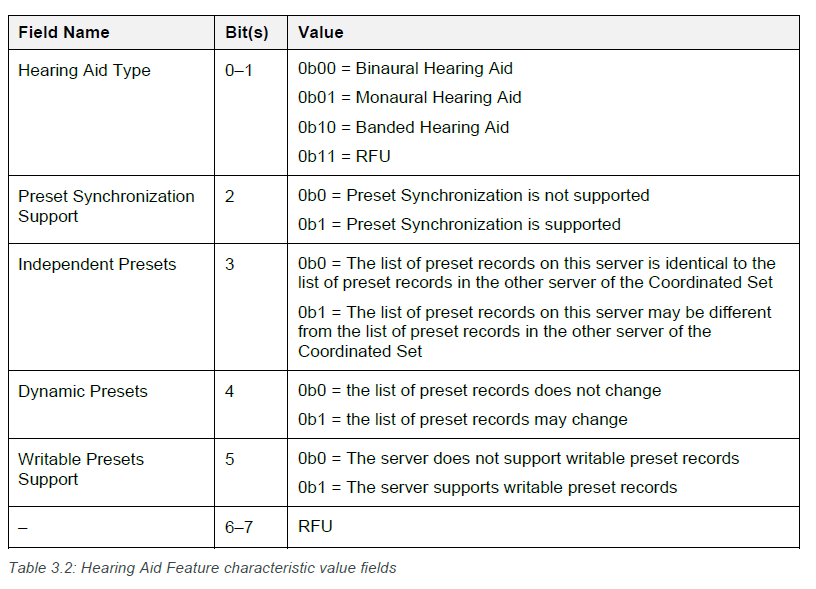
The Name field is a UTF-8 string that contains the human-readable name of the preset.



Hearing aid features：

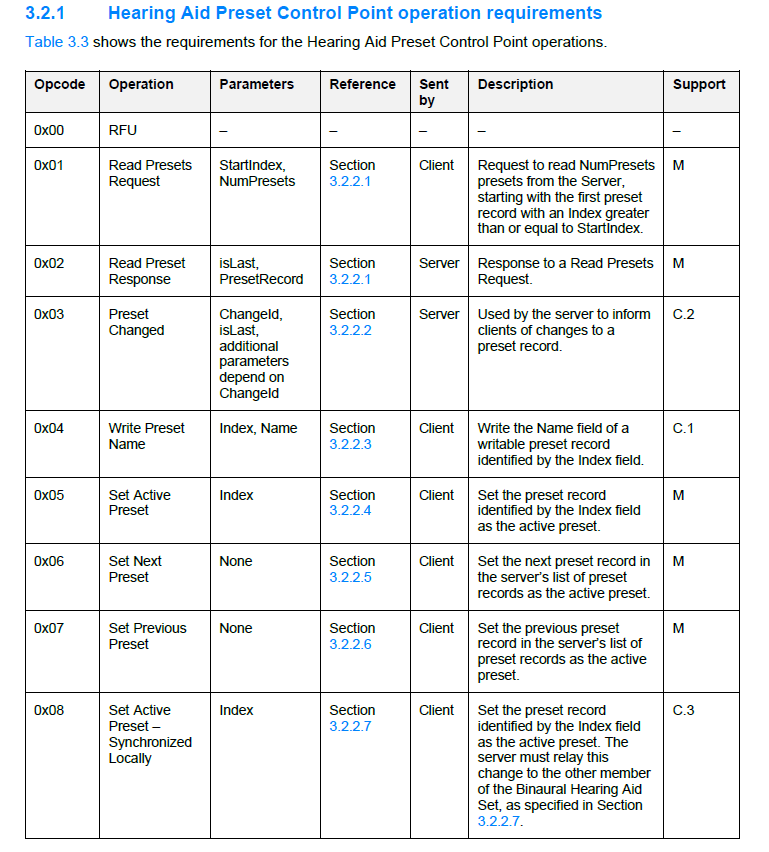
The value of the Hearing Aid Features characteristic exposes the features of the Hearing Access Service supported by the server.

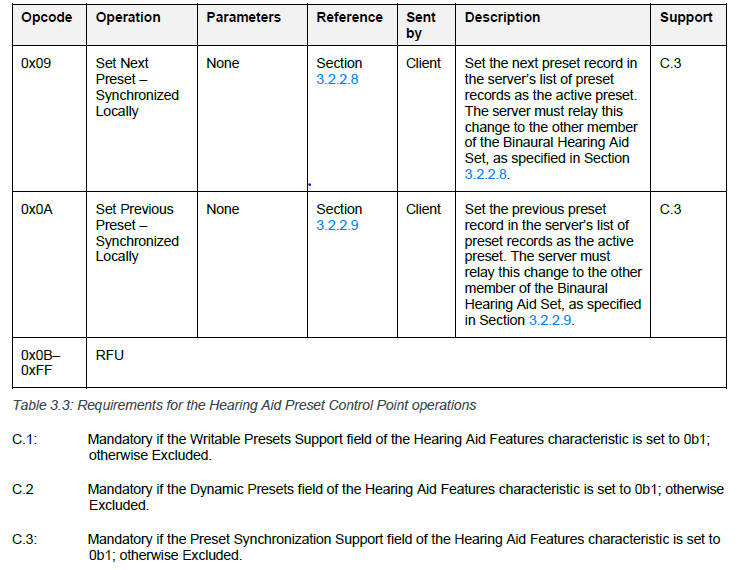
The format of the Hearing Aid Features characteristic is a one octet bitfield



Hearing Aid Preset Control Point characteristic：

A procedure is triggered by writing, notifying, or indicating a value of the Hearing Aid Preset Control Point characteristic that includes an opcode specifying the operation. The procedure might require a parameter that is valid within the context of that opcode. When a client writes a value, the server shall respond either by sending an ATT\_WRITE\_RSP PDU to confirm that it has accepted the request or by sending an ATT\_ERROR\_RSP with an appropriate ATT Error or ATT Application Error Code. One or more notifications or indications of the Hearing Aid Preset Control Point characteristic can follow the response depending on the operation requested by the client.





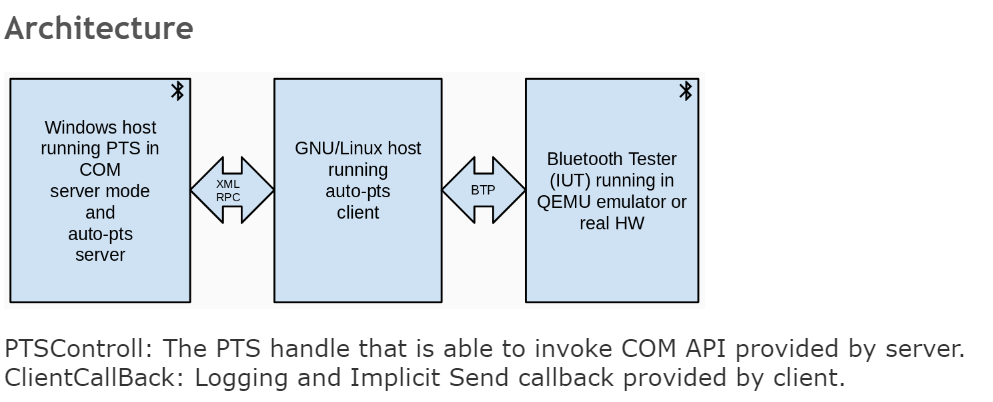
Active Preset Index characteristic：

The value of the Active Preset Index characteristic shall be equal to the Index field of the Active preset record. If no preset is active, the Active Preset Index characteristic value shall be set to 0x00. The server shall not set the value of the Active Preset Index characteristic to the Index field of a preset record with the isAvailable bit of the Properties field set to 0b0 (i.e., an unavailable preset record).

## PTS

Bluetooth Profile Tuning Suite(PTS) is a powerful, software-based black-box testing tool that automates protocol and profile interoperability testing, reduing Bluetooth SIG member's costs and time in their product development, testing and qualification processes.

auto-pts is the Bluetooth PTS automation framework. auto-pts uses PTSControl COM API of PTS to automate testing.



auto-pts server: Implemented in Python 3. Runs on Windows and provides over-the-network XML-RPC interface to PTS.

1. use win32com to get PTS API handle then could invoke COM API to open workspace, run test case.etc.
2. provide XML-RPC server and register PTSControll as the instance.
3. as a XML-RPC client and get the proxy of ClientCallBack

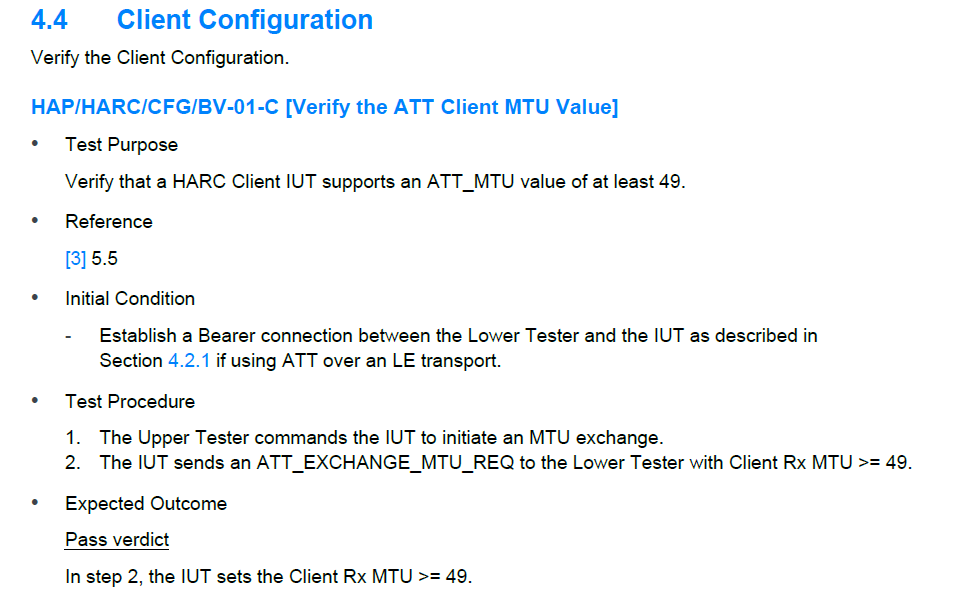
auto-pts client: Implemented in Python 3. Runs on GNU/Linux or Windows, communicates with the auto-pts server (to start/stop test cases, to send response to PTS inquiries) and communicates with the Implementation Under Test to take appropriate actions.

1. as a XML-RPC client and get the proxy of PTSControl
2. provide XML-RPC server and register ClientCallBack as the instance.
3. communicate to IUT with Bluetooth test protocol(BTP)
4. organize and execute test cases

Implementation Under Test (IUT): It is the host running Bluetooth stack to be tested, this could be an emulator or real hardware. The IUT is controlled by using Bluetooth Test Protocol.

1. implement BTP
2. get request from autopts client and return response

Bluetooth Test Protocol (BTP): Used to communicate with the IUT



<http://jenkins-spsd.verisilicon.com/image/LEAudio_daily_test/20230625_1228_build60/test/>

Windows 给linux发送命令，想要实现什么function，他们之间有个wid， Linux 端收到这个wid之后就会有默认的function，如果解这些case，就会看linux 端python的log和btp的log， Linux端默认的function可能不满足这个case，或者是板子内部的image 使得Linux发送的btp命令没有得到正确的响应，那么就需要修改function和板子的image，如果这两个地方都没有问题的话，就可以用sniffer来记录板子和dongle端的OTA 日志。

流程是：

Windows端把PTS打开， Linux端和Windows端进行交互，告诉Windows要执行哪些cases，windows端把case运行起来，执行该case需要哪些命令，windows发送linux wid命令，然后linux有对应的function，然后给板子发送btp命令，板子发送得到对应的btp命令之后跟windows上的dongle进行交互，并且给linux响应，windows接收dongle controller的event，然后pts会根据流程来判断case是否是pass的.

Windows 上的pts会接受linux传来的case的参数，wid 是PTS软件提供的，pts软件会有自己的一系列检查流程，可以看日志观察是哪个wid出了问题，如果出了问题，去看相应的wid的函数，是否符合预期，检查参数是否有问题，调用的函数是否有问题。一个wid会有固定的函数。主要是去修改这些函数的实现。

LE Audio的这些case其实跟之前的L2CAP 的case的测试过程是一样的，只是相当于在之前的L2CAP的基础上再包了一层，本质上还是用软件来实现。咋们可以简单的看看这个case，这就是client端的日志，可以看到在跑case之前，就会先得到运行这些case需要哪些commands, 接下来就是针对每一个command里面的function，执行一些功能，比如说简单的reset板子，给板子发了reset的命令，板子的回复，这些都是差不多的，命令的交互，从这个btp的log可以看到板子的回复，对比是否正确，然后就是client得到要跑的这个case的wid，一些描述，确认这个case的参数之后发送response到server端，server端就开始去跑case，针对每个wid，里面就会有一些函数去实现，这个时候client可以通过server端的callback看到每个case的检查过程，这里面就是通过蓝牙发送了哪些command，得到了哪些event，每一对command和event的结果是什么，pts会去判断这些过程，同时开发也会去根据这些日志来解bug。

从这个case的日志中我们就可以看到就是这么个流程，然后会发现现在HAP的case还没有解嘛，所以这个wid都还没有实现，观察具体是在哪一步有问题，这样就可以在wid函数里面去实现相应的功能。

Auto-pts:

Server 端：init\_server, recovery\_dongle, 还有一个superguard 超时设置，如果超时，就利用ykush recovery dongle, get the proxy of ClientCallBack, 接收client端的request。

init\_pts(包括client拿到server的代理，open workspace，update PIXIT param 然后start PTS )

btp\_init, get iut,也就是我们的被测设备

init\_stack: init ble stack

setup\_test\_cases: 选择哪些case要跑

run\_test\_cases—> run\_test\_case\_thread\_entry—>pre\_run,可以看client端的日志，比如这个case，可以看到执行这个case需要哪些commands， 针对每一个command， 就会去具体的function里面看具体又执行了什么，比如start， flush\_serial, start socket, 然后又从板子处得到accept的message， 可以看到client端会一步一步执行这些命令，可以看到client给板子发送的btp命令，以及可以从板子处得到的response， 同时可以结合着btp的这个log，可以看到给板子发送的命令，然后板子回了什么，这些都是在运行PTS之前的一些行为，比如reset之类的一些操作。

其次又回这个主程序，set\_current\_test\_case client 端收到pts callback，可以看到有wid这些东西，同时还会进入到具体的bap\_wid\_hdl, 然后就会发送response到server，告诉server可以开始运行了，同时client端还可以收到这个case涉及到的一些蓝牙命令，以及pts在ptscontrol里面对于每一步的判断。pts软件判断这个case是否是pass的。