

Out-of-Band Pairing Uniform Resource Identifier

Bluetooth® Specification

- **Version:** v1.0
- **Version Date:** 2026-09-28
- **Prepared By:** Core Specification Working Group

Abstract:

This specification defines how a Uniform Resource Identifier (URI) obtained through an out-of-band (OOB) medium can be used to convey information needed for a device to perform Bluetooth® OOB pairing with another device that is associated with the URI. This specification shows how this information can be used with a Quick Response (QR) code or other OOB methods to help guide the selection of a peer device to pair with, and to streamline the pairing process with no additional user interactions.

Version History

Version Number	Date (yyyy-mm-dd)	Comments
v1.0	2026-09-28	Adopted by the Bluetooth SIG Board of Directors.

Acknowledgments

Name	Company
Hai Shalom	Google LLC
Nick Hunn	GN Hearing A/S
Toby Nixon	Bluetooth SIG, Inc.
Mayank Batra	Qualcomm Technologies, Inc.
Kim Schulz	Samsung Electronics Co., Ltd.
Pål Håland	Nordic Semiconductor ASA
Andrew Estrada	Sony Group Corporation
Hai Shalom	Apple Inc.

Use of this specification is your acknowledgement that you agree to and will comply with the following notices and disclaimers. You are advised to seek appropriate legal, engineering, and other professional advice regarding the use, interpretation, and effect of this specification.

Use of Bluetooth specifications by members of Bluetooth SIG is governed by the membership and other related agreements between Bluetooth SIG and its members, including those agreements posted on Bluetooth SIG's website located at www.bluetooth.com. Any use of this specification by a member that is not in compliance with the applicable membership and other related agreements is prohibited and, among other things, may result in (i) termination of the applicable agreements and (ii) liability for infringement of the intellectual property rights of Bluetooth SIG and its members. This specification may provide options, because, for example, some products do not implement every portion of the specification. All content within the specification, including notes, appendices, figures, tables, message sequence charts, examples, sample data, and each option identified is intended to be within the bounds of the Scope as defined in the Bluetooth Patent/Copyright License Agreement ("PCLA"). Also, the identification of options for implementing a portion of the specification is intended to provide design flexibility without establishing, for purposes of the PCLA, that any of these options is a "technically reasonable non-infringing alternative."

Use of this specification by anyone who is not a member of Bluetooth SIG is prohibited and is an infringement of the intellectual property rights of Bluetooth SIG and its members. The furnishing of this specification does not grant any license to any intellectual property of Bluetooth SIG or its members. **THIS SPECIFICATION IS PROVIDED "AS IS" AND BLUETOOTH SIG, ITS MEMBERS AND THEIR AFFILIATES MAKE NO REPRESENTATIONS OR WARRANTIES AND DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, TITLE, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR THAT THE CONTENT OF THIS SPECIFICATION IS FREE OF ERRORS.** For the avoidance of doubt, Bluetooth SIG has not made any search or investigation as to third parties that may claim rights in or to any specifications or any intellectual property that may be required to implement any specifications and it disclaims any obligation or duty to do so.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, BLUETOOTH SIG, ITS MEMBERS AND THEIR AFFILIATES DISCLAIM ALL LIABILITY ARISING OUT OF OR RELATING TO USE OF THIS SPECIFICATION AND ANY INFORMATION CONTAINED IN THIS SPECIFICATION, INCLUDING LOST REVENUE, PROFITS, DATA OR PROGRAMS, OR BUSINESS INTERRUPTION, OR FOR SPECIAL, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, AND EVEN IF BLUETOOTH SIG, ITS MEMBERS OR THEIR AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF THE DAMAGES.

Products equipped with Bluetooth wireless technology ("Bluetooth Products") and their combination, operation, use, implementation, and distribution may be subject to regulatory controls under the laws and regulations of numerous countries that regulate products that use wireless non-licensed spectrum. Examples include airline regulations, telecommunications regulations, technology transfer controls, and health and safety regulations. You are solely responsible for complying with all applicable laws and regulations and for obtaining any and all required authorizations, permits, or licenses in connection with your use of this specification and development, manufacture, and distribution of Bluetooth Products. Nothing in this specification provides any information or assistance in connection with complying with applicable laws or regulations or obtaining required authorizations, permits, or licenses.

Bluetooth SIG is not required to adopt any specification or portion thereof. If this specification is not the final version adopted by Bluetooth SIG's Board of Directors, it may not be adopted. Any specification adopted by Bluetooth SIG's Board of Directors may be withdrawn, replaced, or modified at any time. Bluetooth SIG reserves the right to change or alter final specifications in accordance with its membership and operating agreements.

Copyright © 2026. All copyrights in the Bluetooth Specifications themselves are owned by Apple Inc., Ericsson AB, Intel Corporation, Google LLC, Lenovo (Singapore) Pte. Ltd., Microsoft Corporation, Nokia Corporation, and Toshiba Corporation. The Bluetooth word mark and logos are owned by Bluetooth SIG, Inc. Other third-party brands and names are the property of their respective owners.

Contents

1	Introduction.....	6
1.1	Language	6
1.1.1	Language conventions	6
1.1.2	Reserved for Future Use	7
1.1.3	Prohibited	7
1.2	Requirements in tables.....	8
1.2.1	Table requirements for unused cells.....	8
1.3	Conformance.....	8
2	Using an OPU to initiate OOB pairing	9
2.1	Discovering and selecting Bluetooth devices to pair with	9
2.2	Secure OOB pairing.....	10
2.3	An OOB approach	10
3	Configuration.....	13
3.1	Roles	13
3.1.1	OPU Encoder role	13
3.1.2	OPU Decoder role.....	13
3.2	Core Configuration dependencies.....	13
4	OPU principle of operation.....	14
5	The OPU	17
5.1	The OPU and its relation to the Bluetooth Core Specification.....	17
5.2	Bluetooth OOB pairing scheme components	17
5.3	OPU data format usage	18
5.4	Formatting the OPU components	19
5.5	Capturing and parsing the OPU elements	19
5.6	Generic Access Profile requirements	20
5.6.1	Requirements for OPU on the LE transport.....	20
5.6.2	Requirements for OPU on the BR/EDR transport.....	21
5.7	Extended OPU components.....	21
5.7.1	Rules for the vendor-specific component.....	23

5.8	OPU message structure in augmented Backus–Naur form	24
6	Security requirements.....	26
6.1	Random number	26
6.2	BR/EDR devices with Secure Connections.....	26
6.3	LE devices with Secure Connections	26
6.4	BR/EDR/LE devices	27
7	Examples of OPUs as QR codes	28
7.1	Example 1	28
7.2	Example 2	29
8	Acronyms and abbreviations	31
9	References	33

1 Introduction

This OOB Pairing URI (OPU) specification defines and provides guidance on how a Bluetooth® URI can assist users to seamlessly pair devices.

1.1 Language

1.1.1 Language conventions

In the development of a specification, the Bluetooth SIG has established the following conventions for use of the terms “*shall*”, “*mandatory*”, “*shall not*”, “*should*”, “*should not*”, “*may*”, “*optional*”, “*must*”, and “*can*”. In this Bluetooth specification, the terms in [Table 1.1](#) have the specific meanings given in that table, irrespective of other meanings that exist.

Term	Definition
shall or mandatory	—used to express what is required by the specification and is to be implemented exactly as written without deviation
shall not	—used to express what is forbidden by the specification
should or may or optional	—not mandatory. Used to express either: - what is recommended by the specification without forbidding anything (“should”) - what is permissible within the limits of the specification (“may” or “optional”)
should not	—used to indicate that something is discouraged but not forbidden by the specification
must	—used to indicate either: - an indisputable statement of fact that is always true regardless of the circumstances - an implication or natural consequence if a separately-stated requirement is followed
can	—used to express a statement of possibility or capability

Table 1.1: Language conventions terms and definitions

Where more than one item is permitted but not required, the choices to include or support those items are independent from one another unless the specification explicitly states otherwise. Each item that is implemented shall be implemented exactly as written without deviation.

1.1.1.1 Implementation alternatives

When specification content indicates that there are multiple alternatives to satisfy specification requirements, if one alternative is explained or illustrated in an example it is not intended to limit other alternatives that the specification requirements permit.

1.1.1.2 Discrepancies

It is the goal of Bluetooth SIG that specifications are clear, unambiguous, and do not contain discrepancies. However, members can report any perceived discrepancy by filing an erratum and can request a test case waiver as appropriate.

1.1.2 Reserved for Future Use

Where a field in a packet, Protocol Data Unit (PDU), or other data structure is described as "Reserved for Future Use" (irrespective of whether in uppercase or lowercase), the device creating the structure shall set its value to zero unless otherwise specified. Any device receiving or interpreting the structure shall ignore that field; in particular, it shall not reject the structure because of the value of the field.

Where a field, parameter, or other variable object can take a range of values, and some values are described as "Reserved for Future Use," a device sending the object shall not set the object to those values. A device receiving an object with such a value should reject it, and any data structure containing it, as being erroneous; however, this does not apply in a context where the object is described as being ignored or it is specified to ignore unrecognized values.

When a field value is a bit field, unassigned bits can be marked as Reserved for Future Use and shall be set to 0. Implementations that receive a message that contains a Reserved for Future Use bit that is set to 1 shall process the message as if that bit was set to 0, except where specified otherwise.

The acronym RFU is equivalent to Reserved for Future Use.

1.1.3 Prohibited

When a field value is an enumeration, unassigned values can be marked as "Prohibited." These values shall never be used by an implementation, and any message received that includes a Prohibited value shall be ignored and shall not be processed and shall not be responded to.

Where a field, parameter, or other variable object can take a range of values, and some values are described as "Prohibited," devices shall not set the object to any of those Prohibited values. A device receiving an object with such a value should reject it, and any data structure containing it, as being erroneous.

"Prohibited" is never abbreviated.

1.2 Requirements in tables

Requirements in this specification are defined as "Mandatory" (M), "Optional" (O), "Excluded" (E), "Not Applicable" (N/A), or "Conditional" (C.#). Conditional statements (C.#) are listed directly below the table in which they appear.

- "M" for mandatory. See [Table 1.1: Language conventions terms and definitions](#).
- "O" for optional. See [Table 1.1: Language conventions terms and definitions](#).
- "E" for excluded. "Excluded" means not permitted in this context; cannot be supported or included for this purpose. The item can still be supported or included if allowed for some other purpose (e.g., a feature can be mandatory for one role and excluded for another; a device that supports both roles must support this feature).
- "C.#" for conditional. "Conditional" means that an item is required, optional, or prohibited based on whether one or more other items are supported or included (# represents any number). Within the definition of the condition, if those other items mean that "not permitted" applies, it has the same meaning as "E".

1.2.1 Table requirements for unused cells

An unused cell is a table cell without a value or content, which is indicated with either the word "none" (without quotation marks) or a hyphen (i.e., a "minus" sign).

1.3 Conformance

Each capability of this specification shall be supported in the specified manner. This specification may provide options for design flexibility, because, for example, some products do not implement every portion of the specification. For each implementation option that is supported, it shall be supported as specified.

2 Using an OPU to initiate OOB pairing

This specification specifies the use of a URI [1] to assist users to seamlessly pair with new devices. This pairing technique simplifies the user experience by allowing users to scan a QR code [5] or use any other OOB mechanism to obtain the URI, which provides the user's device all the information it requires to discover and pair with another Bluetooth device that is in either bondable mode or non-bondable mode (see Volume 3, Part C, Section 4.3 in [4]).

Information in the OPU can be used to select a specific device for pairing without the need to choose the correct device from a Bluetooth settings page. For example, a QR code could be used to select the Peripheral device intended to be paired in an environment that contains advertisements from multiple devices.

The OPU contains information that can authenticate a peer device without the need to enter a passkey or compare numbers, and the Man-in-the-Middle (MITM) resistance depends on the security properties of the OOB method in use.

This specification provides examples of how to use this feature, specifying the URI components that need to be included. This specification includes standardized formatting details to expose this information in the form of an OPU. Sections 2 and 7 of this specification provide examples of the OPU being used to enable interactions using QR codes, Section 3 defines the OPU Encoder and OPU Decoder roles, Section 4 provides an overview of the OPU principle of operation, and Section 5 defines the GAP requirements and the format of the OPU. Section 6 defines the security requirements for devices implementing this specification.

2.1 Discovering and selecting Bluetooth devices to pair with

The Bluetooth Core Specification [4] defines how Peripheral devices can be put in discoverable and bondable mode, and how Secure Connections Pairing is performed over the air.

Figure 2.1 shows one possible implementation, when a user needs to select one device to pair with from the list of devices that are advertising and listed in the Bluetooth settings page.

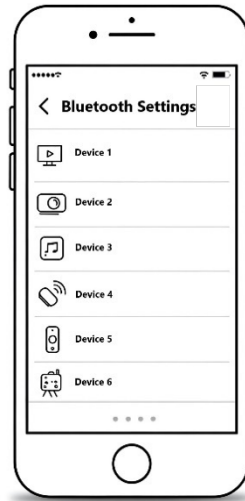


Figure 2.1: An example of using the Bluetooth settings page to select the correct device to pair with

As the number of Bluetooth devices continues to increase, the list of discovered devices in the Bluetooth settings page also continues to grow, often with some meaningless or confusing device names. The user selection task may become complex and can become more complicated if there are multiple similar devices advertising in the area.

2.2 Secure OOB pairing

In addition to the device selection complexity, an additional risk is associated with the long list of discovered devices. An attacker can set up an evil twin device (i.e., a device with the same appearance value and the same or similar device name to the device the user is trying to use) to lure unsuspecting users and perform an MITM attack.

Such an attack can occur when using “Just Works” pairing. Because the user has no passkey to type or number to compare, the user wouldn't know that they are connected to an attacker instead of the intended device. The resulting connection is encrypted, but its endpoint might be malicious. Furthermore, some users might ignore or skip the prompt to compare numbers and thus approve the connection without verifying that both devices display the same number, resulting in a similar outcome.

The Secure Connections OOB pairing method is considered secure, provided the OOB channel itself is secure. If the OOB channel is compromised, then the pairing can be, too.

2.3 An OOB approach

To address the challenges mentioned in Sections 2.1 and 2.2, OOB methods can be useful in new and existing use cases. One way of using OOB methods is with QR codes. A user can use the camera on their smartphone to scan a QR code, capturing information to instruct their smartphone to select a specific device for pairing.

Figure 2.2 shows how OPU can be implemented in a car display. A dynamically generated QR code lets the owner of the car scan the code, which then directs their smartphone to seamlessly and securely

perform pairing with their car's Bluetooth system. Once pairing starts, the QR code disappears from the screen; it will not be reused.

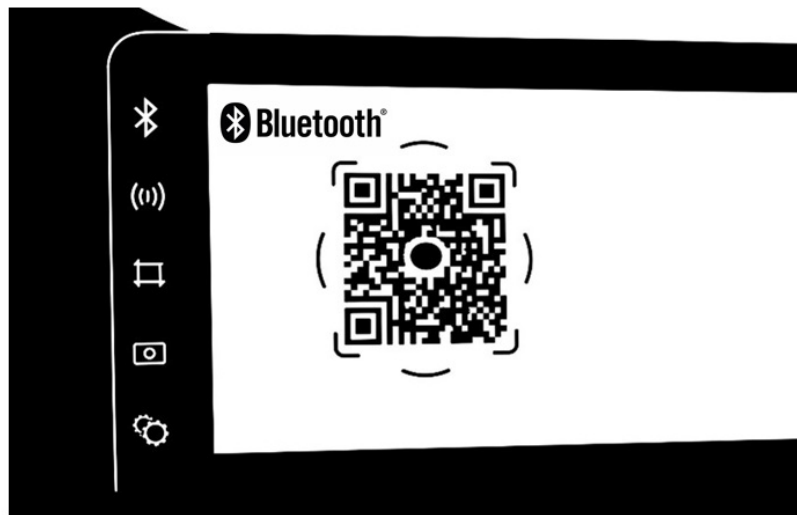


Figure 2.2: Using a dynamically generated QR code to pair a smartphone with a car

Figure 2.3 shows a different example, where a dynamically generated QR code is displayed by a smart Bluetooth speaker and alarm clock in a hotel room. A new guest can scan the QR code and seamlessly start streaming their favorite music without the risk of accidentally choosing the speaker in the adjacent room, or the risk of the guest in the adjacent room pairing with the speaker in the new guest's room. Once pairing starts, the QR code disappears from the screen; it will not be reused.



Figure 2.3: Using a QR code to pair with a smart Bluetooth speaker and alarm clock in a hotel room

These QR codes provide specific information that a smartphone can use for the pairing process, simplifying the selection experience and securing the pairing process.

3 Configuration

3.1 Roles

This specification defines two roles:

3.1.1 OPU Encoder role

A device that implements the OPU Encoder role supports the OPU and can both generate a URI that can be used for Bluetooth OOB pairing and complete Bluetooth OOB pairing with a specific generated URI (e.g., the car screen example in [Figure 2.2](#)).

3.1.2 OPU Decoder role

A device that implements the OPU Decoder role supports the OPU and can acquire and parse the URI (e.g., a device that scans a QR code using a camera application), and then initiates Bluetooth OOB pairing with the information it acquired.

3.2 Core Configuration dependencies

There are no additional dependencies beyond minimal compliance with the Core Configuration requirements.

4 OPU principle of operation

This specification uses the Secure Connections OOB pairing method as specified in Volume 2, Part H, Section 7.2.2 in [4] for BR/EDR, and Volume 3, Part H, Section 2.3.5.6.4 in [4] for LE.

This specification specifies a method for communicating OOB data over an OOB channel in one direction. The authentication of the device sending the OPU communication will be based on the sending device's commitment of the public key (C) included in the OPU, and the authentication of the device receiving the OPU communication will be based on the receiving device knowing a random number (r) included in the OPU, which must remain secret. Additional security requirements are specified in Section 6.

A device in the OPU Encoder role generates a random number, a public-private key pair, and a commit value. The OPU is then generated according to the data format rules specified in Section 5 (including any additional mandatory and optional parameters).

Once a device in the OPU Decoder role receives the OPU through an OOB channel, the device can start the standard OOB pairing method with the device that generated the OPU.

Figure 4.1 shows the message flows of OOB pairing with OPU over a BR/EDR transport. Figure 4.2 shows the message flows of OOB pairing with OPU over an LE transport. Non-initiating device B generates the OPU that includes the rb and Cb values (see Volume 2, Part H, Table 7.1 in [4]), a display name, and device B's address (additional parameters are required in the OPU itself). Device A receives the OPU through an OOB channel and starts the OOB pairing method with device B.

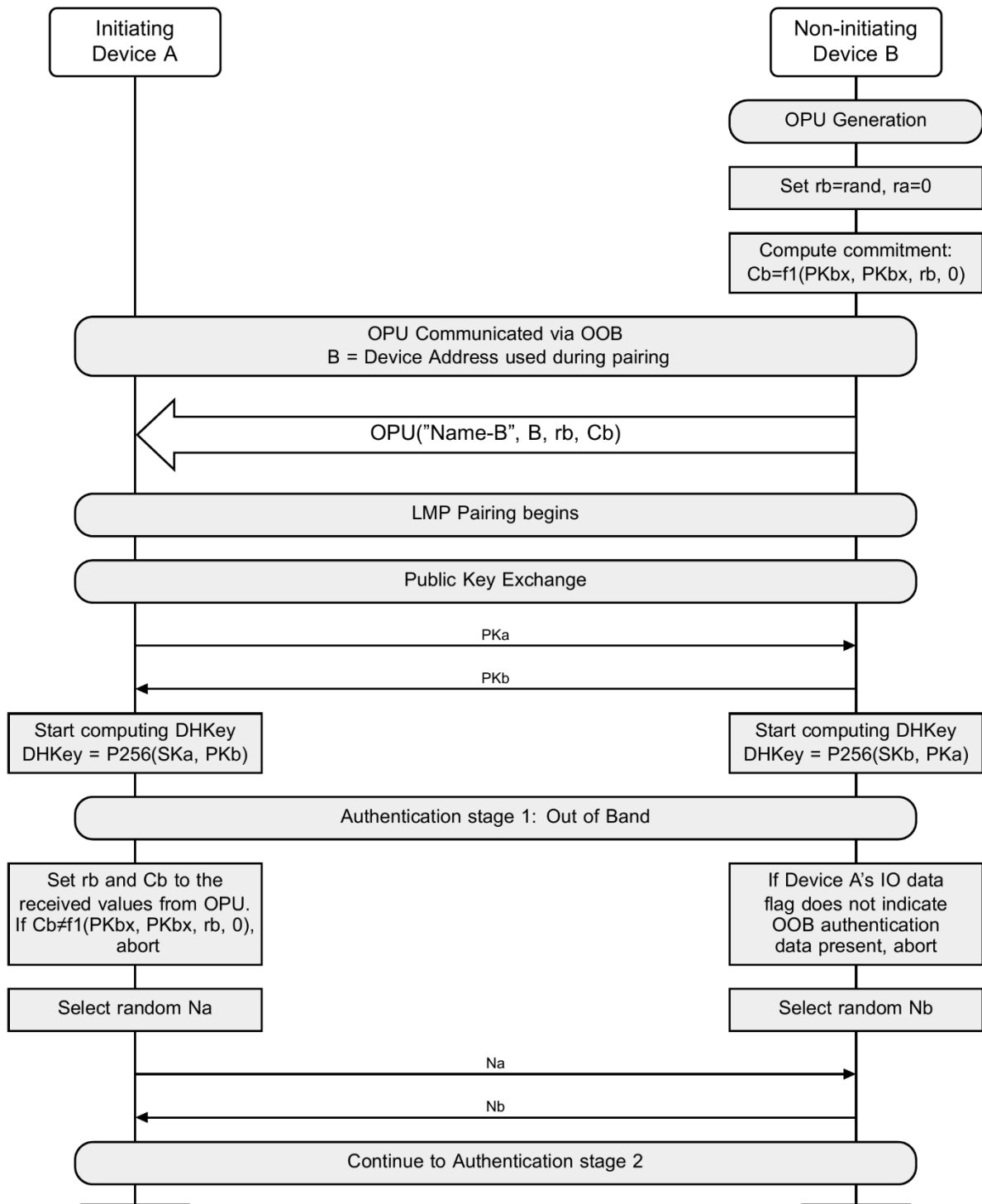


Figure 4.1: OOB pairing with OPU over BR/EDR transport

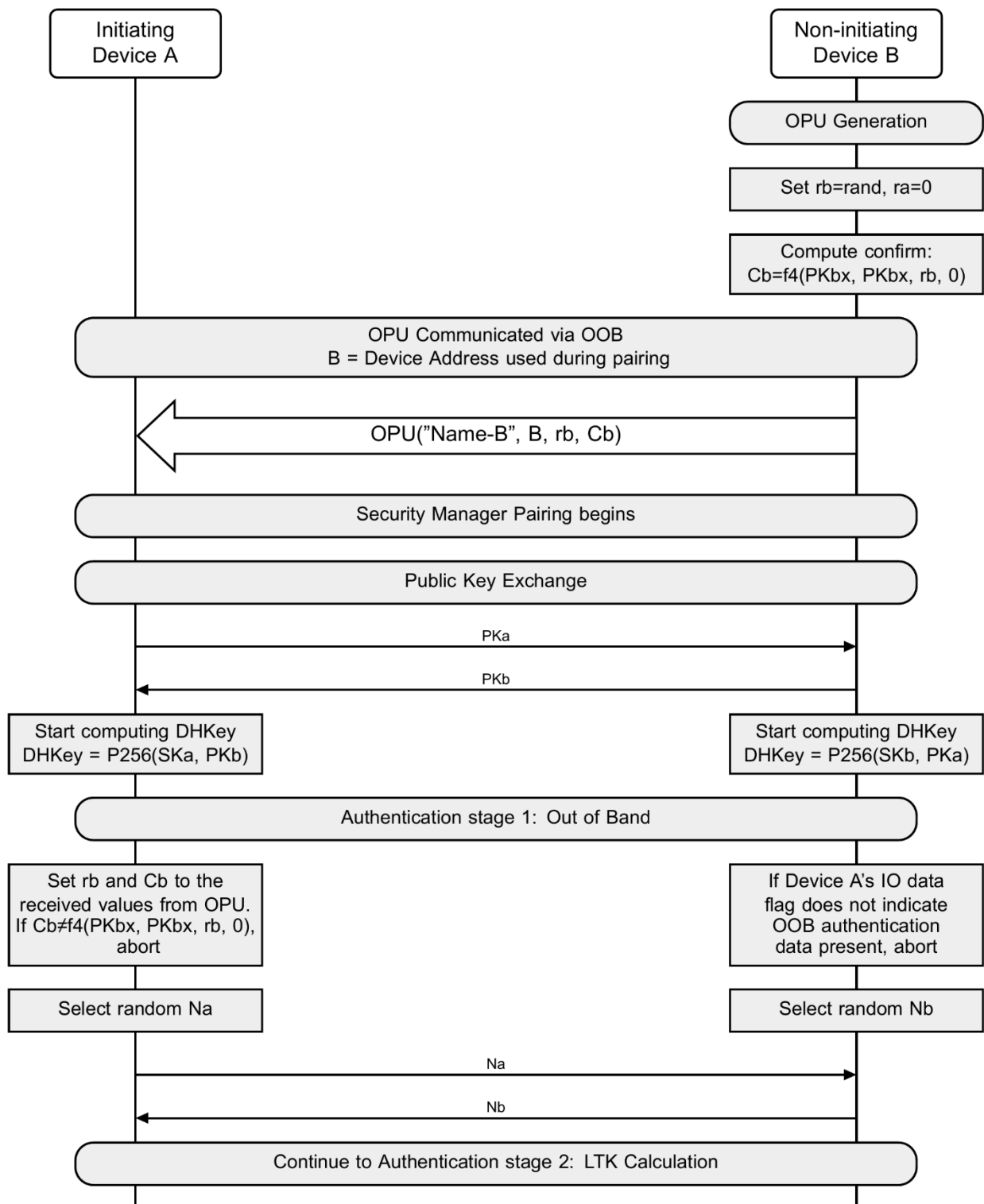


Figure 4.2: OOB pairing with OPU over LE transport

5 The OPU

To provide an interoperable experience, all URIs and other OOB methods shall present data required for Bluetooth OOB pairing in the same format. This data is formatted in a similar manner to the Uniform Resource Identifier (URI): Generic Syntax document [1], in the form of a string with the scheme prefix identifier of "BLUETOOTH:" to indicate the presence of Bluetooth data to the scanning device.

5.1 The OPU and its relation to the Bluetooth Core Specification

All the data conveyed is defined in the underlying Bluetooth Core Specification [4]. To differentiate the OPU from any other Bluetooth URI usages, the "BLUETOOTH:" scheme prefix shall be followed by an `identifier:data` pair signifying that it is a Core Specification OPU. The pair consists of a CORE identifier and the acronym name of this specification (OPU) as the data, in the string form of "CORE:OPU;". This combination of the Bluetooth scheme identifier, Bluetooth Core Specification feature name, and subsequent components shall be referred to as the OPU and is common for all OOB methods. Subsequent components may appear in any order.

5.2 Bluetooth OOB pairing scheme components

Multiple components need to be present in an OPU to direct a device to pair with a peer device. The components are listed in Table 5.1, which specifies the component details that shall be included in the OPU. The formats identified in the "Format" column are defined in Table 5.2.

Each component contains information specified in the Bluetooth Core Specification [4] and listed in the Bluetooth Core Specification Supplement [6].

Component	Req	ID	Format	Value	Reference
Display Name ^{Note 1}	M	DN	String	A UTF-8 (Unicode Transformation Format) string containing a minimum of 4 and a maximum of 32 human-readable characters, which is then encoded in Base64 [3]	See [3].
Device Address Type	C.1	AT	Bit	"0": Public Device address "1": Random Device address	See Volume 6, Part B, Section 1.3 in [4].
Device Address	M	AD	Address	The Bluetooth Device Address, represented by 12 uppercase hexadecimal characters	See Volume 3, Part C, Section 15.1 in [4].

Component	Req	ID	Format	Value	Reference
Transport Type for Pairing	M	TR	Number	"0": BR/EDR "1": LE All other values: RFU	See Volume 2, Part H, Section 7.2.2 in [4] for BR/EDR, and Volume 3, Part H, Section 2.3.5.6.4 in [4] for LE.
Secure Connections Random number ^{Note 2}	M	RN	Number	A 128-bit (16 octets) value represented by 32 uppercase hexadecimal numbers	See Volume 3, Part H, Section 2.3.5.6.4 in [4] for LE.
Secure Connections Confirm value	M	CFM	Number	A 128-bit (16 octets) value represented by 32 uppercase hexadecimal numbers	See Volume 3, Part H, Section 2.3.5.6.4 in [4] for LE.
Note 1: The Display Name value may be identical to the Bluetooth Device Name exposed by the Bluetooth device to remote devices (see Volume 3, Part C, Section 3.2.2 in [4]). Note 2: See additional requirements in Section 6.1 .					

Table 5.1: OPU components

C.1: Mandatory if Transport Type for Pairing is LE (TR="1"), otherwise Excluded.

5.3 OPU data format usage

The following common rules shall be applied to the URI data format used with the OPU:

- Each OPU component shall be identified by an identifier listed in [Table 5.1](#) or [Table 5.3](#).
- Each identifier shall be expressed in uppercase ASCII (American Standard Code for Information Interchange) characters.
- Each identifier shall be followed by a colon ":" as a delimiter, followed by the data that is relevant for that element (which may be empty).
- A semicolon ";" shall be used to terminate each `identifier:data` pair.
- A semicolon ";" that is not attached to any `identifier:data` pair field shall indicate the end of an OPU, and any additional subsequent characters shall be ignored.
- All String value fields of components shall be encoded in Base64 [3] and shall not use colons or semicolons. Leading zeros and padding may be omitted.
- Space characters that appear outside of Base64-encoded string components shall be ignored.

- If an identifier other than the vendor-specific identifier is present more than once, then the implementation shall abort its parsing operation.
- Vendor-specific data, if it exists, shall be included in the vendor-specific identifier defined in [Table 5.3](#).

5.4 Formatting the OPU components

The rules specified in [Table 5.2](#) shall be used to format OPU components.

Identifier Type	Description	Example
Bit	The value, “0” or “1”, used to identify single bits.	AT:1
Number	All numbers shall be represented as hexadecimal values using uppercase letters and numbers.	TR:1
Address	Device Addresses shall be represented by 12 uppercase hexadecimal characters. Leading zeroes shall be retained.	AD:001A2B3C4D5E
String	The resulting ASCII string from Base64 encoding of the UTF-8 encoded strings or of binary data.	A display name of “Awesome Headphones” will result in the following: DN:QXdlc29tZSBIZW FkcGhvbmVz

Table 5.2: Encoding formats for OPU elements

5.5 Capturing and parsing the OPU elements

When a user instructs their device to read an OPU, the OPU Decoder receives the complete text string and should parse it in the following order:

1. Detect the existence of a Bluetooth URI with the prefix “BLUETOOTH:”. If no matching prefix is found, then the implementation shall abort its parsing operation; otherwise, it shall continue to process the string from the first character after the colon “:” delimiter.
2. Tokenize the URI using the semicolon “;” delimiter to create a list of `identifier:data` pairs. If tokenization results in an error, then the implementation shall abort its parsing operation.
3. Detect the Bluetooth Core Specification component “CORE:” and the value “OPU”. If the Bluetooth Core Specification component is not found, or if the value does not match the value “OPU”, then the implementation shall abort its parsing operation.
4. For each tokenized component:
 - a. Use the colon “:” delimiter to separate the identifier from the data.
 - b. If the identifier is not defined in [Table 5.1](#) and [Table 5.3](#) (unknown identifier), then the identifier shall be ignored, and parsing shall proceed to the next tokenized component.

- c. If the parsed value length is truncated due to omission of leading zeros or padding, then the parsed value shall be restored back to its original size.
 - d. If the identifier is the vendor-specific identifier, then it can be ignored, or be Base64 decoded to identify the Company Identifier or the Organization Identifier (see Section 5.7.1) and then processed in an implementer-specific way.
 - e. If the identifier is defined in Table 5.1 or Table 5.3, then decode the value accordingly (either Base64 string decode, number, or address). The decoded string should be padded to the required length specified in the Bluetooth Core Specification [4], along with the restoration of leading zeroes.
 - f. If an `identifier:data` pair with a valid identifier contains malformed or insufficient data, then the implementation shall abort its parsing operation.
5. If the parsed OPU does not contain all mandatory values, then the implementation shall terminate the operation and should notify the user about the error.
 6. If the parsing is aborted because of an error, then the implementation should notify the user about the error.

5.6 Generic Access Profile requirements

If the Central does not support the transport type indicated by the Transport Type for Pairing component, then the Central shall abort the pairing operation, and the UI should inform the user that pairing failed because the devices are incompatible.

5.6.1 Requirements for OPU on the LE transport

When a device in the OPU Encoder role, acting in the GAP LE Peripheral role, has generated an OPU for the purpose of pairing, the device shall be configured in the Undirected Connectable mode (see Volume 3, Part C, Section 9.3.4 in [4]) and shall perform connectable undirected advertising (see Volume 6, Part B, Section 6.2.4 in [4]).

After the device in the OPU Decoder role, acting in the GAP LE Central role has decoded the information from the OPU, the device shall proceed to connection establishment and pairing based on the value of the Transport Type for Pairing component.

If the Transport Type for Pairing component indicates that the transport type for pairing is LE, then the device in the OPU Decoder role, acting in the GAP LE Central role shall proceed as follows:

1. The Central may use the AdvA field and the TxAdd bit (see Volume 6, Part B, Section 2.3 in [4]) of connectable undirected advertising packets or scan response packets to match an advertiser entry with the Device Address and the Device Address Type components the Central acquired from the OPU.
2. If no entry matches, then the Central may continue scanning for up to a vendor-defined period, while continuing to match new scan results. This period should be set to no more than 15 seconds.

3. If no matches are found after the vendor-defined period, then the Central shall abort the operation, and the UI should inform the user that the pairing failed because the device was not found.
4. The Central shall establish a connection to the Peripheral that matches the entry in the advertising data or the scan response data (see Volume 6, Part B, Section 4.4.4 in [4]). If a connection cannot be established, then the Central shall abort the operation, and the UI should inform the user that connection to the requested device failed because a connection could not be established.
5. The Central shall initiate the LE Secure Connections OOB Pairing procedure (see Volume 3, Part H, Section 2.3.5 in [4]).
6. If the pairing procedure results in a successful pairing, then the UI should inform the user about the successful pairing. Otherwise, the UI should inform the user about the error and possible ways to resolve the issue (e.g., try again).

5.6.2 Requirements for OPU on the BR/EDR transport

When a device in the OPU Encoder role, acting in the GAP paged device role (see Volume 3, Part C, Section 2.2.1 in [4]), has generated an OPU for the purpose of pairing, the device shall be configured in the GAP BR/EDR Connectable mode, and shall enter the Page Scan substate (see Volume 2, Part B, Section 8.3.2 in [4]).

After the device in the OPU Decoder role, acting in the GAP paging device role, has decoded the information from the OPU, the device shall proceed to connection establishment and pairing based on the value of the Transport Type for Pairing component.

If the Transport Type for Pairing component indicates that the transport type for pairing is BR/EDR, then the device in the OPU Decoder role, acting in the GAP paging device role shall proceed as follows:

1. The device may use the Extended Inquiry Response (EIR) data for discoverable devices to get more information about the peer device.
2. The device shall initiate the paging procedure (see Volume 2, Part B, Section 8.3 in [4]) using the address value of the Device Address component.
3. If paging fails or times out, then the device shall abort the operation, and the UI should inform the user that the pairing failed because the device was not found.
4. The device shall initiate the OOB Pairing Procedure (see Volume 2, Part H, Section 7.2.2 in [4]).
5. If the pairing procedure results in a successful pairing, then the UI should inform the user about the successful pairing. Otherwise, the UI should inform the user about the error and possible ways to resolve the issue (e.g., try again).

5.7 Extended OPU components

An OPU may contain additional data elements to provide additional information to the Central. These data elements are listed in [Table 5.3](#).

Component	Req	ID	Format	Value	Comment
Uniform Resource Locator (URL)	O	URL	String	A UTF-8 formatted URL link used to provide more information about the device, or additional related tools (e.g., a companion app) encoded in Base64 [3]	Companion apps allow users to interact with their Bluetooth devices in ways that are not possible with the hardware alone (e.g., exposing additional capabilities or settings). The URL can point to an application store website, or to an intermediary web server. HTTPS should be used.
Manufacturer Name	O	MNF	String	A UTF-8 string that represents the name of the manufacturer of the device, containing a minimum of 2 and a maximum of 32 human-readable characters, which is then encoded in Base64 [3]	See [3].
Model Number	O	MDL	String	A UTF-8 string that represents the model number that is assigned by the device vendor, containing a minimum of 1 and a maximum of 16 human-readable characters, which is then encoded in Base64 [3]	See [3].
Vendor ID Source	C.3	VIS	Number	An 8-bit (1 octet) value that designates which organization assigned the value used in the Vendor ID component, represented by 2 uppercase hexadecimal characters	The Vendor ID Source values can be specified by a higher-layer specification.
Vendor ID	O	VID	Number	A 16-bit (2 octets) value that uniquely identifies the vendor of the device, represented by 4 uppercase hexadecimal characters	The Vendor ID values can be specified by a higher-layer specification.
Product ID	O	PID	Number	A 16-bit (2 octets) value that distinguishes between different products made by the vendor identified in the Vendor ID component, represented by 4 uppercase hexadecimal characters	The vendors manage Product ID field values.
LE Advertising Data	C.1	LAD	String	A collection of Length-Type-Value (LTV) encoded fields, structured as the Advertising Data (AD) payload used for device discovery and	See Volume 3, Part C, Section 11 in [2], and [3].

Component	Req	ID	Format	Value	Comment
				capability announcement in Bluetooth LE, encoded in Base64 [3]	
BR/EDR Extended Inquiry Response	C.2	EIR	String	A collection of LTV encoded fields, structured as the Extended Inquiry Response (EIR) payload used for device discovery and capability announcement in BR/EDR, encoded in Base64 [3]	See Volume 3, Part C, Section 8 in [2], and [3].
Vendor Specific	O	VS	String	Any vendor-specific data that may be ignored by a device that scans it for the purpose of Bluetooth OOB pairing, without any functional impact to standardized features	See Section 5.7.1 and Table 7.1 in [2].

Table 5.3: Extended OPU data

- C.1: Optional if Transport Type for Pairing is set to “1” (LE) or if the OPU Encoder is a BR/EDR/LE device; otherwise Excluded.
- C.2: Optional if Transport Type for Pairing is set to “0” (BR/EDR) or if the OPU Encoder is a BR/EDR/LE device; otherwise Excluded.
- C.3: Mandatory if Vendor ID is included in the OPU; otherwise Excluded.

5.7.1 Rules for the vendor-specific component

Vendor-specific data shall be a Base64-encoded string that starts with one of the following components:

- Company Identifier represented in 4 hexadecimal characters (see Table 7.1 in [2]) and formatted as “CID:<Company Identifier>;”
- Organization Identifier represented in 2 hexadecimal characters (see Section 6.7.1 in [2]) and formatted as “OID:<Organization Identifier>;”

The Company Identifier or Organization Identifier is followed by any arbitrary data added by the vendor.

For example, vendor-specific data with Company Identifier 0x3456 and some data that is formatted as “CID:3456;data1;data2;in any format;” results in the encoded Vendor Specific component “VS:Q0lEOjM0NTY7ZGF0YTE7ZGF0YTI7aW4gYW55IGZvcmlhdDs==”. The Vendor Specific component may be present more than once, and each instance shall use a unique Company Identifier or a unique Organization Identifier.

5.8 OPU message structure in augmented Backus–Naur form

This section defines the OPU message structure in the Augmented Backus–Naur Form (ABNF) with case-sensitive string support [8].

```

; =====
; Top-Level OPU Message Structure
; =====
message          = %s"BLUETOOTH:" spec-type content terminator
terminator       = ";"
spec-type        = %s"CORE:OPU;"

content          = mandatory-fields optional-fields repeating-fields
                  < Constraint: Standard identifiers (except VS) may
                  appear in any order but must follow the uniqueness
                  and conditional rules below. >

; =====
; 1. Mandatory Fields
; =====
mandatory-fields = dn-p ad-p tr-p rn-p cfm-p at-p
                  < Constraint: AT is mandatory if TR=1;
                  excluded if TR=0. >

; =====
; 2. Optional Fields
; =====
optional-fields  = [url-p] [mnf-p] [mdl-p] [vis-p] [vid-p] [pid-p] [lad-p] [eir-p]
                  < Constraints:
                  LAD is allowed only if TR=1 or device is a BR/EDR/LE device.
                  EIR is allowed only if TR=0 or device is a BR/EDR/LE device. >

; =====
; 3. Repeating Fields
; =====
repeating-fields = *vs-pair

; =====
; Pair Definitions
; =====
separator        = ";"

; Mandatory IDs
dn-p             = %s"DN"   ":" string-val   separator
ad-p             = %s"AD"   ":" address-val  separator
tr-p             = %s"TR"   ":" number-val   separator
rn-p             = %s"RN"   ":" number32-val separator
cfm-p            = %s"CFM"  ":" number32-val separator
at-p             = %s"AT"   ":" bit-val      separator

; Optional IDs
url-p            = %s"URL"  ":" string-val   separator
mnf-p            = %s"MNF"  ":" string-val   separator
                  < Constraint: Decoded value must be 2-32

```

```

        human-readable characters. >
mdl-p      = %s"MDL" ":" string-val      separator
           < Constraint: Decoded value must be 1-16
             human-readable characters. >
vis-p      = %s"VIS" ":" number2-val     separator
           < Constraint: VIS is mandatory if VID is in use;
             otherwise excluded. >
vid-p      = %s"VID" ":" number4-val     separator
pid-p      = %s"PID" ":" number4-val     separator
lad-p      = %s"LAD" ":" string-val      separator
eir-p      = %s"EIR" ":" string-val      separator

; Repeating ID
vs-pair    = %s"VS"  ":" vs-encoded-string separator

; =====
; Primitive Data Types & Core Rules
; =====
bit-val    = "0" / "1"
address-val = 12HEXDIG
number-val  = HEXDIG
number2-val = 2HEXDIG
number4-val = 4HEXDIG
number32-val = 32HEXDIG

; Strict Base64 (RFC 4648)
string-val  = 1*(4b64-char) [ (2b64-char "==") / (3b64-char "=") ]
b64-char    = ALPHA / DIGIT / "+" / "/"

; Vendor Specific Internal Logic
vs-encoded-string = < a string-val that, when decoded per RFC 4648,
                    matches the vs-raw rule >

vs-raw      = (cid-prefix / oid-prefix) vendor-data
cid-prefix  = %s"CID:" number4-val ";"
oid-prefix  = %s"OID:" number4-val ";"
vendor-data = *VCHAR

; Standard ABNF Core Rules (STD 68)
UPALPHA     = %x41-5A
HEXDIG      = DIGIT / %x41-46

```

6 Security requirements

This section describes the security requirements for devices that generate or use the OPU.

6.1 Random number

The random number r is used for the OOB Pairing procedure stage 1, phase 2, and conveyed by the OPU with the data type RN.

Devices that implement the OPU Encoder role shall generate a new random number r for every pairing attempt using the requirements for random number generation as defined in Volume 2, Part H, Section 2 in [4].

6.2 BR/EDR devices with Secure Connections

BR/EDR devices using OPU for pairing with a remote device shall satisfy all of the security requirements of BR/EDR Security Mode 4 Level 4 (see Volume 3, Part C, Section 5.2.2 in [4]). Secure Simple Pairing (SSP) with P-192 elliptic curve shall not be used with OPU.

A device implementing the OPU Encoder role and a device implementing the OPU Decoder role shall generate a new Elliptic Curve Diffie-Hellman (ECDH) public private key pair (phase 1) for every pairing attempt (see Volume 2, Part H, Section 7.1 in [4]).

When a device that implements the OPU Decoder role decodes an OPU for the purpose of OOB pairing, the device shall set the OOB_Auth_Data flag included in the LMP_IO_CAPABILITY_REQ PDU to OOB Authentication Data received (0x01) (see Volume 2, Part C, Section 4.2.7 in [4]).

When a device that implements the OPU Encoder role encodes an OPU for the purpose of OOB pairing, the device shall start a timer with an implementation-defined timeout, which may be zero. If, before the timer expires, the device receives a remote capability response that does not have the OOB data present flag set, then the device shall abort the pairing request by sending a negative reply for the capability request event indicating the reason code set to *Connection Rejected due to Security Reasons* (0x0E).

6.3 LE devices with Secure Connections

LE devices using OPU for pairing with a peer device shall satisfy all of the security requirements of LE Security Mode 1 Level 4.

A device implementing the OPU Encoder role and a device implementing the OPU Decoder role shall generate a new ECDH public private key pair (phase 1) for every pairing attempt (see Volume 3, Part H, Section 2.3.5.6 in [4]).

When a device that implements the OPU Decoder role decodes an OPU for the purpose of OOB pairing, the device shall set the OOB data flag parameter included in the Security Manager Protocol (SMP) Pairing Request PDU (see Volume 3, Part H, Section 3.5.1 in [4]).

When a device that implements the OPU Encoder role encodes an OPU for the purpose of OOB pairing, the device shall start a timer with an implementation-defined timeout, which may be zero. If, before the timer expires, the device receives an SMP Pairing Request PDU from the Initiator that has the OOB data flag set to OOB Authentication data not present (0x00), then the device shall abort any unfinished pairing requests.

6.4 BR/EDR/LE devices

BR/EDR/LE devices shall generate or use an OPU for either BR/EDR or LE transport, and may use the Cross Transport Key Derivation (CTKD) feature to generate a Long Term Key (LTK) for the other transport.

7 Examples of OPUs as QR codes

This section shows examples of an OPU that is used to generate QR codes.

7.1 Example 1

A smart home and ambiance LE Bluetooth-enabled smart soundbar that lets the user play calming ambient sounds, control illumination, monitor sleep and well-being, and control a smart aromatherapy diffuser implements the OPU Encoder role, and presents the following OPU:

- Display Name: “Living Peacefully”
- Device Address Type [AT]: 1 (Random Device Address)
- Device Address [AD]: 4B:12:34:AB:CD:EF
- Transport Type for Pairing [TR]: 1 (LE)
- LE Secure Connections Random number [RN]: 0xD8F3C10A_E79B2D55_80C4EAA6_1F3B927C
- LE Secure Connections Confirm number [CFM]: 0x6C7E2A91_D84E5F3B_C9A71B52_840F9E56
- LE Advertisement Data [LAD]:
 - LTV Length: 0x03 bytes
 - AD Type: 0x19 “Appearance”
 - Standalone Speaker 0x0841: [41 08]
- Manufacturer Name: “SJC Corp.”
- Model Number: “18”

This results in a 186-character OPU:

```
BLUETOOTH:CORE:OPU;DN:TGl2aW5nIFBlYWw1ZnVsbnHk=;AT:1;AD:4B1234ABCDEF;TR:1;RN:D8F3C10AE79B2D5580C4EAA61F3B927C;CFM:6C7E2A91D84E5F3BC9A71B52840F9E56;LAD:AxlBCA==;MNF:U0pDIENvcnAu;MDL:MTg=;
```

This string is represented by the QR code in [Figure 7.1](#).



Figure 7.1: The resulting QR code for Example 1

7.2 Example 2

A BR/EDR car kit that implements the OPU Encoder role allows the owner of the car to pair their phone easily and seamlessly by displaying the following OPU on the car’s display:

- Display Name: “InfinitiStream Bluetooth”
- Device Address [AD]: AABBC012345
- Transport Type for Pairing [TR]: 0 (BR/EDR)
- Secure Connections Random number [RN]: 0xC9F03A7B_E21D885A_440C6B1F_9E5D7230
- Secure Connections Confirm value [CFM]: 0x7AE40C9B_1F88D2E5_93C0A174_B6D2F5E0
- BR/EDR Extended Inquiry Response [EIR]:
 - LTV Length: 0x05 bytes
 - AD Type: 0x03 “Complete List of 16-bit Service Class UUIDs” and two services
 - Service Class UUID 0x110B (Audio Sink): [0B 11]
 - Service Class UUID 0x111E (Handsfree): [1E 11]
- URL [URL]: <https://infinitistream.bluetooth>

- Vendor ID Source: 0x01
- Vendor ID: 0x1234
- Product ID: 0x5678

This results in a 236-character OPU:

```
BLUETOOTH:CORE:OPU;DN:SW5maW5pdGlTdHJlYW0gQmx1ZXRvb3Ro;AD:AABBCC012345;TR:0;RN:C9F03A7BE21D885A440C6B1F9E5D7230;CFM:7AE40C9B1F88D2E593C0A174B6D2F5E0;EIR:BQMLER4R;URL:aHR0cHM6Ly9pbmZpbml0aXN0cmVhbS5ibHVldG9vdGg=;VIS:1;VID:1234;PID:5678;;
```

This string is represented by the QR code in [Figure 7.2](#).



Figure 7.2: The resulting QR code for Example 2

8 Acronyms and abbreviations

Acronym/Abbreviation	Meaning
ABNF	Augmented Backus–Naur Form
AD	Advertising Data
ASCII	American Standard Code for Information Interchange
BR/EDR	Basic Rate/Enhanced Data Rate
CTKD	Cross Transport Key Derivation
ECDH	Elliptic Curve Diffie-Hellman
EIR	Extended Inquiry Response
LE	Low Energy (as in Bluetooth Low Energy)
LTK	Long Term Key
LTV	Length Type Value
MITM	Man-in-the-Middle
OOB	out-of-band
OPU	OOB Pairing URI
PDU	Protocol Data Unit
QR code	Quick Response code
RFU	Reserved for Future Use
SMP	Security Manager Protocol
SSP	Secure Simple Pairing
UI	User Interface
UTF	Unicode Transformation Format
UUID	Universally Unique Identifier

Acronym/Abbreviation	Meaning
URI	Uniform Resource Identifier
URL	Uniform Resource Locator

Table 8.1: Acronyms and abbreviations

9 References

- [1] Berners-Lee, T., Fielding, R., and Masinter, L., “RFC 3986: Uniform Resource Identifier (URI): Generic Syntax”, January 2005, <https://datatracker.ietf.org/doc/html/rfc3986>
- [2] Bluetooth Assigned Numbers, <https://www.bluetooth.com/specifications/assigned-numbers/>
- [3] Josefsson, Simon, “RFC 4648: The Base16, Base32, and Base64 Data Encodings”, October 2006, <https://datatracker.ietf.org/doc/html/rfc4648>
- [4] Bluetooth Core Specification (amended), Version 5.0 or later
- [5] International Organization for Standardization (ISO), “ISO/IEC 18004:2024, Information technology–Automatic identification and data capture techniques–QR code bar code symbology specification”, August 2024, <https://www.iso.org/standard/83389.html>
- [6] Bluetooth Core Specification Supplement, Version 13 or later
- [7] GATT Specification Supplement, <https://www.bluetooth.com/specifications/gss>
- [8] Kyzivat, P., RFC 7405: “Case-Sensitive String Support in ABNF”, December 2014, <https://www.rfc-editor.org/rfc/rfc7405.htmlhttps://datatracker.ietf.org/doc/html/rfc7405>