IP Link Protocol

Bluetooth® Protocol Specification

- **Version:** v1.0
- **Version Date:** 2026-11-03
- **Prepared By:** Internet Working Group

Abstract:

This protocol specifies how Internet Protocol (IP) Version 6 (IPv6) packets are transported over Bluetooth® Low Energy (LE).

Version History

Version Number	Date (yyyy-mm-dd)	Comments
v1.0	2026-11-03	Adopted by the Bluetooth SIG Board of Directors.

Acknowledgments

Name	Company
Martin Turon	Google LLC
Jan Slupski	Telink Semiconductor (Shanghai) Co., Ltd
Marcel Holtmann	Intel Corporation
Himanshu Bhalla	Intel Corporation
Inga Stotland	Intel Corporation
Szymon Slupik	Silvair, Inc.
Piotr Winiarczyk	Silvair, Inc.
Krishna Shingala	Nordic Semiconductor ASA
Chris Hansen	Nordic Semiconductor ASA
Kim Schulz	Samsung Electronics Co., Ltd.
LC Ko	MediaTek, Inc.
PG Di Marco	Ericsson AB
Bjarne Klemmensen	Oticon A/S
Dharam Kumar	Cypress Semiconductor Corporation
Victor Zhodzishsky	Infineon Technologies AG
Nancy Lee	Signify Netherlands, B.V.

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 © 2021–2026. 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
1.4 Bluetooth specification release compatibility 8
1.5 Encoding conventions 8
1.6 Terminology 9
2 IP Link architecture 10
2.1 Layered architecture 10
2.1.1 Physical layer 11
2.1.2 Data link layer 12
2.1.3 Network layer 13
2.2 Overview of IP Link operation 13
2.3 Architectural concepts 14
2.3.1 Addressing 14
2.3.2 Tag resolution 14
2.3.3 Security architecture 14
3 IP Link description 16
3.1 IP Link PDU 16
3.1.1 IP Link PDU format 16
3.1.2 IP Link control messages 20
3.1.3 Data link layer behavior 22
3.2 Security 25
3.2.1 Security toolbox 25
3.2.2 Key derivation 26
3.2.3 Network Key hierarchy 27
3.2.4 Replay protection 28
3.2.5 Encryption and authentication 30
3.2.6 Privacy 30
3.3 MAC sublayer 32
3.3.1 AE bearer 32
3.3.2 L2CAP bearer 34
3.3.3 Node Types 34
3.3.4 Bearer selection 35
3.3.5 Connection termination 40
3.3.6 Connection establishment timing considerations 40
4 Parameters and constants 41
4.1 Parameters 41

4.2 Constants 41
5 Sample data 42
5.1 Security toolbox sample data 42
5.1.1 s1 salt generation function 42
5.1.2 k2 function 42
5.2 IP Link PDU message sample data 42
5.2.1 Message #1 42
5.2.2 Message #2 43
5.2.3 Source and destination IRT and ARK 44
6 Acronyms and abbreviations 45
7 References 47
Appendix A Implementation suggestions 48
A.1 System architecture 48
A.2 6LoWPAN adaptation 49
A.2.1 Payload length adaptation 49
A.2.2 Address adaptation 49
A.2.3 Prefix compression 49
A.3 IPLinkPayload 49
A.3.1 Protocol signaling 50
A.4 Link configuration 50
A.4.1 Address configuration 50
A.4.2 RootKey management and lookup 50
A.4.3 Neighbor table 51
A.5 HCI implementation considerations 52
A.5.1 HCI sequences 52
A.6 Example scenarios 54
A.6.1 Commissioning 54
A.6.2 Operational Mesh 54

1 Introduction

The IP Link Protocol specifies an interoperable data link layer for transporting IPv6 packets over Bluetooth® LE wireless technology.

Note: IP Link is designed to be able to interface with IPv6. To learn more about IPv6, see the Internet Engineering Task Force (IETF) document, RFC 8200: Internet Protocol, Version 6 (IPv6) Specification [7], by S. Deering and R. Hinden.

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: 1. what is recommended by the specification without forbidding anything ("should") 2. 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: 1. an indisputable statement of fact that is always true regardless of the circumstances 2. 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.

Certain terms used in this specification have been updated and are no longer used by Bluetooth SIG. For a list of terms that have been updated and their replacement terms, see the Appropriate Language Mapping Tables [10].

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).

To use an additional form of notation in this specification’s tables—such as an ellipsis (…) to indicate a gap in a series of values—add an explanation of that notation in the specification’s section on table conventions for requirements and unused cells. If a particular table uses a notation that differs from the initially expressed table conventions for unused cells, explain this additional notation in the introduction of the table to which it applies.

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.

1.4 Bluetooth specification release compatibility

This specification shall be used with the Bluetooth Core Specification Version 6.0 or later [1], with the following features enabled:

- LE L2CAP Connection-Oriented Channel Support
- LE Data Packet Length Extension
- LE Advertising Extensions
- Decision-Based Advertising Filtering (DBAF)

1.5 Encoding conventions

The following encoding conventions apply to this specification.

1.5.1.1 Endianness and field ordering

All multiple-octet numeric values defined in this specification shall be encoded and sent in big endian order (also known as network byte order). Whenever bits within an octet are listed in a table, they are listed with the higher order or most significant bit(s) first and the next lower order bit(s) each row thereafter.

1.6 Terminology

The terms listed below are defined as follows.

Term	Definition
Address or IP Link Address	The identity of a node as defined by the IP Link.
Network	A group of nodes that share common security credentials and can communicate with other nodes in the network over an IP Link transport.
Node	A Bluetooth LE device that has been provisioned with security credentials and IP Link addresses.

2 IP Link architecture

2.1 Layered architecture

The IP Link Protocol specification is based upon the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) 7498-1 standard Open Systems Interconnection (OSI) layering architecture and the Institute of Electrical and Electronics Engineers (IEEE) 802.3-2018 standard as shown in Table 2.1.

(For more information about these two standards, see the notes following Table 2.1.)

Number	ISO/IEC 7498-1 OSI model	IEEE 802.3-2018 standard for Ethernet	IP Link	Description
7	Application layer	–	–	–
6	Presentation layer			
5	Session layer			
4	Transport layer			
3	Network layer	–	IPv6 and network management	Defined by the Internet Engineering Task Force (IETF). (For more information about IPv6, see the notes following this table.)
		–	IPv6 over Low-Power and Wireless Personal Area Networks (6LoWPAN) compression	Defines how an IPv6 packet can be compressed to an IP Link PDU and sent/received over Bluetooth® LE. 6LoWPAN is defined by the IETF. (For more information about 6LoWPAN, see the notes following this table.)
2	Data link layer	Logical Link Control (LLC) sublayer	6LoWPAN adaptation	Defined by the IETF, with implementation suggestions for the IP Link provided in Section A.2. (For more information about 6LoWPAN, see the notes following this table.)

Number	ISO/IEC 7498-1 OSI model	IEEE 802.3-2018 standard for Ethernet	IP Link		Description
			IP Link PDU encoding/decoding		Defined in the Bluetooth IP Link Protocol.
			MAC layer multiplexing		Defined in the Bluetooth IP Link Protocol.
		Media Access Control (MAC) sublayer	Advertising Extensions (AE) bearer (see Section 3.3.1)	L2CAP bearer (see Section 3.3.2)	Defined in the Bluetooth IP Link Protocol.
			ADVB-U	LE-U	Defined in the Bluetooth Core Specification [1].
1	Physical layer	–	Bluetooth LE PHY		Defined in the Bluetooth Core Specification [1].

Table 2.1: IP Link Protocol stack and comparison with ISO/IEC OSI model and IEEE 802.3-2018 standard

Notes

- IP Link resides on Layer 2 of the ISO/OSI model. For full ISO/OSI layer descriptions and their responsibilities, see ISO/IEC 7498-1:1994: Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model – Part 1 [8] from the International Organization for Standardization (ISO).
- For a full description of Layer 2 responsibilities and typical frame examples, see 802.3-2018 - IEEE Standard for Ethernet [9] from the IEEE.
- IP Link is designed to be able to interface with IPv6. IPv6 is further described in RFC 8200: Internet Protocol, Version 6 (IPv6) Specification [7] by Deering, S. and Hinden, R.
- As IPv6 header size is significant, the interfacing is done with usage of the IPv6 header compression described in RFC 6282: Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks [2] by Hui, J. and Thubert, P.

2.1.1 Physical layer

The physical layer as defined by the ISO/IEC OSI model is the underlying bearer for data link layer packets upon which raw bits are transmitted between neighboring network entities or nodes. The IP Link uses LE advertising physical channels or LE piconet physical channels as described in Volume 1, Part A, Section 3.2, Figure 3.3 of the Core Specification [1] and as shown in Figure 2.1.

Note: For full ISO/OSI layer descriptions and their responsibilities, see ISO/IEC 7498-1:1994: Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model – Part 1 [8] from the International Organization for Standardization (ISO).

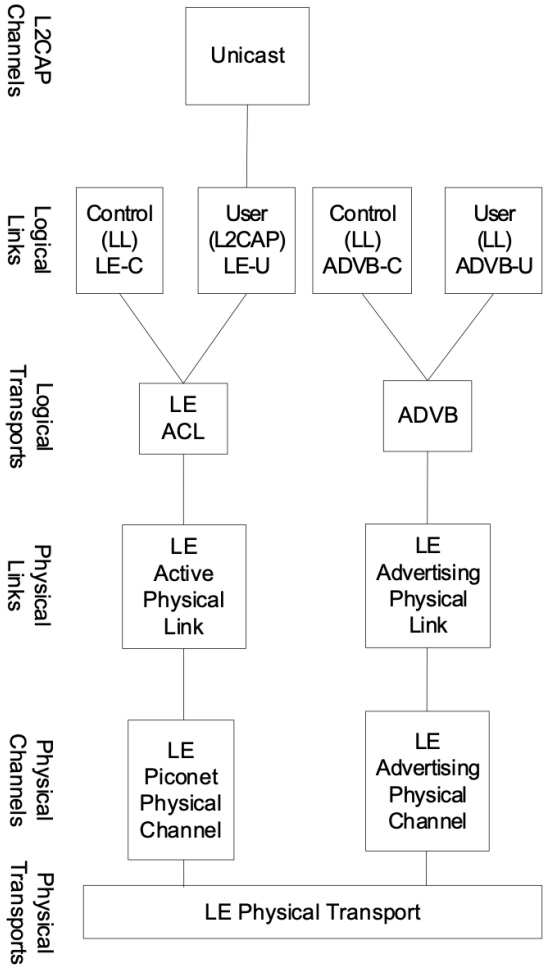


Figure 2.1: Core physical layer transport architecture

2.1.2 Data link layer

The ISO/IEC OSI data link layer defines the encoding protocol and procedural means to transport and validate data messages between adjacent network nodes. The data link layer provides the link layer reliability, security, authentication, encryption, and privacy features of the IP Link.

The IP Link resides at the data link layer and consists of the LLC and MAC sublayers.

> Note: For full ISO/OSI layer descriptions and their responsibilities, see ISO/IEC 7498-1:1994: Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model – Part 1 [8] from the International Organization for Standardization (ISO).

2.1.2.1 LLC sublayer

From the perspective of the IP Link, the LLC sublayer defines the 6LoWPAN adaptation, IP Link PDU encoding/decoding, and multiplexing of the IP Link PDU on to the available MAC sublayers (see Section 2.1.2.2).

2.1.2.2 MAC sublayer

From the perspective of the IP Link, the MAC sublayer defines transports for communicating IP Link PDUs between nodes. The two supported transports include a connectionless AE bearer and a connection-oriented L2CAP bearer.

The AE bearer uses connectionless extended advertising PDUs over the advertising broadcast user data (ADVB-U) defined in the LE advertising broadcast (see Volume 1, Part A, Section 3.5.4.7 in [1]) using the advertising procedure described in Volume 1, Part A, Section 4.2.2.2 in [1] and the decision-based scanning procedure described in Volume 1, Part A, Section 4.2.2.10 in [1].

The L2CAP bearer uses L2CAP PDU connection-oriented channels over the LE-U (see Volume 1, Part A, Section 3.6 in [1]).

See Section 3.3 for a detailed definition of the IP Link MAC sublayer.

2.1.3 Network layer

From the perspective of the IP Link, the network layer is IPv6.

2.2 Overview of IP Link operation

The IP Link operation described by this specification:

- Enables IPv6 packets to be sent from a node to one or more other nodes, i.e., unicast or multicast
- Provides standard encryption and obfuscation mechanisms for security and privacy
- Delivers messages to neighboring nodes with low latency
- Provides built-in forward compatibility

To increase efficiency, the IP Link can transport IPv6 packets compressed and encoded according to IETF RFC 6282. 6LoWPAN fragmentation and reassembly protocol (Section 5.3 in [4]) can be used to transport larger payloads over the IP Link.

> Note: For more information about IPv6 header compression according to IETF RFC 6282, see RFC 6282: Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks [2] by Hui, J. and Thubert, P.

Nodes can enable link layer security on any traffic using a set of keys provided by the upper layer protocol, where either a Network Key or a Temporary Peer Key may be used as defined in Section 2.3.2.

IP Link traffic may be either acknowledged or unacknowledged. An unacknowledged transmission provides no indication of whether the transmission attempt was successful. An acknowledged transmission is sent with a bit set to indicate that a link layer acknowledgment is expected. The transmitter listens for a designated time for such a link layer acknowledgment, receipt of which indicates that the IP Link PDU was received by the destination.

2.3 Architectural concepts

2.3.1 Addressing

The IP Link layer defines two types of addresses: short and long.

The addressing scheme specified by the upper layer assigns short and long addresses to the node.

2.3.1.1 Short address

A short address is 16 bits in length. A short address is used for either unicast or broadcast.

Short addresses are encoded as defined in Table 2.2.

Values	Address Type
0x0000 to 0xFFFE	Unicast address
0xFFFF	Broadcast address

Table 2.2: IP Link short address allocations

2.3.1.1.1 Broadcast address

The broadcast address is a reserved short address that is known by all nodes.

The broadcast address is used only in the Destination Address (DST) field of a message and not in the Source Address (SRC) field. A message sent to the broadcast address is processed by all nodes that receive it (per Section 3.3.1.4.1). An example use case for data link layer broadcast is to support multicast by the upper layer.

2.3.1.2 Long address

A long address is 64 bits in length. A long address is always a unicast address and is uniquely assigned to exactly one node within a network.

2.3.2 Tag resolution

The IP Link uses IP Link resolvable tags (IRTs) to send PDUs or to establish a connection to another node in the network. IRTs are used to help recipient nodes filter incoming advertising PDUs that are not sent by nodes that are not part of the network.

An IRT uses the DBAF Resolvable Tag format (see Volume 6, Part B, Section 2.3.1.11 in [1]). The IRT identifies the short or long IP Link addresses used by a node within a network (see Section 3.3.1.3).

2.3.3 Security architecture

The IP Link provides a general-purpose link layer security mechanism for securing communication between neighboring nodes. The core security material defined for this purpose is the RootKey.

An IP Link Network is configured with one or more RootKeys and an associated AddressKey.

A RootKey is one of two types:

- Temporary Peer Key
- Network Key

The RootKey is used to derive two types of keys: an EncryptionKey and a PrivacyKey, as specified in Section 3.2.2.

The IP Link payload is authenticated and encrypted using the EncryptionKey, as specified in Section 3.2.5.

PrivacyKey is used for privacy and obfuscation, as described in Section 3.2.6.

The upper layer above the IP Link configures the Root Keys. Distribution of key material between sets of nodes is not provided by the IP Link itself but may be performed either out-of-band or via a separate provisioning protocol specified by the upper layer. See Appendix A for information on implementation considerations between the IP Link and the upper layers.

The AddressKey is the core security material used to derive Address Resolution Keys (ARKs). To identify nodes in a private manner, the ARK is used to generate and resolve IRTs. The AddressKey can be derived from distributed security material or generated randomly and distributed by the upper layer above the IP Link.

2.3.3.1 Temporary Peer Keys

A Temporary Peer Key (TempPeerKey) is a pair-wise key that is used to authenticate and encrypt communications between two peer nodes to establish a private link. Two nodes can establish a private link if they are configured with a shared TempPeerKey, the corresponding Key Identifier (KEYID32), the long address of the associated peer, and the last known sequence number (SEQ) (see Section 3.2.4.1) of the peer node for the key.

2.3.3.2 Network Keys

A Network Key (NetKey) is a symmetric key that is shared between a set of nodes that together form a network. A node can have knowledge of more than one NetKey. Nodes on a network link are configured with the shared NetKey and corresponding Key Identifier (KEYID32), the long address of any associated peers, and the last known SEQ (see Section 3.2.4.1) of each peer node for the key.

2.3.3.3 Encryption and authentication

The data link layer payload is encrypted and authenticated using the AES-128 CCM algorithm (see Section 3.2.1.3).

2.3.3.4 Privacy

In addition to encrypting and authenticating the message payload, information in the header is obfuscated for privacy reasons by using the PrivacyKey. For example, the SRC and DST are obfuscated as a measure against traffic analysis attacks.

3 IP Link description

This section provides details for each of the layers outlined in the layered architecture (Section 2.1) and for implementation of security. Figure 3.1 provides a view of the sublayers of an IP Link stack.

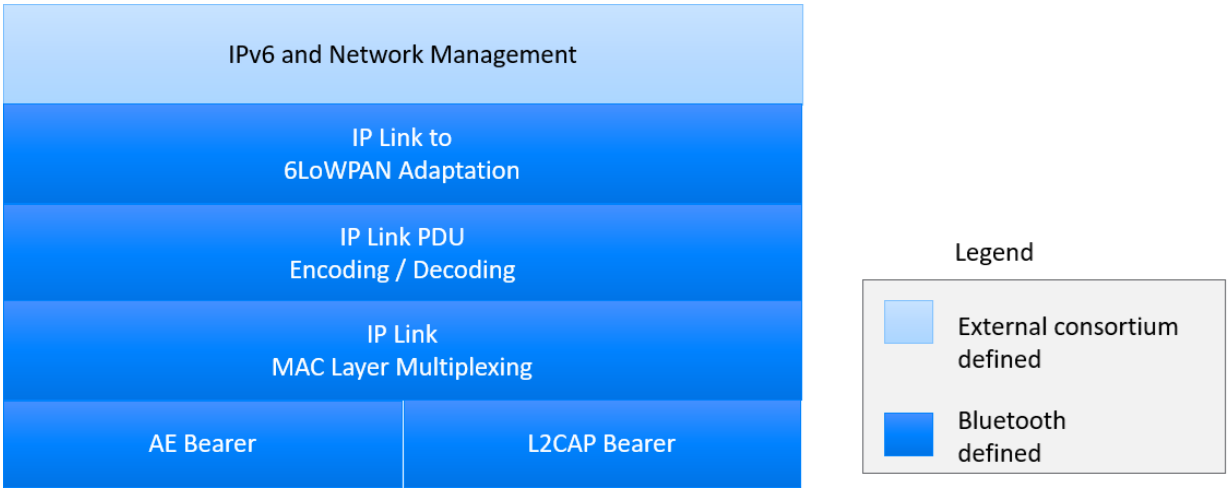


Figure 3.1: IP Link stack

3.1 IP Link PDU

This section describes the encoding and decoding of the IP Link PDU format.

The maximum size of a fully encoded IP Link PDU shall be the IPL_MAX_PDU bytes (see Section 4.2). The maximum PDU size shall include all encoded headers, the payload, and the Message Integrity Check (MIC). The maximum size is agnostic to the underlying bearer and fits within a single AE bearer event (see Section 3.3.1) or an L2CAP Service Data Unit (SDU).

3.1.1 IP Link PDU format

The IP Link PDU format is defined in Table 3.1.

Field names in [brackets] are optional and only present when activated by bits in the Frame Control (FCTL) field (see Section 3.1.1.1).

Field Name	Bits	Description
FCTL	16	Frame Control field
NID	16	Network Identifier
[KEYID8]	0 or 8	Key Identifier (KEYID8) field
[FID]	0 or 8	Frame Identifier (FID) field
SEQ	32	Sequence number

Field Name	Bits	Description
SRC	16 or 64	Source address
DST	16 or 64	Destination address
IPLinkPayload	variable	An IP Link payload
MIC	64	Message Integrity Check for the network

Table 3.1: IP Link PDU field definitions

3.1.1.1 FCTL

The FCTL field indicates the encoding of the frame headers, including what addressing and link control modes are being used.

Field Name	Bit(s)	Description
VER	2	The Version (VER) field identifies the protocol version. VER shall equal 0. Other values are reserved for future use.
OP	3	The Opcode (OP) field determines the type of the IP Link PDU: 0b000 = Data PDU 0b001 = Acknowledgment PDU (ACK) and 8-bit FID field is present in the IP Link PDU 0b010 = Bearer Acknowledgment PDU (BACK) 0b011 – 0b111 = RFU If OP=0b001 or OP=0b010 then the Acknowledgment Requested (AR) shall be set to 0.
AR	1	The AR bit is defined as follows: 0 = No acknowledgment requested 1 = Acknowledgment is requested. The 8-bit FID field is present in the IP Link PDU Setting the AR field bit to 1 indicates to the destination node that an acknowledgment is requested per Section 3.1.3.2. When DST is the broadcast address, AR shall be set to 0.

Field Name	Bit(s)	Description
MD	1	The More Data (MD) bit is defined as follows: 0 = No more data 1 = More data pending Setting the MD field bit to 1 indicates that at least one more IP Link PDU is pending (queued) for the same destination node. When DST is the broadcast address, MD shall be set to 0.
KIP	1	The Key Identifier Present (KIP) bit is defined as follows: 0 = KEYID8 field is not present in the IP Link PDU 1 = 8-bit KEYID8 field is present in the IP Link PDU
SAZ	1	The Source Address Size (SAZ) bit is defined as follows: 0 = 16-bit SRC short address is present in the IP Link PDU 1 = 64-bit SRC long address is present in the IP Link PDU The SRC field is defined in Section 3.1.1.6.
DAZ	1	The Destination Address Size (DAZ) bit is defined as follows: 0 = 16-bit DST short address is present in the IP Link PDU 1 = 64-bit DST long address is present in the IP Link PDU The DST field is defined in Section 3.1.1.7.
BPREF	2	Bearer Selection Preference (BPREF) is defined as follows: 0b00 = Continue on the bearer used to deliver this PDU 0b01 = Connection establishment as Peripheral preferred 0b10 = Connection establishment as Central preferred 0b11 = Connection establishment in either role preferred When OP=0b001 (ACK) or OP=0b010 (BACK), BPREF shall not be set to 0b11. When the PDU is sent over L2CAP bearer, BPREF shall be set to 0b00 on transmission and shall be ignored on reception. See Section 3.3.4.
RFU	4	Reserved for future use. These bits shall be set to 0 and ignored by receivers.

Table 3.2: FCTL field format

3.1.1.2 NID

The NID field is a 16-bit field that contains a unique identifier for the network. It is provided by the upper layer for the purpose of packet filtering and quickly determining whether the packet is to be processed by the IP Link layer. The NID can be set to a random number or generated through some other method as long as it satisfies the requirements for KEYID32 and KEYID8 as described in Section 3.1.1.3.

3.1.1.3 KEYID8

The KEYID8 field is optional per the KIP flag of the FCTL field and allows the sender to identify which key is being used to encrypt and obfuscate the message . The full key identifier is an internal 32-bit field denoted as the KEYID32 (see Section 2.3.3). When included in the PDU header, the KEYID32 is truncated to the lower 8 bits as KEYID8.

The upper layer provides the KEYID32 at the same time as the RootKey per Section 2.3.3. The upper layer shall assign a unique KEYID32 value to a unique RootKey. In addition, the upper layer shall select KEYID32 values such that any {NID, KEYID8} pair maps to a unique KEYID32 unambiguously. See Section A.5 for implementation suggestions regarding key management by the upper layer.

3.1.1.4 FID

The FID field is used to identify which unicast PDU is being acknowledged by an ACK. The FID field shall be present only when AR=1 or OP=1. To keep the FID field unique, the node shall maintain a free running, incrementing counter for each generated PDU that requires an acknowledgment, i.e., with AR=1. When the FID field reaches its maximum value, the next increment shall overflow back to zero. See Section 3.1.3.4 for details on handling of the FID field.

3.1.1.5 SEQ

The SEQ is a 32-bit number maintained by a node and incremented whenever a node transmits a new PDU (see Section 3.2.4.1) for a given key. The SEQ is a critical security field used for generating the nonce of the message.

The behavior of the SEQ differs from the FID when a PDU is retransmitted. For example, if a given PDU is sent consecutively three times over the AE bearer, the three transmissions will have the same FID but different SEQs. See Section 3.1.3.4 for details on handling of the SEQ field.

3.1.1.6 SRC

The SRC is a 16-bit (SRC16) or 64-bit (SRC64) number. The SRC identifies the node within the network that originates this message as specified in Section 3.2.4. This address shall be a short unicast or long address. The SRC shall not be the broadcast address.

The SRC field shall be set to an address owned by the originating node as per Section 2.3.1.

3.1.1.7 DST

The DST is a 16-bit (DST16) or 64-bit (DST64) number. The DST identifies the node or nodes within the network that the message is intended for. This address shall be either a short unicast or a short broadcast or a long address.

The upper layer sets the DST field for each outbound PDU. When generating a PDU that is a reply to another PDU, the DST of the reply shall be set to the SRC of the original PDU.

3.1.1.8 IPLinkPayload

The IPLinkPayload is a byte string provided by the upper layer.

Details for processing the contents of the IPLinkPayload are not provided in this specification (see Section A.3 for implementation suggestions).

3.1.1.9 MIC

The MIC field authenticates that IPLinkPayload has not been changed (see Section 3.2.1.3). The length of the MIC field is 64 bits.

3.1.2 IP Link control messages

3.1.2.1 Acknowledgment PDU

The Acknowledgment PDU (ACK) is defined as any PDU with OP=0b01.

The link layer ACK follows the standard PDU format.

<table>
<tr><th>Field Name</th><th>Bits</th><th>Notes</th></tr>
<tr><td>FCTL</td><td>16</td><td>FCTL field = 000010XXXXXX0000b
<table>
<tr><th>Field</th><th>VER</th><th>OP</th><th>AR</th><th>MD</th><th>KIP</th><th>SAZ</th><th>DAZ</th><th>BPREF</th><th>RFU</th></tr>
<tr><td>Value</td><td>00</td><td>001</td><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>XX</td><td>0000</td></tr>
</table>
Where bits marked as X can be set to 0 or 1</td></tr>
<tr><td>NID</td><td>16</td><td>Set to match the PDU that is acknowledged</td></tr>
<tr><td>[KEYID]</td><td>0 or 8</td><td>Set to match the PDU that is acknowledged</td></tr>
<tr><td>FID</td><td>8</td><td>Set to match the PDU that is acknowledged</td></tr>
<tr><td>SEQ</td><td>32</td><td>Sequence number</td></tr>
<tr><td>SRC</td><td>16 or 64</td><td>Source address</td></tr>
<tr><td>DST</td><td>16 or 64</td><td>Destination Address</td></tr>
<tr><td>MIC</td><td>64</td><td>Message Integrity Check</td></tr>
</table>

Table 3.3: ACK PDU format

The FID field of the ACK shall be set to the FID field of the PDU being acknowledged.

The NID field of the ACK shall be set to the NID of the PDU being acknowledged.

The SAZ and SRC fields of the ACK shall be set to match the DAZ and DST of the PDU being acknowledged.

The DAZ and DST fields of the ACK shall be set to match the SAZ and SRC of the PDU being acknowledged.

A link layer ACK sent with MD=1 means the sender of the ACK has more data queued for the destination of the ACK. For example, a common use case is a parent node that has cached messages for a low-power child node when acknowledging a message from the child node.

3.1.2.2 Bearer Acknowledgment PDU

The Bearer Acknowledgment PDU (BACK) is defined as any PDU with OP=0b10.

The link layer BACK follows the standard PDU format.

<table>
<tr><th>Field Name</th><th>Bits</th><th>Notes</th></tr>
<tr><td>FCTL</td><td>16</td><td>FCTL field = 0001000XXXXX0000b
<table><tr><th>Field</th><td>VER</td><td>OP</td><td>AR</td><td>MD</td><td>KIP</td><td>SAZ</td><td>DAZ</td><td>BPREF</td><td>RFU</td></tr><tr><th>Value</th><td>00</td><td>010</td><td>0</td><td>0</td><td>X</td><td>X</td><td>X</td><td>XX</td><td>0000</td></tr></table>Where bits marked as X can be set to 0 or 1</td></tr>
<tr><td>NID</td><td>16</td><td>Set to match the PDU that is acknowledged</td></tr>
<tr><td>[KEYID]</td><td>0 or 8</td><td>Set to match the PDU that is acknowledged</td></tr>
<tr><td>SEQ</td><td>32</td><td>Sequence number</td></tr>
<tr><td>SRC</td><td>16 or 64</td><td>Source address</td></tr>
<tr><td>DST</td><td>16 or 64</td><td>Destination Address</td></tr>
<tr><td>MIC</td><td>64</td><td>Message Integrity Check</td></tr>
</table>

Table 3.4: BACK PDU format

The BACK PDU is sent as a reply to an IP Link Data PDU (OP=0b00) with MD=1 and AR=0 to convey the Bearer Preference (BPREF) for subsequent communication between nodes.

The NID field of the BACK shall be set to the NID of the PDU being acknowledged.

The SAZ and SRC fields of the BACK shall be set to match the DAZ and DST of the PDU being acknowledged.

The DAZ and DST fields of the BACK shall be set to match the SAZ and SRC of the PDU being acknowledged.

See Section 3.3.4 for more information about the usage of the BACK PDU.

3.1.3 Data link layer behavior

3.1.3.1 Receiving a data link layer message

This section describes how the IP Link processes incoming PDUs.

Over AE bearer, the IP Link uses the Decision-Based Advertising Filtering (DBAF) feature of the Controller to listen for decision PDUs containing Resolvable Tags addressed to one of the IRTs that identifies the node (see Section 3.3.1.4.1). If the AE bearer accepts the PDU or if the PDU is received over the L2CAP bearer, the following steps shall determine if the PDU is accepted:

- If KEYID8 is present, it is used to determine the KEYID32 (see Section 3.1.1.2). The KEYID32 is used to reference a RootKey (NetKey or TempPeerKey) and query its derived EncryptionKey and PrivacyKey. The received PDU is then de-obfuscated as described in Section 3.2.6.

- If KEYID8 is not present, the RootKey type is TempPeerKey (see Section 2.3.3.1). For each TempPeerKey associated with the NID the PrivacyKey is queried and used to de-obfuscate the PDU as described in Section 3.2.6. If the de-obfuscated SRC address identifies the node associated with the TempPeerKey, then the EncryptionKey is queried and processing continues, otherwise the message shall be ignored.

The EncryptionKey is then used to decrypt and authenticate the PDU (see Section 3.2.3.1). Any collisions because of truncation of a KEYID32 to a particular KEYID8 are resolved by whether the PDU authenticates after decryption and the DST matches any of the addresses of the receiving node on the network identified by the NID.

- If the message does not authenticate against any known EncryptionKey, then the message shall be ignored.
- If the last authenticated SEQ for the SRC and key is unknown, the SEQ for the first authenticated message from the peer SRC on the key may be considered valid to initialize SEQ tracking of that peer using a trust-first policy.
  - Alternatively, the upper layer may initialize SEQ tracking of a peer via an out-of-band synchronization protocol.
- If the SEQ is equal to or less than the SEQ from the last authenticated message from that SRC, then the message shall be ignored.
- Otherwise, the message shall be delivered for processing to the upper layer.

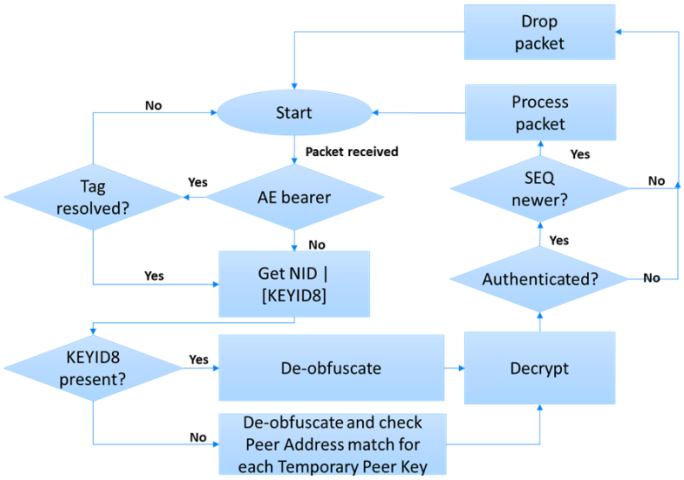


Figure 3.2: IP Link receive path flow diagram

3.1.3.2 Responding to a data link layer message

When an authenticated PDU is received with AR=1, the receiver shall send an ACK in response. The upper layer may request the receiver suppress forwarding of duplicates for a given NetKey when the same FID from an SRC is seen. When duplicate suppression is enabled, the receiver shall track the last forwarded FID received as FID_{last}, and not forward duplicates for FID_{last} until a new FID is received or a timeout of IPL_DUP_TIMEOUT is exceeded. The timeout avoids indefinite suppression of a given FID value in cases of node reset, and is calculated as follows (see Section 4.1 for details on the parameters used in this calculation):

$$IPL_DUP_TIMEOUT = IPL_ACK_TIMEOUT \times (1 + (1 + IPL_BACKOFF_JITTER) \times IPL_RETRY_COUNT^2)$$

3.1.3.3 Transmitting a data link layer message

The upper layer shall provide an IPv6 packet for transmission to the IP Link layer. The upper layer provides all the following PDU fields to the IP Link layer:

- NID – The NID field is set by the upper layer to the identifier corresponding to the network.
- KEYID32 – The RootKey unique identifier used to query the EncryptionKey and PrivacyKey associated with the NID (see Section 3.2.2). These keys shall be used for encryption and obfuscation.
- FCTL – The upper layer shall provide the parameters to construct the FCTL field, including the values of the MD, AR, and preferences for bearer selection to construct the BPREF bits.
- SRC – The SRC field is set by the upper layer to the long or short unicast address of the node sending the IP Link PDU.
- DST – The DST field is set by the upper layer to a long or short unicast address or to the broadcast address to identify the destination node or nodes.
- IPLinkPayload – The raw payload. Section A.2 describes a method to transport IPv6 packets encoded by a 6LoWPAN adaptation layer.

The SEQ field shall be set by the IP Link layer to the SEQ of the node for the given key and SRC. The SEQ shall then be incremented by 1 for each new IP Link PDU.

The MIC field shall be set as defined in Section 3.2.1.3.

After being formed, the PDU shall be encrypted according to Section 3.2.5 and obfuscated according to Section 3.2.6. Then the node shall send the PDU according to the transport selection logic specified in Section 3.3.4.

3.1.3.4 Retransmitting a data link layer message

When transmitting a PDU with AR=1, the node shall set a timeout for receipt of the acknowledgment. The acknowledgment timeout and number of retransmissions are determined by the upper layer. The default number of retransmissions is IPL_RETRY_COUNT. The acknowledgment timeout is calculated as follows (see Section 4.1 for details on the parameters used in this calculation):

$$t = IPL_ACK_TIMEOUT \times (IPL_BACKOFF_BASE)^{n} \times (1.0+\text{random}(0,1)) \times IPL_BACKOFF_JITTER$$

Where:

n = the transmission count of the message, starting with 0
t = the resultant retransmission timeout for this transmission

If an acknowledgment with a matching FID is received before the timeout, the timeout shall be canceled. If no acknowledgment is received before the timeout, the queued PDU is re-encrypted with the same FID and a new SEQ and resent. Figure 3.3 shows an example of a node retransmitting a PDU until receipt of acknowledgment.

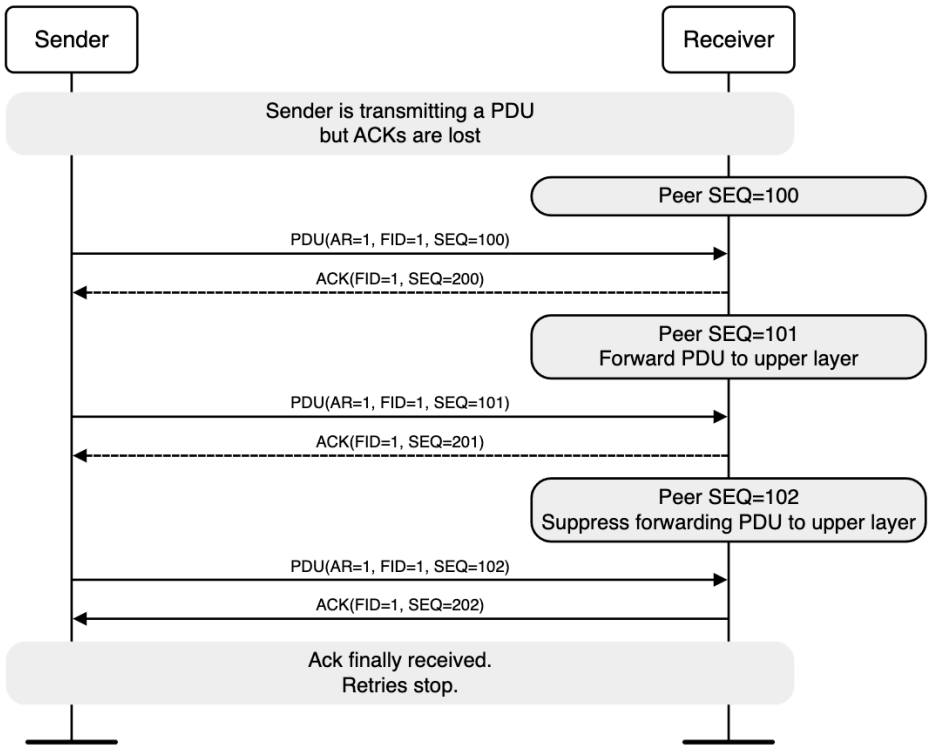


Figure 3.3: IP Link retransmission and acknowledgment

3.2 Security

This section describes how IP Link security is implemented.

3.2.1 Security toolbox

3.2.1.1 Encryption function *e*

The encryption function *e*, as defined in Volume 3, Part H, Section 2.2.1 in [1], shall be used. This can be summarized as:

ciphertext = e(key, plaintext)

3.2.1.2 CMAC function

RFC 4493 [3] specifies the Cipher-based Message Authentication Code (CMAC) that uses AES-128 as the block cipher function, also known as AES-CMAC. The inputs to AES-CMAC are as follows:

k is the 128-bit key

m is the variable length data to be authenticated

The 128-bit message authentication code (MAC) is generated as follows:

MAC = AES-CMAC(k, m)

Note: Generation of the 128-bit message authentication code uses the same notation as in RFC 4493 [3]. This is functionally the same as the notation used in other Bluetooth specifications, where MAC = AES-CMAC$_k$(m).

A node can implement AES functions in the Bluetooth® Host or use the HCI_LE_Encrypt command (see Volume 2, Part E, Section 7.8.22 in [1]) to use the AES function in the Controller.

3.2.1.3 CCM function

NIST SP 800-38C [5] specifies the AES Counter with CBC-MAC (CCM) (see Volume 6, Part E, Section 1 in [1]). This specification defines AES-CCM as a function that takes four inputs and results in two outputs:

ciphertext, mic = AES-CCM (k, n, m, a)

The inputs to AES-CCM are as follows:

k is the 128-bit key

n is a 104-bit nonce

m is the variable length data to be encrypted and authenticated (also known as “plaintext”)

a is the variable length data to be authenticated (also known as “associated data” / AAD)

The outputs to AES-CCM are as follows:

ciphertext is the variable length data after it has been encrypted

mic is the MIC value of m and a (also known as the “message authentication code”)

3.2.1.4 Salt function (s1)

The input to function s1 is as follows:

M is a non-zero length octet array or an ASCII encoded string

If M is an ASCII encoded string, M shall be converted into an octet array by replacing each string character with its ASCII code while preserving the order. For example, if M is the message string “SALT”, then M gets converted into the following octet array: 0x53, 0x41, 0x4c, and 0x54.

ZERO is the 128-bit value:

0x0000 0000 0000 0000 0000 0000 0000 0000

The output of the salt generation function s1 is as follows:

s1(M) = AES-CMAC$_{ZERO}$ (M)

3.2.2 Key derivation

For any given link key, such as a NetKey or TempPeerKey, other keys are derived as follows.

3.2.2.1 Key derivation function (k2)

The NetKey material derivation function k2 is used to generate instances of the EncryptionKey and PrivacyKey to use for encryption and obfuscation in node communication.

The definition of this key derivation function makes use of the MAC function AES-$CMAC_T$ with a 128-bit key T. The inputs to function k2 are as follows:

N is 128 bits

P is 1 or more octets

The key (T) is computed as follows:

$T = AES\text{-}CMAC_{SALT}(N)$

SALT is the 128-bit value computed as follows:

SALT = s1("ipl2")

The output of the key derivation function k2 is as follows[1]:

T0 = empty string (zero length)
$T1 = AES\text{-}CMAC_T(T0 \,||\, P \,||\, 0x01)$
$T2 = AES\text{-}CMAC_T(T1 \,||\, P \,||\, 0x02)$
$T3 = AES\text{-}CMAC_T(T2 \,||\, P \,||\, 0x03)$

$k2(N, P) = (T1 \,||\, T2 \,||\, T3) \bmod 2^{256}$

3.2.3 Network Key hierarchy

An illustration of the Network Key hierarchy is provided in Figure 3.4.

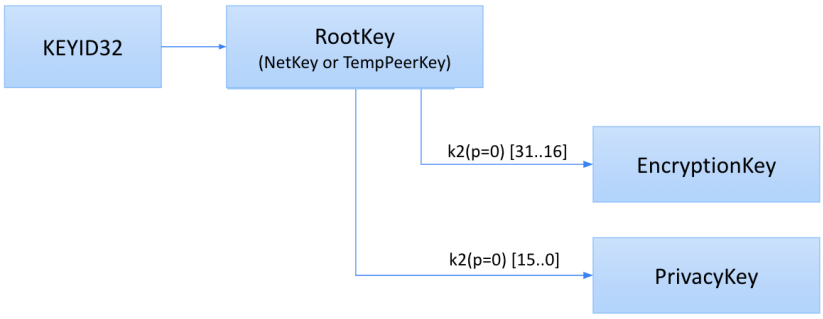


Figure 3.4: Network Key hierarchy

[1] *x* || *y* denotes the concatenation of *x* and *y*.

3.2.3.1 Network Key set

The NetKey shall be a 128-bit root security credential provided by the upper layer and should be generated using a true random number generator (TRNG) compatible with the requirements in Volume 2, Part H, Section 2 in [1]. Each NetKey has a distinct 32-bit KEYID32 associated with it. For every network, the group of NetKeys and the associated KEYID32s are known as the Network Key set. The upper layer shall provide the active NetKey and KEYID32 to use for each outbound PDU.

3.2.3.2 EncryptionKey and PrivacyKey derivation

Each IP Link PDU is cryptographically processed using security material derived from the RootKey (either a NetKey or TempPeerKey). The security material derived from the RootKey is composed of the EncryptionKey and the PrivacyKey.

The EncryptionKey and PrivacyKey are derived using the k2 function from the RootKey as follows:

EncryptionKey || PrivacyKey = k2(RootKey, 0x00)

Both the EncryptionKey and PrivacyKey are 128 bits in length.

When sending the PDU, the KEYID32 field provided by the upper layer shall be used to look up the RootKey and corresponding EncryptionKey and PrivacyKey to encrypt and obfuscate the PDU.

When receiving a PDU, the KEYID32 field contained in the PDU header is used to look up the EncryptionKey as described in Section 3.1.3.1. If no active EncryptionKey is found for the given KEYID32 or decryption fails, the PDU is dropped.

3.2.3.3 AddressKey

Each IP Link PDU is sent with an IRT; the IRT is calculated based on an ARK. The ARK is derived from an NID and an address using an AddressKey.

The AddressKey is root security material, provided by the upper layer, that is associated with a particular NetKey set and shared by all nodes on a given network. The AddressKey can be a true random or derived from RootKey material or network-specific information.

The AddressKey is 128 bits in length.

3.2.4 Replay protection

A replay attack is an attack in which a man-in-the-middle node overhears an authenticated message and replays it later to exploit some malicious effect, such as unlocking a door. Each message sent by any node using a given key must be associated with a unique per-PDU nonce (i.e., a number used only once). The nonce is incorporated into the cryptographic processing of the message to achieve uniqueness and prevent replay. Because uniqueness of the nonce must hold across all nodes using a key, the nonce is composed of the SRC and an incrementing SEQ of the sender. As such, multiple nodes must not have duplicate addresses when using the same key. Duplicate address resolution is up to the upper layer to enforce and may be performed using true random generation of addresses, static allocation by a provisioning node, or other means.

3.2.4.1 SEQ

The SEQ is a 32-bit value contained in the SEQ field of the IP Link PDU. It primarily helps to guard against replay attacks. Having a different SEQ in each new IP Link PDU for every message source (identified by the unicast address contained in the SRC field) is required for applying replay protection on the link.

A node shall maintain a separate SEQ for each NetKey used for transmission from a given SRC64 address. Each node shall increment its SEQ for a given key and source whenever a new IP Link PDU is generated with that key. After the SEQ reaches the maximum value (0xFFFFFFFF), the key is expired and shall not be used. A transmitting node using a key for the first time shall initialize its SEQ with a random number ranging from [0, 2^28] unless the initial value is explicitly set by an upper layer synchronization protocol. The upper layer should avoid key expiration by rolling to a new key before SEQ exhaustion occurs. The IP Link layer should indicate to the upper layer when the SEQ for a given key is near expiration so that the upper layer can generate a new key to be used and indicate to the IP Link that the old key is to be deleted.

Receiving nodes shall track the latest SEQ for each peer SRC used on a given NetKey for the purposes of replay protection as described in Section 3.1.3.1.

The set of installed NetKeys and associated SEQs on a node shall be persistent across reboots.

3.2.4.2 Nonce

The nonce is a 13-octet value, following the format specified by NIST SP 800-38C [5]. Each new message sent by a source node must have a unique nonce for a given key, otherwise message security could be compromised.

The nonce is a concatenation of the following fields: PAD, SEQ, and SRC64. Table 3.5 shows the layout of the nonce ($nonce_{net}$). PAD shall equal zero. SRC64 shall be unique for each node within a network using the same NetKey. The SEQ shall increment for each PDU sent by each unique source node, therefore enforcing the required property that a given nonce be used only once for any given key.

PAD	SEQ	SRC64
1 octet	4 octets	8 octets

Table 3.5: IP Link nonce format

The nonce is used with an EncryptionKey for network data authentication and encryption (see Section 3.2.5). The long source address is used for the nonce. To generate the nonce for PDUs that are sent with a short address, the receiving node should refer to its neighbor table (see Section A.4.3) to determine the appropriate long address for its neighbors.

The SRC16 address is not part of the nonce because this address may not be unique across the network for the duration of a given NetKey.

3.2.5 Encryption and authentication

The IPLinkPayload payload is encrypted and authenticated using AES-CCM – Counter Mode Cypher Block Chaining Message Authentication Code. This is identical to the way that LE encryption and authentication work.

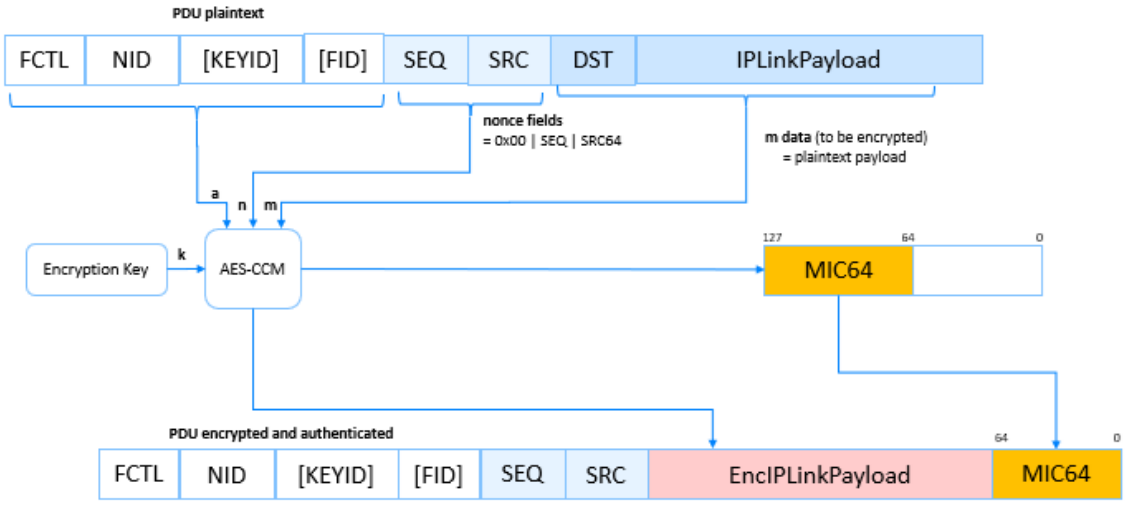


Figure 3.5: Data link encryption

EncIPLinkPayload, MIC = AES-CCM (k=EncryptionKey, n=nonce$_{net}$, m= DST || IPLinkPayload, a=additional)

Where:

additional = FCTL || NID || [KEYID] || [FID]

Decryption is simply the reverse operation, where the encrypted IPLinkPayload is provided as the message (m data) and the resultant MIC shall match the one provided in the PDU.

DST, IPLinkPayload, MIC = AES-CCM (k=EncryptionKey, n=nonce$_{net}$, c=EncIPLinkPayload, a=additional)

3.2.6 Privacy

Obfuscation is done on the private header fields (i.e., [FID], SEQ, and SRC). These values are combined with a single AES operation such that a passive eavesdropper would struggle to determine the identity of a node from listening to raw PDUs.

The obfuscation occurs after the MIC value has been calculated. The obfuscation is calculated using information available from within the PDU. While obfuscation makes it difficult for a simple passive eavesdropper to track nodes, a determined attacker could still discover patterns within this obfuscation that could ultimately reveal the SRC or SEQ of a node. Critically, unlike encryption, obfuscation does not require unique inputs to the cryptographic function. As a result, it does not mitigate the risk of PrivacyKey compromise over time. The nonce of the obfuscation includes the KEYID32, such that when the KEYID32 changes, any obfuscation attacks would need to start again.

To obfuscate the PDU, AES-CCM is used, with several modifications. The authentication data is not provided (0-length) and only the encryption blocks (AES-CTR) are used (the authentication counter mode blocks are discarded). To build the nonce for obfuscation, the first 8 octets of the MIC obtained when encrypting the PDU are combined with the KEYID32 and one padding octet. These octets are known as the PrivacySeed value.

The PrivacySeed value and the private header fields are used along with the PrivacyKey as inputs to the AES-CCM encryption function. The output of this is known as the ObfuscatedData value.

The PDU is transmitted as the concatenation of the plaintext header fields, the ObfuscatedData, the EncIPLinkPayload, and the MIC.

An illustration of the data link layer obfuscation is provided in Figure 3.6.

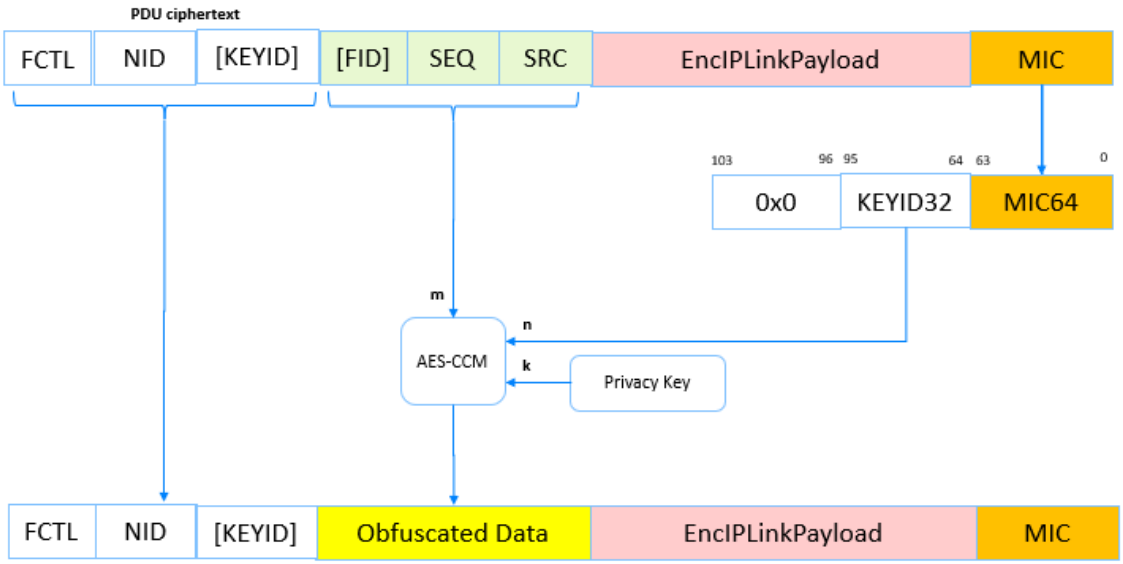


Figure 3.6: Data link layer obfuscation

PrivacySeed = (0x00 || KEYID32 || MIC)

HeaderFields = ([FID] || SEQ || SRC)

Where FID is present in HeaderFields only when present in the message, per Section 3.1.1.1.

ObfuscatedData = *AES-CCM (k=PrivacyKey, n=PrivacySeed, m= HeaderFields, a=None)*

When reversing this, the following operations are performed:

PrivacySeed = (0x00 || KEYID32 || MIC)

HeaderFields = *AES-CCM (k=PrivacyKey, n=PrivacySeed, c=* ObfuscatedData*, a=None)*

([FID] || SEQ || SRC) = HeaderFields

Where FID is present in HeaderFields only when present in the message, per Section 3.1.1.1.

3.3 MAC sublayer

The IP Link Protocol defines two MAC sublayer bearers over which IPv6 packets can be transported:

- An AE bearer
- An L2CAP bearer

3.3.1 AE bearer

To transmit a PDU over the AE bearer, the IP Link shall use non-connectable, non-scannable undirected events with decision PDUs (ADV_DECISION_IND) and auxiliary PDUs (see Volume 6, Part B, Section 4.4.2.6 in [1]). An AE bearer event is defined as the set of advertising and extended advertising events transmitting a single IP Link PDU as depicted in Figure 3.7. Because IP Link PDUs are limited in size, chaining should not be used.

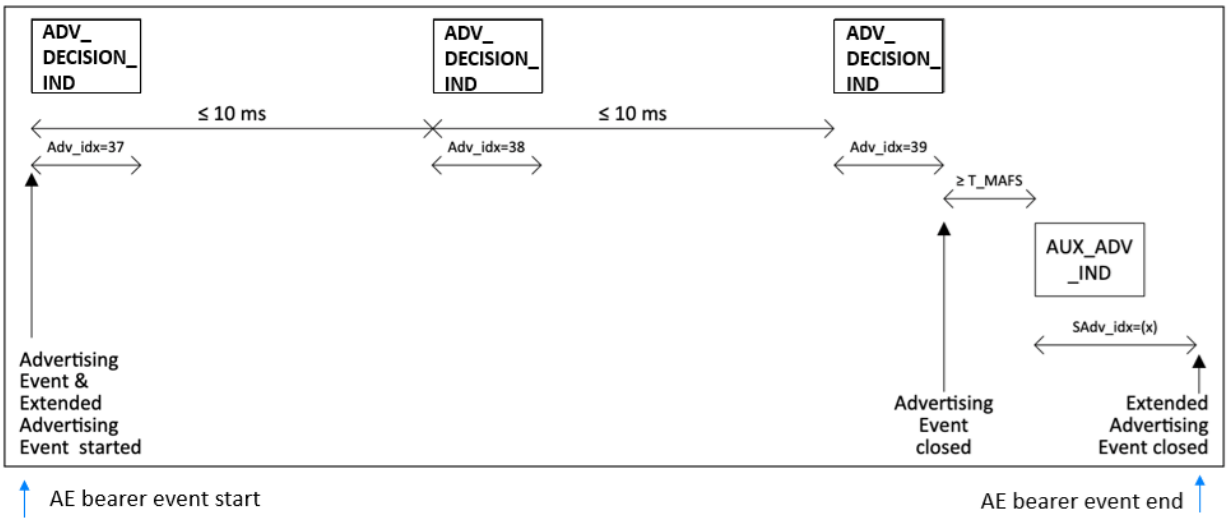


Figure 3.7: Extended Advertising Event format for AE bearer event (Volume 6, Part B, Figure 4.27 [1]).

3.3.1.1 IP Link AD type

The IP Link PDU shall be sent using the Bluetooth IP Link Packet AD Type format. The format of the data is depicted in Table 3.6. The AD Type field shall contain the «Bluetooth IP Link Packet» as defined in Bluetooth Assigned Numbers [6].

Length	AD Type	Contents
0xXX	«Bluetooth IP Link Packet»	IP Link PDU

Table 3.6: IP Link advertising format

3.3.1.2 Advertising addressing

When sending AE bearer advertisements, the advertising event shall include the node's Bluetooth Device Address (BDADDR). It may use a random static or non-resolvable private address. The same BDADDR shall be used when entering connection establishment procedures either as Central or as Peripheral (see Section 3.3.4.4). The node should not change its BDADDR if it established at least one L2CAP bearer connection.

3.3.1.3 IRT

An IRT is a Resolvable Tag that resolves to a NID and node address.

The purpose of an IRT is to improve radio use by avoiding hopping to data channels for unwanted auxiliary packets, and to elide or scramble fields that expose identifying information.

A Resolvable Tag takes the form shown in Table 3.7 (see Volume 6, Part B, Section 2.3.1.11 in [1]).

Hash	Prand
(3 octets)	(3 octets)

Table 3.7: Resolvable Tag format

Where:

hash = ah(ARK, prand)

3.3.1.4 ARK

An ARK is the output of a key derivation function that takes a NID and node address as inputs:

$ARK = AES\text{-}CMAC_k(m)$

Where:

k = AddressKey (as associated with a given NetKey set)

$m = m_{short}$ = 0x00 || NID || short address

or

$m = m_{long}$ = 0x01 || NID || long address

For generating the ARK for the broadcast address, the same formula for m_{short} shall be applied with the short address equal to the broadcast address (0xFFFF).

3.3.1.4.1 IRT registration

Each node shall register with the Bluetooth LE Controller to listen for decision advertisements to all IRTs that identify the node: the broadcast address for the network, any short addresses of the node, and any long addresses of the node. Each IRT is associated with a unique ARK, which is statically derived from an AddressKey (see Section 3.2.3.3) as described in Section 3.3.1.4. For example, a node registers three ARKs as Test Parameters for the decision instructions (see Volume 6, Part E, Section 7.8.145 in [1]:

- ARK_{short} – The short address of the node on the given network.
- ARK_{long} – The long address of the node on the given network.
- $ARK_{broadcast}$ – The broadcast address for the given network.

The short and long addresses of a node are set by the upper layer.

Each of these ARK values shall be registered with the Controller so that a node only follows AUX_ADV_IND for PDUs that target the node (the resolution of the IRTs indicates the PDU is intended for the node that is listening). Likewise, a sender of an IP Link PDU shall include in a decision PDU a Resolvable Tag that is generated using the ARK that corresponds to the NID and DST fields contained in the PDU.

3.3.2 L2CAP bearer

The L2CAP bearer enables nodes to communicate at higher bandwidths over a dedicated Bluetooth LE connection. The L2CAP bearer carries IP Link PDUs over a Bluetooth connection.

The L2CAP bearer shall use unencrypted connections. Connections without link layer security are used to allow nodes to migrate across multiple routers without the need for individually bonding to each one and to minimize the latency of connection establishment.

When using the L2CAP bearer, an LE L2CAP connection-oriented channel in LE Credit Based Flow Control Mode (see Volume 3, Part A, Section 3.4 in [1]) is used to transmit IP Link-secured PDUs, where each IP Link PDU shall fit into a single L2CAP SDU. During connection establishment (see Volume 3, Part A, Section 4.22 in [1]), the originator shall set the Simplified Protocol/Service Multiplexer field to the IP_LINK SPSM value (see [6]). The Maximum Transmission Unit value shall be any valid value greater than or equal to the maximum IP Link PDU size.

3.3.2.1 Transmission over an L2CAP bearer

When the upper layer requests that the IP Link transmits a PDU, the upper layer provides a DST in the form of DST16 or DST64. The IP Link shall reuse the same connection for any communication to DST16 or DST64 addresses referring to the same node.

If a connection to the DST node already exists, the IP Link shall use that connection to transmit the PDU. On the receiver side, the IP Link shall send an ACK PDU if required by the AR bit. A node may send a BACK PDU as a response to an IP Link Data sent over AE bearer with MD=1 and AR=0 (e.g. if the originator is not aware of an existing connection to the receiving node, that was established for another IP Link network).

3.3.3 Node Types

The Generic Access Profile (GAP) roles used by an IP Link node depend on the Node Type.

Node Type	Required GAP Roles	Description
Full Feature Node (FFN)	Broadcaster Observer	An FFN can support higher-power operation and scanning as a Central. An example of such a role is a router on a Bluetooth IP network. FFN may

Node Type	Required GAP Roles	Description
	Peripheral Central	coordinate end devices and is preferred as a Central. An FFN may act as Central-only if it is unable to support any more connections as a Peripheral.
Basic Feature Node (BFN)	Broadcaster Observer Peripheral	A BFN can achieve low-power operation by acting as Peripheral-only and assuming it initiates communication to an FFN only when needed and sleeps otherwise. An example of such a role is an end device on a Bluetooth IP network. This set of GAP roles is mandatory for an IP Link node to support.

Table 3.8: Node Type definitions and required GAP roles

An IP Link node shall support the Broadcaster role to send AE events on the AE bearer and shall support the Observer role to receive AE events on the AE bearer.

3.3.4 Bearer selection

The following sections describe the bearer selection logic for when the IP Link uses either an AE bearer or L2CAP bearer to send a PDU.

3.3.4.1 Bearer selection during transmission

Bearer selection parameters are internally set by the upper layer to control the behavior of a node based on its Node Type. These parameters are static for the life of a node but may change depending on already existing connections and available resources.

- The *Peripheral-available* parameter states whether the node can act as a Peripheral and whether the node is willing to create a Peripheral connection.
- The *Central-available* parameter states whether the node can act as a Central and whether the node is willing to accept connections for incoming IP Link connectable advertisements.

When both roles are indicated as available, the upper layer should specify the priority and preference of each (*Central-preferred/Peripheral-preferred*). Otherwise, the *Central-preferred* role is considered as preferred by default when sending a control PDU response.

An example of a scenario for role selection in a Bluetooth IP network is the configuration of router devices as *Central-preferred* and operational end nodes as *Peripheral-preferred*. In this scenario, end nodes initiate communication. Router nodes store PDUs destined for their end node children and are continuously scanning to create connections as needed to provide network connectivity. In this example, router-to-router communication is initiated over the AE bearer, to which the recipient router can respond on the AE bearer, and L2CAP bearer connections on router nodes can be preserved for communicating to children.

For multicast communication between powered devices, such as router-to-router, the AE bearer is used with the broadcast address as the DST. For multicast to and from low-power end nodes, those messages are unicast over the L2CAP bearer at the data link layer, with the multicast destination in the IPv6 header.

Routers can fan out such multicast to their list of children as separate transmissions over L2CAP as appropriate and can also forward outbound multicast using the AE bearer.

When transmitting a PDU, the IP Link shall determine which bearer to use as follows:

- If there is an existing connection to the DST, send the PDU over L2CAP using the connection.
- Else if MD=0, the PDU shall be sent over the AE bearer as detailed in Section 3.3.4.2.
- Else if MD=1:
  - If the DST is the broadcast address, the PDU shall be dropped as invalid.
  - Otherwise, the PDU shall be sent over the AE bearer as detailed in Section 3.3.4.2.

An example of the IP Link PDU and ACK exchange over AE is shown in Figure 3.8.

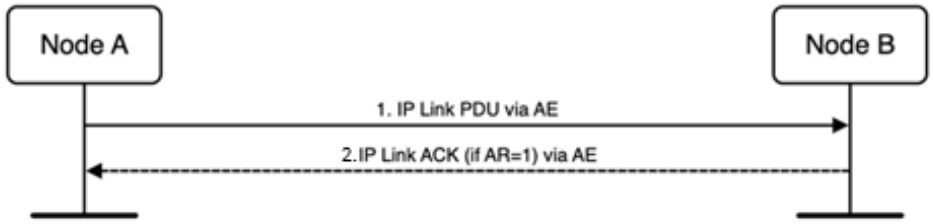


Figure 3.8: Sending PDU and receiving ACK over AE bearer

3.3.4.2 Sending over the AE bearer

When sending IP Link PDUs over the AE bearer, the MD and BPREF fields are used to determine if an L2CAP bearer can be established for subsequent IP Link Data PDU exchanges between SRC and DST nodes. The value of the MD bit indicates that there are more PDUs queued for the same DST node. There are two scenarios in which the MD bit may be set to 1:

- Sending an IP Link Data PDU - the node informs the DST about cached PDUs that follow the sent Data PDU.
- Sending an IP Link ACK PDU - the node responds to a Data PDU and also informs the DST node about the cached PDUs that follow the sent ACK PDU.

In such cases, it is the upper layer's responsibility to provide the bearer selection parameters indicated in Section 3.3.4.

The value of the BPREF field is determined by the bearer selection parameters and resource availability on the node at the time of transmission over AE bearer. For example, the upper layer may prefer to enter the connection establishment procedure as a Peripheral, but the node may not be able to start advertising for a period of time. In such cases, the BPREF field may be set to 0b00 (continue over AE bearer).

If the DST of the PDU being sent is a unicast address, the BPREF field shall be set as follows:

- If the PDU is an IP Link Data PDU (OP=0b000), the BPREF field shall be set to any value defined in the Originator column in Table 3.9 that reflects the preference and resource availability for establishing a connection.

- If the PDU is a response (either ACK or BACK PDUs) to a received IP Link Data PDU, the BPREF field shall be set to a value from Responder row in Table 3.9 that results in a valid combination using the BPREF of the received PDU as a lookup in the Originator row. The bearer selection logic from Section 3.3.4.4 shall determine if an L2CAP bearer is established.

If the MD bit of the received PDU was set to 0, and if the MD bit of the corresponding IP Link Control PDU is set to 0, then the BPREF field for the response PDU shall be set to 0b00 (continue over AE bearer).

If the DST of the PDU being sent is the broadcast address, the BPREF field shall be set to 0b00 (continue over AE bearer).

3.3.4.3 Receiving over the AE bearer

Upon receiving an IP Link Data PDU with MD=1, the node uses the BPREF field as a lookup in the Originator column in Table 3.9 and selects a BPREF field from the Responder row that results in a valid combination. Depending on the value of the AR bit in the received PDU, the node shall use the selected BPREF as follows:

- If the received PDU does not require acknowledgment (AR=0), then the node shall send a bearer level acknowledgment (BACK) as detailed in Section 3.3.4.2. However, as shown in Figure 3.9, if the selected Responder BPREF field does not permit connection establishment (i.e., BPREF = 0b00, continue over AE bearer), the BACK shall not be sent.
- If the received PDU has AR=1, the node shall use the selected BPREF value when sending the ACK PDU as detailed in Section 3.3.4.2.

A node receives an IP Link Control PDU as a response (ACK or BACK) to an IP Link Data PDU that it has sent previously. Upon receiving an IP Link Control PDU over the AE bearer, if either the original data PDU or the control PDU indicate more data available (MD=1), then the BPREF field in the control PDU shall be used to determine if an L2CAP bearer can be established with the peer node as described in Section 3.3.4.4.

If the node receives any IP Link PDU with BPREF set to 0b00 (continue over AE bearer) from a node to which it intended to establish an L2CAP bearer, then the node should not attempt to establish an L2CAP bearer with the originator of the PDU until it receives another PDU over AE with BPREF set to a value that indicates a preference for a valid connection role.

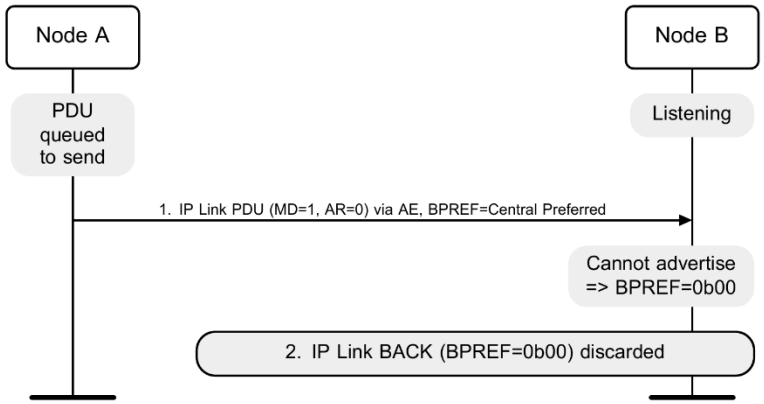


Figure 3.9: Discarding BACK PDU if resulting BPREF is 0b00 (continue over AE bearer)

3.3.4.4 Connection establishment procedures for L2CAP bearer

The connection establishment procedure starts after an exchange over the AE bearer that results in a valid combination of the transmitted and received connection preference (BPREF). The exchange consists of:

- Originator sends IP Link Data PDU with MD=1 and Responder sends control PDU (ACK or BACK), as depicted in Figure 3.10.
- Originator sends IP Link Data PDU with MD=0 and AR=1 and Responder sends ACK PDU with MD=1, as depicted in Figure 3.11.

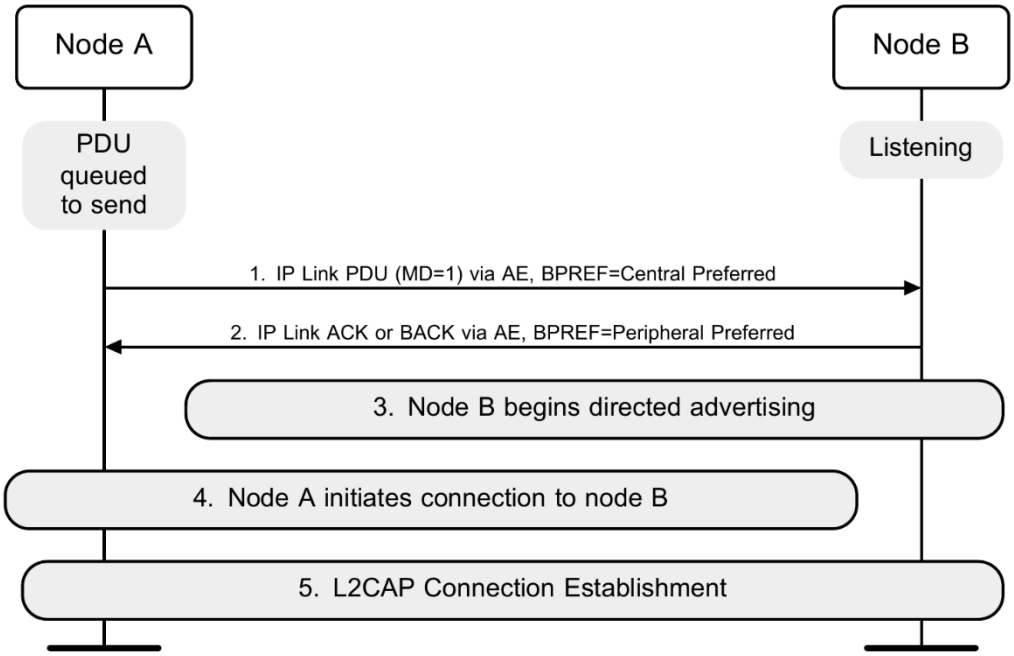


Figure 3.10: Sending Data PDU with MD=1 and receiving ACK/BACK PDU

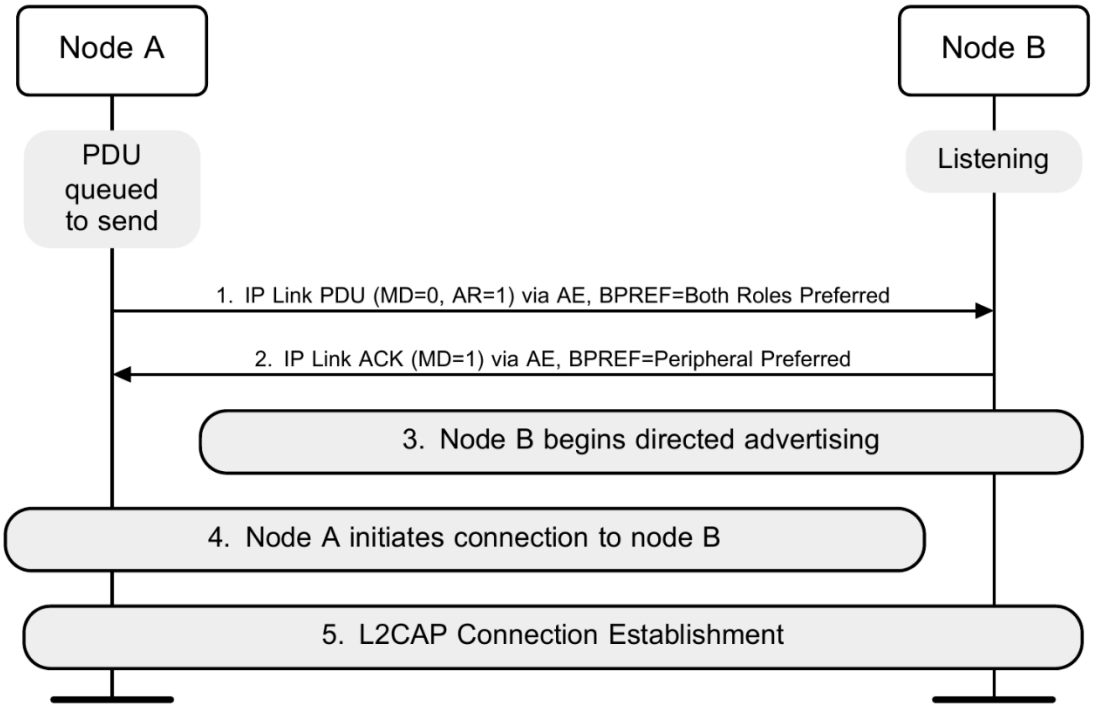


Figure 3.11*: Sending Data PDU with MD=0 and receiving ACK with MD=1*

The combinations of the role preferences that are acceptable for establishing an L2CAP bearer are depicted in Table 3.9.

Responder / Originator	**No Connections (0b00)**	**Peripheral-Preferred (0b01)**	**Central-Preferred (0b10)**
No Connections (0b00)	Valid, No connection	Invalid	Invalid
Peripheral-Preferred (0b01)	Valid, No connection	Invalid	Valid, originator as Peripheral
Central-Preferred (0b10)	Valid, No connection	Valid, originator as Central	Invalid
Both-Preferred (0b11)	Valid, No connection	Valid, originator as Central	Valid, originator as Peripheral

Table 3.9: BPREF combinations for connection establishment

After IP Link PDUs are exchanged, both the Originator and the Responder know their peer's BDADDR and the roles to be used for establishing a connection.

If the resulting role is Central, then the node shall initiate a connection as part of the Direct Connection Establishment procedure (see Volume 3, Part C, Section 9.3.8.2 in [1]) as a Host to establish an L2CAP connection with the peer node. If an L2CAP connection is not established within IPL_C_CONN_TIMEOUT, then the procedure shall be terminated.

If the resulting role is Peripheral, then the node enters Directed Connectable mode. In Directed Connectable mode, the node shall send connectable directed advertising events (see Volume 6, Part D,

Section 5 in [1]). If an L2CAP connection is not established within IPL_P_CONN_TIMEOUT, then the procedure shall be terminated.

If there is a connection establishment procedure ongoing with a given BDADDR and the new role is the same as the role for the ongoing procedure, then the node should restart the procedure timeout. If the new role is different than the role for the ongoing procedure, then the node shall abort the ongoing procedure and begin a new one using the new role.

3.3.5 Connection termination

Either side of the L2CAP connection can close the connection using the procedure specified in Volume 3, Part A, Section 4.6 in [1]. The upper layer may determine the policy for when the IP Link should close a connection. Some considerations for when to close the connection include:

- Inactivity timeout. The IP Link has not sent or received IP Link PDUs over the L2CAP bearer for a certain duration of time. The inactivity timeout should be 3 seconds.
- A node does not have data to send, and it receives an IP Link ACK with the MD bit set to 0.
- A node does not have data to send, and it receives an IP Link PDU with the MD and AR bits set to 0.
- Link shutdown. The upper layer is shutting down the link for application-specific or administrative purposes.

3.3.6 Connection establishment timing considerations

Given that the IP Link may establish an ad hoc connection when sending a PDU over an L2CAP bearer, both sending and receiving nodes must agree on the following parameters related to advertising, scanning, and connection establishment: IPL_ADV_INTERVAL, IPL_SCAN_INTERVAL, and IPL_SCAN_WINDOW. The upper layer shall decide on these values for a given network. Default values that may be used are given in Section 4.1.

Likewise, to minimize negotiation over a newly established connection, the upper layer may specify IPL_CONN_INTERVAL as the default connection interval for any connection created for IP Link L2CAP bearer.

4 Parameters and constants

4.1 Parameters

Table 4.1 lists parameters referenced in this specification and their default values.

Parameter	Default Value	Description
IPL_RETRY_COUNT	3	Number of retransmissions before a sender drops an acknowledged, outbound PDU
IPL_ACK_TIMEOUT	1000 ms	Baseline time for ACK timeout calculation
IPL_DUP_TIMEOUT	Variable (see Section 3.1.3.2)	Time to suppress duplicate FIDs before resuming forwarding to the upper layer
IPL_BACKOFF_BASE	1.6	Base multiple for exponential backoff of ACK timeout
IPL_BACKOFF_JITTER	0.25	Variation parameter for ACK timeout
IPL_ADV_INTERVAL	20 ms	Advertisement interval for initiating L2C transport
IPL_CONN_INTERVAL	30 ms	Connection interval for L2C transport
IPL_SCAN_INTERVAL	45 ms	Scan interval for establishing L2C transport
IPL_SCAN_WINDOW	30 ms	Scan window for establishing L2C transport
IPL_C_CONN_TIMEOUT	IPL_ACK_TIMEOUT	Maximum time the Central waits for a connection
IPL_P_CONN_TIMEOUT	IPL_ACK_TIMEOUT	Maximum time the Peripheral waits for a connection

Table 4.1: IP Link parameters

An IP Link router should scan as close to 100 percent as possible, i.e., (IPL_SCAN_INTERVAL = IPL_SCAN_WINDOW = 30 ms).

4.2 Constants

Table 4.2 lists constants referenced in this specification and their values.

Constant	Value	Description
IPL_MAX_PDU	249 bytes	Maximum size of an IP Link PDU

Table 4.2: IP Link constants

5 Sample data

This section provides examples of sample data that can be used to help verify implementations.

5.1 Security toolbox sample data

In this section, all derivations are from the following keys:

```
NetKey              : 75430A2A8E4141F6B5A65DCA14349BD0
AddressKey          : 7F1D0DF0B4B94FD0A24F3F90BA94D2AD
```

5.1.1 s1 salt generation function

The s1 function is used to generate a 128-bit value, known as a “salt”, from a sequence of octets. In this case, the sequence of octets is the ASCII string “test”.

```
s1 (test)           : b73cefbd641ef2ea598c2b6efb62f79c
```

5.1.2 k2 function

The k2 function is used to convert the root security credentials, NetKey as N, and generate the derived security material, EncryptionKey, and PrivacyKey.

```
k2 N                : 75430A2A8E4141F6B5A65DCA14349BD0
k2 P                : 00
k2 SALT             : 8BC0FED30E99A2C9AB529BB0CC15AFCD
k2 T                : F0F090272F0C251651D7051F718440F6
k2 T0               :
k2 T1               : 1BCCA85E9610546565363354D91A18EB
k2 T2               : ACF6A83B94525A67049A4BA7EA196117
k2 T3               : D1B4BC2202EA9C33159760854D18C50B
k2                  : ACF6A83B94525A67049A4BA7EA196117D1B4BC2202EA9C33159760854D18C50B
EncryptionKey       : ACF6A83B94525A67049A4BA7EA196117
PrivacyKey          : D1B4BC2202EA9C33159760854D18C50B
```

5.2 IP Link PDU message sample data

5.2.1 Message #1

This section provides examples of unacknowledged messages, i.e., messages that do not require an acknowledgment.

```
NetKey              : 75430A2A8E4141F6B5A65DCA14349BD0
EncryptionKey       : ACF6A83B94525A67049A4BA7EA196117
PrivacyKey          : D1B4BC2202EA9C33159760854D18C50B

AddressKey          : 7F1D0DF0B4B94FD0A24F3F90BA94D2AD

NID                 : 0x1234
```

```
PAD                 : 0x00
KEYID32             : 0x00000006
SEQ                 : 0x22334455
DST                 : 0x0002
SRC                 : 0x0001

SRC64               : 0x0000000000000001

IPLinkPayload       : 0xabcd
```

5.2.1.1 Message without KEYID8 present

```
FCTL                : 0x0000

Nonce               : 00223344550000000000000001
Additional Data     : 00001234
Cleartext           : 0000123422334455000010002abcd
Ciphertext          : 0000123422334455000001865c05bc3e5447a37eda5ab4
MIC                 : 3e5447a37eda5ab4
Privacy Seed        : 00000000063e5447a37eda5ab4
Private             : 000012341feef960afb0865c05bc3e5447a37eda5ab4
```

5.2.1.2 Message with KEYID8 present

```
FCTL                : 0x0100

Nonce               : 00223344550000000000000001
Additional Data     : 0100123406
Cleartext           : 0100123406223344550001002abcd
Ciphertext          : 0100123406223344550001865c05bc0efa065d7798900b
MIC                 : 0efa065d7798900b
Privacy Seed        : 00000000060efa065d7798900b
Private             : 01001234062fb9e5e7e874865c05bc0efa065d7798900b
```

5.2.2 Message #2

This section provides an example of a message that requires an acknowledgment and the corresponding acknowledgment response.

Message:

```
NetKey              : 75430A2A8E4141F6B5A65DCA14349BD0
EncryptionKey       : ACF6A83B94525A67049A4BA7EA196117
PrivacyKey          : D1B4BC2202EA9C33159760854D18C50B

AddressKey          : 7F1D0DF0B4B94FD0A24F3F90BA94D2AD

NID                 : 0x1234
PAD                 : 0x00
KEYID32             : 0x00000006
SEQ                 : 0x22334455
DST                 : 0x0002
SRC                 : 0x0001
SRC64               : 0x0000000000000001
FID                 : 0x34
```

```
IPLinkPayload     : 0xabcd
FCTL              : 0x0400

Nonce             : 002233445500000000000000001
Additional Data   : 0400123434
Cleartext         : 040012343422334455000100002abcd
Ciphertext        : 040012343422334455000001865c05bc2390a89e13402595
MIC               : 2390a89e13402595
Privacy Seed      : 00000000062390a89e13402595
Private           : 04001234b4341c34f9ad55865c05bc2390a89e13402595
```

Acknowledgment response:

```
SEQ               : 0x66778899
FCTL              : 0x0800

Nonce             : 006677889900000000000000002
Additional Data   : 0800123434
Cleartext         : 080012343466778899000020001
Ciphertext        : 080012343466778899000026e6ae82a39228900df62
MIC               : e82a39228900df62
Privacy Seed      : 0000000006e82a39228900df62
Private           : 080012347dd5856a25e90a6e6ae82a39228900df62
```

5.2.3 Source and destination IRT and ARK

```
Decision Data: Resolvable Tag

AddressKey        : 7F1D0DF0B4B94FD0A24F3F90BA94D2AD

Address           : 0x0001

NID               : 0x1234
(ARK)             : aab5991d8d23ad91bd8fc99c7e7e42e4
(prand)           : 2df451

(hash)            : 0975bd

(IRT[LSB-MSB])    : bd750951f42d
```

6 Acronyms and abbreviations

Abbreviation	Meaning
6LoWPAN	IPv6 over Low-Power and Wireless Personal Area Networks
ACK	Acknowledgment PDU
ADVB	advertising broadcast
ADVB-U	advertising broadcast user data
AE	Advertising Extensions
AES	Advanced Encryption Standard
API	Application Programming Interface
AR	Acknowledgment Requested
ARK	Address Resolution Key
BACK	Bearer Acknowledgment PDU
BDADDR	Bluetooth Device Address
BPREF	Bearer Selection Preference
CCM	AES Counter with CBC-MAC
CMAC	Cipher-based Message Authentication Code
DAZ	Destination Address Size
DBAF	Decision-Based Advertising Filtering
DST	Destination Address
FCTL	Frame Control
FID	Frame Identifier
GAP	Generic Access Profile
HCI	Host Controller Interface
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPHC	Internet Protocol Header Compression
IPv6	Internet Protocol Version 6
IRK	Identity Resolving Key
IRT	IP Link resolvable tag
ISO / IEC	International Organization for Standardization / International Electrotechnical Commission
KEYID	Key Identifier
KIP	Key Identifier Present

Abbreviation	Meaning
L2CAP	Logical Link Control and Adaptation Protocol
LE	Low Energy
LE-U	Low Energy User Data
LLC	Logical Link Control
LSB	least significant bit
MAC	Media Access Control
MIC	Message Integrity Check
MPL	Multicast Protocol for Low-Power and Lossy Networks
MSB	most significant bit
NID	Network Identifier
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
PHY	physical layer
RFC	Request For Comments
RFU	Reserved for Future Use
RPA	Resolvable Private Address
SAZ	Source Address Size
SDU	Service Data Unit
SEQ	Sequence Number
SRC	Source Address
TRNG	true random number generator

Table 6.1: Acronyms and abbreviations

7 References

[1] Bluetooth Core Specification, Version 6.0 or later

[2] Hui, J., Thubert, P., “RFC 6282: Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks”, September 2011, https://datatracker.ietf.org/doc/html/rfc6282

[3] Song, JH., Lee, J., Iwata, T., “IETF RFC 4493: The AES-CMAC Algorithm”, June 2006, https://datatracker.ietf.org/doc/html/rfc4493

[4] Montenegro, G., Kushalnagar, N., Hui, J., Culler, D., “IETF RFC 4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks”, September 2007, https://datatracker.ietf.org/doc/html/rfc4944

[5] Dworkin, Morris, “NIST SP 800-38C: Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality”, May 2004 (updated July 2007), https://csrc.nist.gov/pubs/sp/800/38/c/upd1/final

[6] Bluetooth Assigned Numbers, http://www.bluetooth.com/specifications/assigned-numbers

[7] Deering, S., Hinden, R., “RFC 8200: Internet Protocol, Version 6 (IPv6) Specification”, July 2017, https://datatracker.ietf.org/doc/html/rfc8200

[8] International Organization for Standardization (ISO), “ISO/IEC 7498-1:1994: Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model – Part 1”, November 1994, https://www.iso.org/standard/20269.html

[9] IEEE, “802.3-2018 - IEEE Standard for Ethernet”, August 2018, https://ieeexplore.ieee.org/document/8457469

[10] Appropriate Language Mapping Tables, https://www.bluetooth.com/language-mapping/Appropriate-Language-Mapping-Table

Appendix A Implementation suggestions

The content in this appendix is provided to help an implementer better understand certain concepts described in this specification.

A.1 System architecture

While the IP Link Protocol is defined as a generic transport, this section describes an example of how it could be used to transport IPv6 datagrams using 6LoWPAN adaptation (Section A.2) as the IPLinkPayload. An example of IPv6 stack architecture that uses the IP Link as a transport layer is shown in Figure A.1.

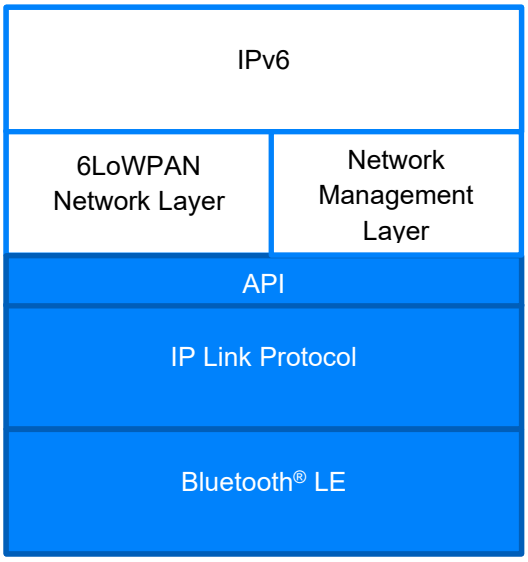


Figure A.1: IP Link system architecture for use in a 6LoWPAN network layer

The upper layer is composed of data and management layers.

The network management layer provides discovery, address assignment, address resolution, link security synchronization, commissioning, and other control plane services. The network management layer can address all the nodes in proximity by sending a broadcast and can address individual nodes in proximity by sending a unicast. IP Link may send or receive unencrypted communication as required for commissioning, discovery, or other network management layer services, using methods described in Section A.4.2.3.

The 6LoWPAN network layer provides data plane services, including forwarding and routing of IPv6 data packets to their final destination. The IPv6 packets can be unicast or multicast, and the forwarding logic interfaces with the IP Link Protocol stack below to determine how to map those packets into IP Link PDUs, including which bearer to signal via the MD flag and which address types to use (short, long, or broadcast).

Considerations for the application programming interface (API) between the IP Link and the upper layer are detailed in this appendix.

A.2 6LoWPAN adaptation

6LoWPAN specifies header compression for IPv6 packets on top of IEEE 802.15.4. This section specifies the guidance for adaptation of 6LoWPAN to the IP Link rather than IEEE 802.15.4. These mechanisms serve as guidelines for how 6LoWPAN implementers may adapt their implementation to the IP Link transport.

> Note: For more information about 6LoWPAN header compression on top of IEEE 802.15.4, see RFC 6282: Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks [2] by Hui, J. and Thubert, P.

A.2.1 Payload length adaptation

The Payload Length is determined by the length of the IP Link PDU, not the IEEE 802.15.4 header.

A.2.2 Address adaptation

Any reference to the IEEE 802.15.4 Extended Address instead uses the IP Link long address.

Any reference to the IEEE 802.15.4 16-bit short address instead uses the IP Link short address.

Any reference to an IEEE EUI-64 address instead uses the IP Link long address.

Any reference to an IEEE 802.15.4 PANID instead uses the IP Link NID.

A.2.3 Prefix compression

6LowPAN defines a field called the Context Identifier Extension for prefix compression. Specifically, the Context Identifier Extension provides a mechanism called “stateful, context-based compression” to condense the 64-bit IPv6 prefix portion of an IPv6 address into 4 bits for either an SRC, a DST, or both. The upper layer specifies mapping of the 4-bit 6LoWPAN context fields to full IPv6 prefixes.

> Note: For more information about prefix compression with the Context Identifier Extension field in 6LowPAN, see Section 3.1.2 in RFC 6282: Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks [2] by Hui, J. and Thubert, P.

A.3 IPLinkPayload

Network designers building networks with heterogeneous link types should limit the maximum length of the IPLinkPayload to below what the maximum IP Link allows. These maximums are determined by the common network management upper layer and fall within the constraints of the underlying bearer.

The IPLinkPayload may contain an IPv6 packet transported to the derived DST (as per Section A.2) encoded with Internet Protocol Header Compression (IPHC) as per the 6LowPAN adaptation guidelines (Section A.2).

> Note: For more information about 6LoWPAN header compression, such as Internet Protocol Header Compression (IPHC), see RFC 6282: Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks [2] by Hui, J. and Thubert, P.

A.3.1 Protocol signaling

The IP Link does not have a field to specify what the encoding of the IPLinkPayload field is, e.g., whether the contents are a 6LowPAN frame or some other format. As such, an implementation could use the RootKey and NID to signal this information. Two networks could be deployed, for example, using unique keys such that one key set is known to contain 6LoWPAN frames and the other key set uses the proprietary protocol.

A.4 Link configuration

A node may have one or more IP Link interfaces. Each interface has multiple parameters that can be configured for the link, including:

- Long Address
- Short Address
- Network Identifier
- Root Keys
- Key Identifiers

Each interface corresponds to an IP Link network and is composed of a unique set of the information above. A node can maintain IP connectivity with its peers over the link type that is appropriate for each peer.

A.4.1 Address configuration

The short and long addresses of a node are set by the upper layer and may correspond to the IPv6 Interface Identifier of the node to increase 6LoWPAN compression.

A.4.2 RootKey management and lookup

The upper layer installs keys on a node by providing a KeyRecord and one or more KeyLookupData that uniquely reference the KeyRecord on the node.

A.4.2.1 KeyLookupData

KeyLookupData is an index record used to find a stored key in the key table.

Field	Size (bytes)	Description
NID	2	The NID as sent in the message
KIP	1	KIP=1 if the Key Identifier is present in the message, KIP=0 otherwise
[KEYID8]	1	When KIP=1, the Key Identifier as sent in the message
[DST]	2,8	When KIP=0, the DST to use for key lookup as sent in the message

Table A.1: KeyLookupData format

When KIP=0, the RootKey is a TempPeerKey:

KeyLookupData = NID || KIP || DST

When KIP=1, the RootKey is a NetKey:

KeyLookupData = NID || KIP || KEYID8

When NID=0xFFFF:

NID=0xFFFF is reserved for public broadcast and is used only for discovery or open unprotected communication with a well-known symmetric key, and not for operational networks.

A.4.2.2 KeyRecord

A KeyRecord stores a RootKey and the lookup data to index the key.

Field	Size (bytes)	Description
KeyLookupData[]	variable	An array of KeyLookupData entries that reference this key
[KEYID32]	0, 4	The KEYID32 associated with this KeyRecord Present if KIP=1 in KeyLookupData
RootKey	16	The RootKey value for this KeyRecord

Table A.2: KeyRecord format

A.4.2.3 Unencrypted communication

Certain use cases such as device discovery or initial commissioning may require unencrypted communication to perform. However, the IP Link does not provide an explicit option to send or receive unencrypted PDUs. As such, to perform the equivalent of unencrypted communication, an upper layer may define a well-known key to identify its protocol and reserve a KEYID32 such as 0xFFFFFFFF for such publicly known key.

A.4.2.4 KEYID32

The KEYID32 field uniquely identifies each NetKey installed on the node.

A.4.3 Neighbor table

Table A.3 contains information that the IP Link maintains per each addressable peer node.

Field	Size (bytes)	Description
Short address	2	Short address of the neighbor
Long address	8	Long address of the neighbor

Field	Size (bytes)	Description
Last SEQ	4	Last authenticated SEQ from the node
KeyLookupData	Variable	Reference to KeyRecord, which scopes the addresses and SEQ used by this peer
ARK_{short}	16	ARK used to calculate the IRT if the DST in the PDU is a short unicast address
ARK_{long}	16	ARK used to calculate the IRT if the DST in the PDU is a long address
$ARK_{broadcast}$	16	ARK used to calculate the IRT if the DST in the PDU is broadcast This entry may be implemented to be shared across many peers.

Table A.3: Neighbor node information

The neighbor table provides the mapping to implement security operations such as:

- Lookup of a long address for nonce calculation when the SRC uses a short address
- Enforcement of incrementing SEQs from a given SRC for replay protection

The stack maintains this table so the local node can send IP Link PDUs to or can receive PDUs from the known set of peer nodes.

A.5 HCI implementation considerations

A.5.1 HCI sequences

The following HCI command sequences are provided to inform how an HCI-based implementation could implement various IP Link operating modes.

Scanning

1. Set Decision Instructions (

   Num_Tests = 3,

   {

   Test_Flags[0] = 0b0011 – New Test, Decision Data contains the relevant field and the check passes

   Test_Field[0] = 0 – Resolvable Tag

   Test_Parameters[0] = ARK_{short}

   },

   {

   Test_Flags[1] = Test_Flags[0]

   Test_Field[1] = Test_Field[0]

Test_Parameters[1] = $ARK_{broadcast}$

},

{

Test_Flags[2] = Test_Flags[0]

Test_Field[2] = Test_Field[0]

Test_Parameters[2] = ARK_{long}

}

2. Extended Scan Parameters (Passive, Filter_Policy = 0b1100 – Decisions-only mode)
3. Extended Scan Enable (Enable, Filter_Duplicates)

Advertising for AE bearer transmission

0. Generate a Resolvable Tag: { Hash = ah(ARK_{DST}, Prand), Prand }
1. Set Extended Adv Params (

Event_Properties = Use decision PDUs,

Own Address Type = Random,

)

2. Set Advertising Set Random Address (Local_BDADDR)
3. Set Decision Data (Decision_Type_Flags = Resolvable Tag Present, Decision_Data_Length = 0x06, Decision_Data = Resolvable Tag at step 0)
4. Set Extended Adv Data (if needed)
5. Set Extended Adv Enable (Enable)

Advertising for connection establishment

1. Set Extended Adv Params (

Event_Properties = Connectable | Directed,

Own Address Type = Random,

Peer Address Type = Random,

Peer Address = Peer_BDADDR

)

2. Set Advertising Set Random Address (Local_BDADDR)
3. Set Extended Adv Enable (Enable)

Connecting

1. LE Set Random Address (Local_BDADDR)
2. LE Extended Create Connection (

   Initiator_Filter_Policy = 0,

   Own Address Type = Random,

   Peer Adress Type = Random,

   Peer Address = Peer_BDADDR

)

A.6 Example scenarios

A.6.1 Commissioning

- During onboarding, nodes would use Broadcast Address + “Unencrypted Key”
  - Fixed Network Id, Fixed Address Key, Fixed NetKey (“Global Joining Credentials”)
- Then transitions to a TempPeerKey that is negotiated via some DH key exchange session…
- Then transitions to Operational keys

A.6.2 Operational Mesh

- The DST field does not necessarily correspond to the final destination on an IPv6 mesh network, but is a destination of the next hop as determined by the network management layer.
- Provide guidelines / example for switching roles, as implied by transport selection section