                    IPv6 over Controller Area Network
                       draft-wachter-6lo-can-00

Abstract

   Controller Area Network (CAN) is a fieldbus initially designed for
   automotive applications. It is a multi-master bus with 11-bit or
   29-bit frame identifiers. The CAN standard (ISO 11898 series)
   defines the physical and data-link layer. This document describes
   how to transfer IPv6 packets over CAN using ISO-TP, a dedicated
   addressing scheme, and IP header compression (IPHC).

Status of This Memo

   This Internet-Draft is submitted in full conformance with the
   provisions of BCP 78 and BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF). Note that other groups may also distribute
   working documents as Internet-Drafts. The list of current Internet-
   Drafts is at https://datatracker.ietf.org/drafts/current/.

   Internet-Drafts are draft documents valid for a maximum of six months
   and may be updated, replaced, or obsoleted by other documents at any
   time. It is inappropriate to use Internet-Drafts as reference
   material or to cite them other than as "work in progress."

   This Internet-Draft will expire on 19 April 2020.

Table of Contents

1. Introduction

   Controller Area Network (CAN) is mostly known for its use in the
   automotive domain. However, it is also used in industrial
   applications as CANopen, building automation and many more.

   It is a two-wire wired-AND multi-master bus that uses CSMA/CR in its
   arbitration field. CAN uses 11-bit (standard ID) and 29-bit
   (extended ID) identifiers to identify frames. The maximum payload
   data size is 8 octets for classical CAN and 64 octets for CAN-FD.

   The minimal MTU of IPv6 is 1280 octets, and therefore, a mechanism to
   support a larger payload is needed. This document uses a slightly
   modified version of the ISO-TP protocol to transfer data up to 4095
   octets per packet. Mapping addresses to identifiers uses an
   addressing scheme with a 14-bit source address, a 14-bit destination
   address, and a multicast bit. This scheme uses extended identifiers
   only.

To make data transfer more efficient IPHC [RFC6282] is used.

Due to the limited address space of 14 bits, random address generation would generate duplicate addresses with an unacceptably high probability. For this reason, a link-layer duplicate address detection is introduced to resolve address conflicts.

An Ethernet border translator is designed to connect a 6LoCAN bus segment to other networks.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Controller Area Network Overview

This section provides a brief overview of Controller Area Network (CAN), as specified in [ISO 11898-1:2015]. CAN has two wires, CAN High and CAN Low, where CAN High is tied to 5V and CAN Low to 0V when transmitting a dominant (0) bit. Both wires are at the same level (approximately 2.5V) when transmitting a recessive (1) bit. Because of the wired-AND structure, a dominant bit overrides a recessive bit.

To resolve collisions in the arbitration field, a CAN controller checks for overridden recessive bits. The sender that was sending the recessive bit then stops the transmission. Therefore an identifier with all zeros has the highest priority.

CAN controllers are usually able to filter frames by identifiers and only pass frames where the filter matches. The identifiers can be masked in order to define which bits of the identifier must match and which ones are ignored.

1.3. ISO-TP Overview

A subset of ISO-TP (ISO 15765-2) is used to fragment and reassemble the packets. This subset of ISO-TP can send packets with a payload size of up to 4095 octets, enough for IPv6 minimum MTU size of 1280 octets. ISO-TP is designed for CAN and its small payload data size and therefore preferred over [RFC4944] fragmentation.

The 6LoWPAN fragmentation would use more than the half of the available payload for the fragmentation headers. This fact prevents 6LoWPAN fragmentation from being used for 6LoCAN.

2. Addressing

   This section provides information about the 14-bit node address to
   CAN identifier mapping.

   Because CAN uses identifiers to identify the frame's content, an
   addressing scheme is introduced to map node addresses to identifiers.
   Every node has a unique 14-bit address. This address is assigned
   either statically or randomly. The addressing scheme uses the 29-bit
   extended identifier only. It is a combination of a source address, a
   destination address, and a multicast bit.

   The address 0x3DFE is reserved for link-layer duplicate address
   detection, and address 0x3DF0 is reserved for the Ethernet border
   translator. Addresses from 0x0100 to 0x3DEF are used as node
   addresses. Other addresses (0x0000 to 0x00FF and 0x3DF0 to 0x3FFF)
   are reserved or used for special purposes. Note that a lower address
   number has a higher priority on the bus.

   6LoCAN does not use the 11-bit standard identifiers. They may be
   used for other purposes.

Address	Description
=================	=====================
0x3DFE - 0x3FFF	Reserved
0x3DFE	LLDAD
0x3DF1 - 0x3DFD	Reserved
0x3DF0	Ethernet Translator
0x0100 - 0x3DEF	Node addresses
0x0000 - 0x00FF	Reserved

Table 1: Address ranges

```
 0|0            1|1            2|
 0|1            4|5            8|
 +-+-------------+-------------+
 |M|    DEST     |     SRC     |
 +-+-------------+-------------+
```
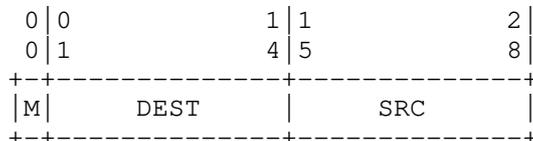
Figure 1: Addressing Scheme

   M : Multicast.

   DEST : Destination Address (14 bits).

   SRC : Source Address (14 bits).

   For example, a destination of 0x3055 and source address of 0x3AAF
   result in the following identifier:

```
 0|0             1|1             2|
 0|1             4|5             8|
 +-+-------------+--------------+
 |0|11000001010101|11101010101111|
 +-+-------------+--------------+
```

                   Figure 2: Unicast identifier example

   A multicast group of 1 and a source address of 0x3AAF result in the
   following identifier:

```
 0|0             1|1             2|
 0|1             4|5             8|
 +-+-------------+--------------+
 |1|00000000000001|11101010101111|
 +-+-------------+--------------+
```

                   Figure 3: Multicast identifier example

2.1. Unicast

   For unicast packets, the multicast bit is set to zero, and the 14-bit
   source address is the address of the sender. The 14-bit destination
   address of the receiver is discovered by IPv6 NDP defined in
   [RFC4861]. Every node MUST be able to receive all frames targeting
   its address as the destination address.

2.2. Multicast

   For multicast packets, the multicast bit is set to one, and the
   14-bit source address is the address of the sender. The 14-bit
   destination address is the last 14 bits of the multicast group.
   Every node MUST be able to receive all frames matching the last 14
   bits of all joined multicast groups as the destination address.

2.3. Address Generation

   Every node has a 14-bit address. This address MUST be unique within
   the CAN bus segment. The address can either be statically defined or
   assigned randomly. For the random address assignment, the node tries
   randomly chosen addresses until the link-layer duplicate address
   detection succeeds. The link-layer duplicate address detection
   prevents nodes from assigning an address already in use.

3. Link-Layer Duplicate Address Detection

   This section provides information about how to perform link-layer
   duplicate address detection (LLDAD).

   LLDAD is introduced to prevent collisions of CAN identifiers and
   makes it possible to use random address assignment with only 14 bits
   of address space. To perform an LLDAD, a LLDAD-request is sent. If
   there is no DAD-response sent back, the DAD is considered successful.
   The node MUST wait for a response for at least 100ms.

   LLDAD-requests are remote transmission request (RTR) frames with the
   desired address as the destination and 14 bits entropy as the source
   address. The entropy prevents identifier collisions when nodes are
   trying to get the same address at the same time.

   DAD-responses are data-frames sent to the LLDAD address (0x3DFE) with
   the responder's address as the source address. Both LLDAD-request
   and DAD-response have a data length of zero.

   The node MUST be configured to receive RTR frames with the desired
   address as the destination address before the LLDAD-request is sent
   and frames with the LLDAD address as long as the LLDAD is in
   progress. This prevents from assigning the same address to more than
   one node when sending the LLDAD-request at the same time. The
   ability to receive RTR frames with the desired address as the
   destination address MUST be kept as long as the node uses the
   address. The response to LLDAD-requests that matches the node
   address MUST be sent before the requesting node stops waiting for the
   response, which is 100ms.

   Figure 4 shows a DAD Fail example where node A performs a LLDAD-
   request on address 0x3055 where this address is already in use by
   node B.

```
          Node A                Node B
             |--LLDAD-request->|
             |                 |
             |<-LLDAD-response-|

        LLDAD-request identifier:
        This frame is a Remote Transmission Request (RTR)
        |0|0             1|1             2|
        |0|1             4|5             8|
        +-+-------------+-------------+
        |0|11000001010101|   entropy     |
        +-+-------------+-------------+

        DAD-response identifier:
        |0|0             1|1             2|
        |0|1             4|5             8|
        +-+-------------+-------------+
        |0|11110111111110|11000001010101|
        +-+-------------+-------------+
```
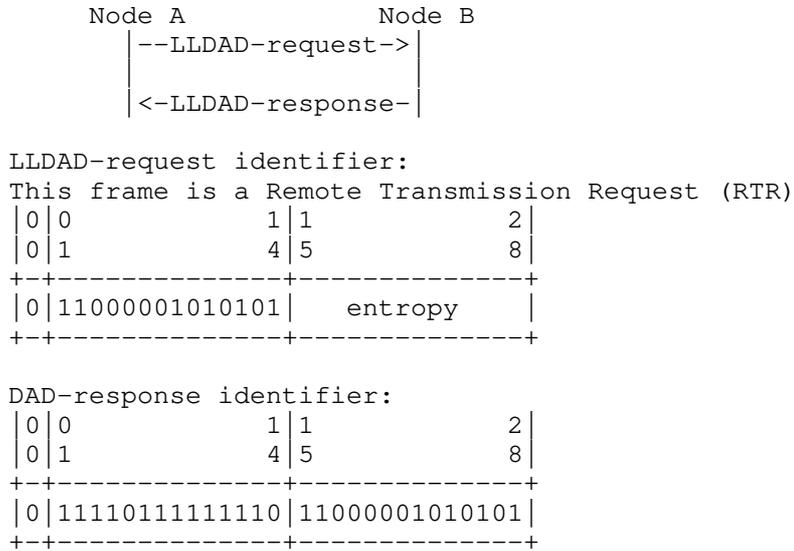
                   Figure 4: DAD Fail example

4. Stateless Address Autoconfiguration

    This section defines how to obtain an IPv6 Interface Identifier.

    It is RECOMMENDED to form an IID derived from the node's address.
    IIDs generated from the node address result in most efficient IPHC
    header compression. However, IIDs MAY also be generated from other
    sources. The general procedure for creating an IID is described in
    Appendix A of [RFC4291], "Creating Modified EUI-64 Format Interface
    Identifiers", as updated by [RFC7136].

    The Interface Identifier for link-local addresses SHOULD be formed by
    concatenating the node's 14-bit address to the six octets 0x00, 0x00,
    0x00, 0xFF, 0xFE, 0x00 and two bits 0b00. For example, an address of
    hexadecimal value 0x3AAF results in the following IID:

```
    |0             1|1             3|3             4|4             6|
    |0             5|6             1|2             7|8             3|
    +---------------+---------------+---------------+---------------+
    |0000000000000000|0000000011111111|1111111000000000|0011101010101111|
    +---------------+---------------+---------------+---------------+
```
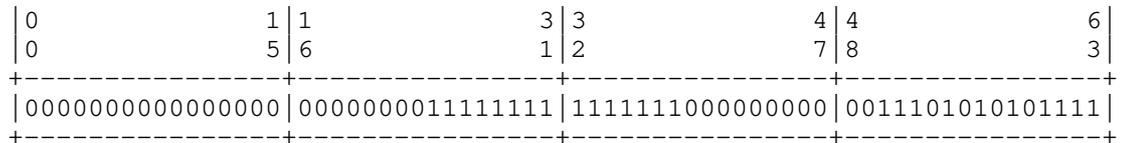
                  Figure 5: IID from Address 0x3AAF

5. IPv6 Link-Local Address

   The IPv6 link-local address [RFC4291] for a 6LoCAN interface is
   formed by appending the Interface Identifier, as defined above, to
   the prefix FE80::/64.

```
   10 bits              54 bits                    64 bits
  +----------+----------------------+--------------------------+
  |1111111010|        (zeros)       |    Interface Identifier  |
  +----------+----------------------+--------------------------+
```

                  Figure 6: Link-Local address from IID

6. ISO-TP

   This section provides information about the use of ISO-TP (ISO
   15765-2) in this document. Parts of ISO-TP are used to provide a
   reliable way for sending up to 4095 octets as a packet. It includes
   a flow-control mechanism for unicast-packets and timeouts.

   Multicast packets do not use any flow-control mechanism and are
   therefore not covered by the ISO-TP standard. However, the
   fragmentation and reassembly mechanism is still used for multicast
   packets.

   ISO-TP defines four different types of frames: Single-Frames (SF),
   First-Frames (FF), Consecutive-Frames (CF), and Flow Control Frames
   (FC). Single-Frames are used when the payload data size is small
   enough to fit into a single CAN frame. For larger payload data
   sizes, a First-Frame indicates the start of the message, Consecutive-
   Frames carry the payload data and Flow Control Frames steer the
   transmission. Network address extension and packet size larger than
   4095 octets defined by ISO 15765-2 MUST NOT be used for 6LoCAN.
   Single-Frame packets are only useful for CAN-FD because the eight
   octets of classical CAN are too small for any IPv6 header.

6.1. Multicast

   Multicast packets MUST be transferred in a Single-Frame when the
   packet fits in a single frame. Multicast packets that are too big
   for Single-Frames start with a First-Frame (FF). The FF contains
   information about the entire payload data size and payload data bytes
   to fill the rest of the remaining frame. The First-Frame is followed
   by a break of 1 millisecond to allow the receivers to prepare for the
   data reception. Consecutive-Frames carry the rest of the payload
   data and a 4-bit sequence number to detect missing or out of order
   frames. The number of Consecutive-Frames depends on the CAN frame
   data size and the payload data size. Consecutive-Frames SHALL have

the maximum possible CAN data size. The last Consecutive-Frame may
have to include padding at the end.

```
          Sender   Multicast Listener
            │-----FF---->│
            │  I 1 ms I  │
            │----CF 1--->│
            │----CF 2--->│
            │     .      │
            │     .      │
            │----CF n--->│
            │            │
```
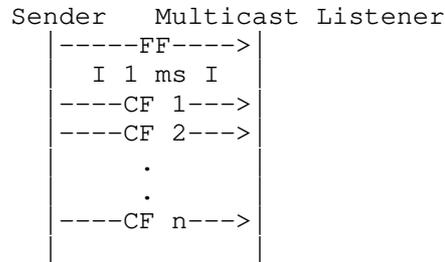
Figure 7: Multicast packet sequence

6.2. Unicast

   Unicast transfers use the same format for First-Frames and
   Consecutive-Frames as the multicast transfer does. In contrast to
   multicast, unicast transfers use Flow-Control-Frames to steer the
   sender's behavior and signalize readiness.

   The receiver can choose a block size and a minimum separation time
   (ST min).

   The block size (BS) defines how many frames are transmitted before
   the sender MUST wait for another FC Frame. A zero BS is allowed and
   denotes that the sender MUST NOT wait for another FC Frame. ST min
   defines the minimal pause between the end of the previous frame and
   the start of the next frame. The receiver MAY change BS and ST min
   for following FC Frames.

   The receiver MUST answer a FF within 1 second. After this timeout
   the sender SHOULD abort and stop waiting for an FC frame. CF frames
   MUST have a separation time less than or equal to one second. After
   this timeout, a receiver SHOULD abort and stop waiting for CF.
   Receivers and sender SHOULD handle more than one packet reception
   from different peers at the same time.

```
                    Sender      Receiver
                      |-----FF---->|
                      |            |
                      |            |
                      |<----FC-----|
                      |            |
                      |----CF 1--->|
                      | I ST min I |
                      |----CF 2--->|
                      |      .     |
                      |      .     |
                      |      .     |
                      |---CF BS--->|
                      |            |
                      |            |
                      |<----FC-----|
                      |            |
                      |--CF BS+1-->|
                      | I ST min I |
                      |--CF BS+2-->|
                      |      .     |
                      |      .     |
                      |      .     |
```
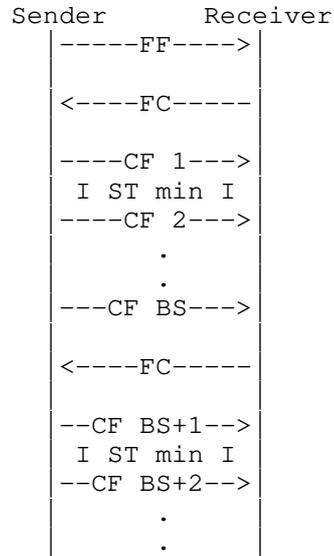
                  Figure 8: Unicast packet sequence.

6.3. Frame Format

   The frame format of ISO-TP is described in this section.

   The first 4 bits denote the Protocol Control Information (PCI). This
   information is used to distinguish the different frame types.

```
   |0   0|0
   |0   3|4
   +----+-----
   |PCI |
   +----+-----
```
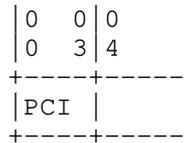
                    Figure 9: ISO-TP Frame format

```
+--------+-------------------+
| Number | Description       |
+========+===================+
|   0    | Single-Frame      |
+--------+-------------------+
|   1    | First-Frame       |
+--------+-------------------+
|   2    | Consecutive-Frame |
+--------+-------------------+
|   3    | Flow-Control-Frame |
+--------+-------------------+
|  4-15  | Reserved          |
+--------+-------------------+
```

Table 2: PCI Numbers

6.4. Single-Frame

   The Single-Frame PCI is 0, and the rest of the octet is padded with
   0. This format is compatible with ISO-TP with data size greater than
   16 octets.

```
|0   0|0   0|        1|1
|0   3|4   7|        5|6
+----+----+--------+--------
|0000|0000|  Size  | Data
+----+----+--------+--------
```
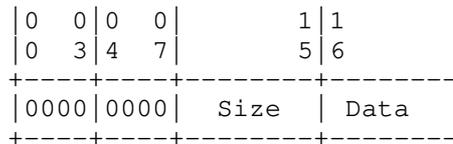
                    Figure 10: Single-Frame Format

   Size : Number of payload data octets.

6.5. First-Frame

   The First-Frame PCI is 1, and the remaining 4-bit nibble of the first
   byte carries the upper 4-bit nibble of the payload data length. The
   second byte contains the lower byte of the payload data length. The
   rest of the frame is filled with payload data. The First-Fame MUST
   have a data length of the maximum CAN data length. For example,
   classic CAN has a maximum data length of 8 octets, and therefore six
   payload bytes are included in the FF.

```
|0   0|0           1|1
|0   3|4           5|6
+----+-------------+-------
|0001|    Size     | Data
+----+-------------+-------
```
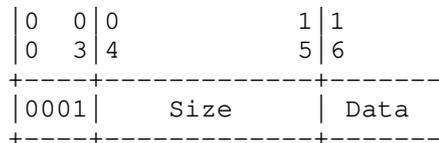
Figure 11: First-Frame Format

   Size : Number of payload data octets

6.6. Consecutive-Frame

   The Consecutive-Frame PCI is two, and the remaining 4-bit nibble of
   the first byte carries an index. This index starts with one for the
   first CF and wraps around at 16. Then it starts at 0 again. The
   index is used to check for lost or out of order frames. When the
   index is not sequential, the reception MUST be aborted. The last
   Consecutive-Frame may have to include padding at the end to obtain a
   valid data length for CAN-FD frames. The RECOMMENDED padding value
   is 0xCC.

```
|0   0|0   0|0
|0   3|4   7|8
+----+----+---------
|0010|Idx | Data
+----+----+---------
```
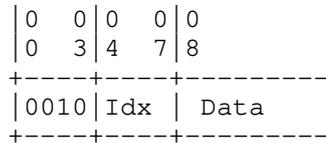
Figure 12: Consecutive-Frame Format

6.7. Flow-Control-Frame

   The Flow-Control-Frame PCI is three, and the remaining 4-bit nibble
   of the first byte carries a Flow-State (FS). The second byte is the
   block size, and the third byte is the ST min. The Flow-States are:

Number	Description
========	======================
0	CTS (Continue To Send)
1	WAIT
2	OVFLW (Overflow)

Table 3: Flow-State

   CTS advises the sender to continue sending CF frames.

   WAIT resets the timeout for receiving an FC frame on the sender side.
   The sender SHOULD only accept a limited number of wait states and
   silently abort when reaching the limit.

OVFLW is sent when the receiver is running out of resources and can't
handle the packet. The sender MUST abort when receiving an OVFLW
Flow-State.

```
|0   0|0   0|         1|1        2|
|0   3|4   7|         5|6        3|
+----+----+--------+--------+
|0011| FS |   BS   | ST min |
+----+----+--------+--------+
```
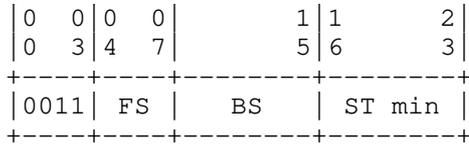
                    Figure 13: Flow-Control-Frame Format

FS : Frame State

BS : Block Size

ST min : Minimal Separation Time

7. Frame Format

   This section provides information about data arrangement in the frame
   data field.

```
+---------------------------+------------------------------------+
| ISO-TP Header (1-3 octets) | Dispatch + LOWPAN_IPHC (2-3 octets) |
+---------------------------+------------+-----------------------+
| In-line IPv6 Header Fields (0-38 octets) | Payload
+------------------------------------------+----------
```
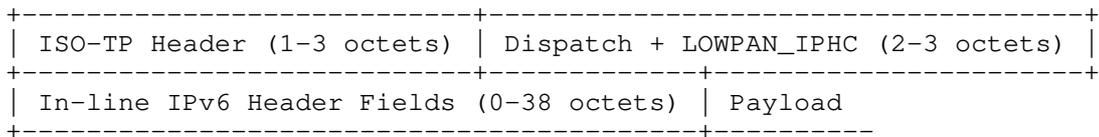
                       Figure 14: 6LoCAN Frame Format

   Packets with a destination or source address of the 0x3DF0
   (Translator address) carry the Ethernet MAC address inline directly
   after the ISO-TP Header. For packets destined for the translator, it
   is the destination MAC address, and for packets originated by the
   translator, it is the source MAC address.

```
+---------------------------+-------------------------------+
| ISO-TP Header (1-3 octets) | Ethernet MAC Address (48 bits) |
+---------------------------+-------+-----------------------+
| Dispatch + LOWPAN_IPHC (2-3 octets) |
+------------------------------------+-----+----------
| In-line IPv6 Header Fields (0-38 octets) | Payload
+------------------------------------------+----------
```
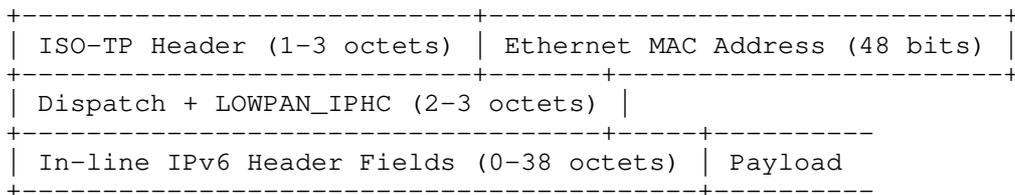
                    Figure 15: 6LoCAN Translator Frame Format

8. Ethernet Border Translator

   This section provides information about translating 6LoCAN packets to
   Ethernet frames.

   The Ethernet Border Translator connects 6LoCAN bus-segments either to
   other 6LoCAN bus-segments or other technologies. Ethernet is a
   widely used technology that provides enough bandwidth to connect
   several 6LoCAN segments. A mechanism like the 6LBR is not necessary
   because there is no routing on 6LoCAN segments. To provide routing
   or switching capabilities, the Ethernet Border Translator connects a
   6LoCAN network to such devices via Ethernet.

   Bus segments MUST NOT have more than one translator. The translator
   has a fixed node address (0x3DF0) and a range of Ethernet MAC
   addresses. Every packet sent to this node address or any multicast
   address is forwarded to Ethernet. Every Ethernet frame matching the
   MAC address range and every multicast Ethernet frame is forwarded to
   the 6LoCAN bus-segment.

   For translating a 6LoCAN packet to an Ethernet frame, the source
   address is extended with the first 34 bits of the translator MAC
   address and the IPHC compressed headers are decompressed. The
   destination MAC is carried in-line before the compressed IPv6 header
   (see Section 7, Figure 15). ICMPv6 messages MUST be checked for
   Link-Layer Address Options (LLAO), and if an LLAO is present, it MUST
   be changed to the extended link-layer address. For translating
   Ethernet frames to 6LoCAN packets, the source MAC address is carried
   in-line, the destination node address is the last 14 bits of the MAC
   address, and the IPv6 headers are compressed using IPHC.

   For multicast Ethernet frames, the last 14 bits of the multicast
   group is the destination address, and the multicast bit is set. The
   destination address MAY also be reconstructed from the destination
   multicast address. The destination Ethernet MAC address is formed
   from the destination IP address as described in [RFC2464] section 7.

   If the translator includes a network stack, the translator MUST NOT
   use a MAC address within the ranges used for translation, with the
   following exception: The translator MAY use the extended MAC address
   that corresponds to the translator node address.

   Figure 18 shows an example setup of a 6LoCAN segment connected to an
   Ethernet network.

   Figure 16 shows a translation from Ethernet MAC to CAN identifier.
   The source (src) MAC address is carried in-line in the CAN frame

data. The translator MAC address for this example is
02:00:5E:10:3D:F0.

```
|0                               4|4                               9
|0                               7|8                               5
+--------------------------------+--------------------------------+
| dest MAC (02:00:5E:10:30:55)   | src MAC (02:00:5E:10:00:FF)     |
+--------------------------------+--------------------------------+
                     Ethernet MAC

|0|0             1|1             2|
|0|1             4|5             8|
+-+--------------+--------------+
|0|dest (0x3055) | src (0x3DF0)  |
+-+--------------+--------------+
          CAN identifier
```
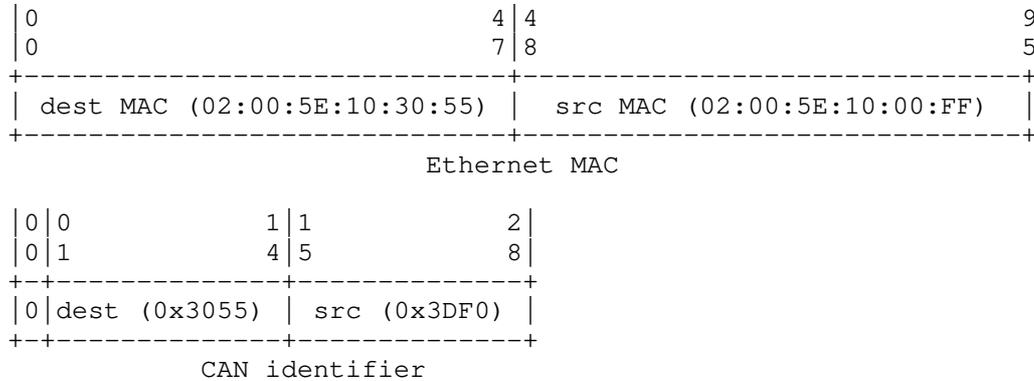
       Figure 16: Example address translation from Ethernet MAC to CAN
                            identifier.

Figure 17 shows a translation from a multicast Ethernet MAC to CAN
identifier. The source MAC address is carried in-line in the CAN
frame data.

```
|0                               4|4                               9|
|0                               7|8                               5|
+--------------------------------+--------------------------------+
| dest MAC (33:33:00:01:00:01)   | src MAC (02:00:5E:10:00:FF)     |
+--------------------------------+--------------------------------+
                     Ethernet MAC

|0|0             1|1             2|
|0|1             4|5             8|
+-+--------------+--------------+
|1|dest (0x0001) | src (0x3DF0)  |
+-+--------------+--------------+
          CAN identifier
```
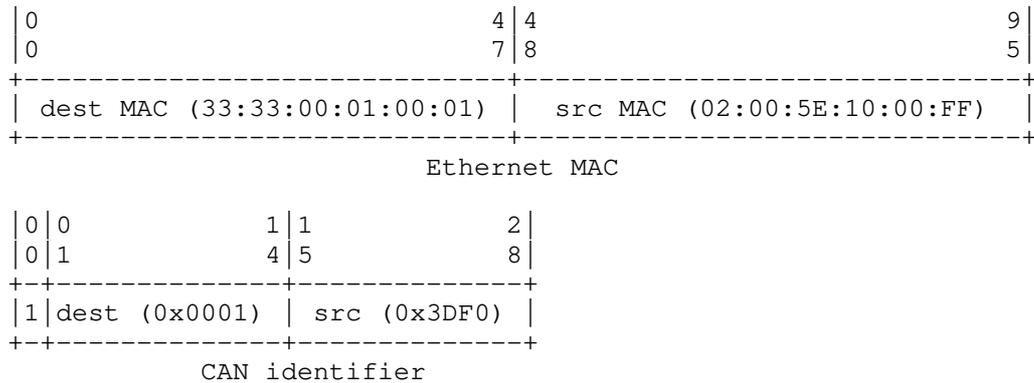
         Figure 17: Example address translation from Ethernet to CAN for
                            multicast Frames.

Section 8 shows a translation CAN identifier to Ethernet MAC. The
destination (dest) MAC address is carried inline in the CAN frame
data. The translator MAC address for this example is
02:00:5E:10:3D:F0.

```
|0|0             1|1             2|
|0|1             4|5             8|
+-+-------------+-------------+
|0|dest (0x3DF0) | src (0x3055) |
+-+-------------+-------------+
            CAN identifier
|0                        4|4                        9|
|0                        7|8                        5|
+--------------------------+--------------------------+
| dest MAC (02:00:5E:10:00:FF) | src MAC (02:00:5E:10:30:55)  |
+--------------------------+--------------------------+
                    Ethernet MAC

                                    +-----+  +-----+
                                    |CAN  |  |CAN  |   ...
                                    |node |  |node |
                                    +--+--+  +--+--+
  +--------+---+    +---+----------+---+   |        |
  |        |   |    |   |Ethernet  |   |   |        |
  | Switch |ETH|<--->|ETH|Border    |CAN|------+--------+---- ...
  |        |   |    |   |Translator|   |   |        |
  +--------+---+    +---+----------+---+
```
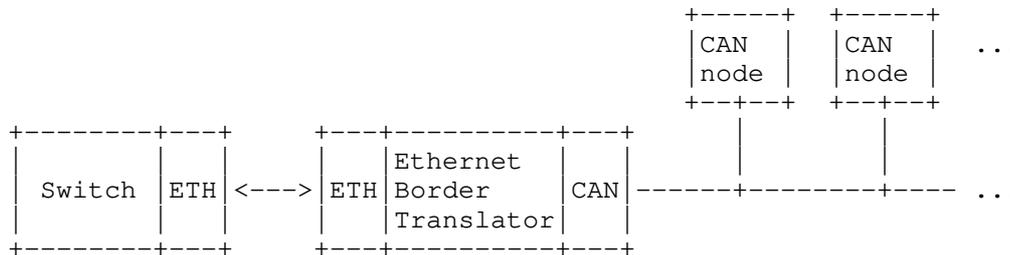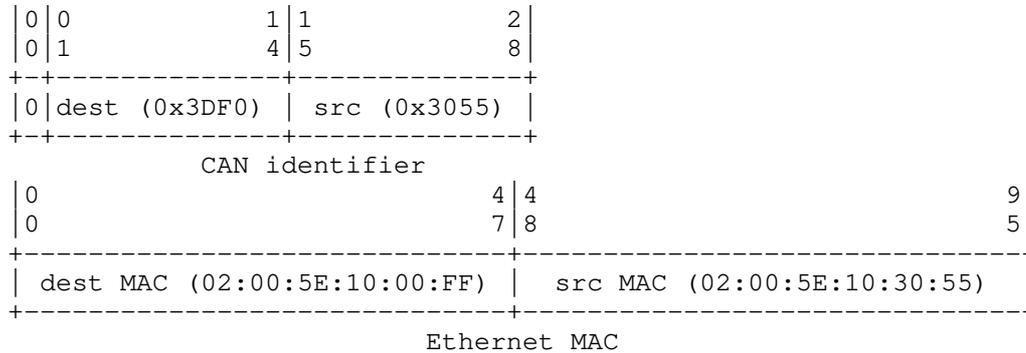
Figure 18: Example setup with Ethernet Border Translator

9. IANA Considerations

   The MAC addresses generated by extending the node's address may be
   randomly generated and, therefore, MUST NOT set the UAA-bit.

10. Security Considerations

   This document doesn't provide any security mechanisms. Traffic on
   the bus can be intersected, spoofed, or destroyed. For
   confidentiality and integrity, mechanisms like TLS or IPsec need to
   be applied.

   The small 14-bit node address space makes it hard to track nodes
   globally and therefore has inherent privacy properties.

11. Reference Implementation

   As a reference, this standard proposal is implemented in the Zephyr
   RTOS from version 2.0 ongoing. This implementation can be tested
   with the overlay-6locan.conf on echo_server and echo_client
   application.

12. Normative References

    [RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
                Requirement Levels", BCP 14, RFC 2119,
                DOI 10.17487/RFC2119, March 1997,
                <https://www.rfc-editor.org/info/rfc2119>.

    [RFC2464]   Crawford, M., "Transmission of IPv6 Packets over Ethernet
                Networks", RFC 2464, DOI 10.17487/RFC2464, December 1998,
                <https://www.rfc-editor.org/info/rfc2464>.

    [RFC4291]   Hinden, R. and S. Deering, "IP Version 6 Addressing
                Architecture", RFC 4291, DOI 10.17487/RFC4291, February
                2006, <https://www.rfc-editor.org/info/rfc4291>.

    [RFC4861]   Narten, T., Nordmark, E., Simpson, W., and H. Soliman,
                "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861,
                DOI 10.17487/RFC4861, September 2007,
                <https://www.rfc-editor.org/info/rfc4861>.

    [RFC4944]   Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler,
                "Transmission of IPv6 Packets over IEEE 802.15.4
                Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007,
                <https://www.rfc-editor.org/info/rfc4944>.

    [RFC6282]   Hui, J., Ed. and P. Thubert, "Compression Format for IPv6
                Datagrams over IEEE 802.15.4-Based Networks", RFC 6282,
                DOI 10.17487/RFC6282, September 2011,
                <https://www.rfc-editor.org/info/rfc6282>.

    [RFC7136]   Carpenter, B. and S. Jiang, "Significance of IPv6
                Interface Identifiers", RFC 7136, DOI 10.17487/RFC7136,
                February 2014, <https://www.rfc-editor.org/info/rfc7136>.

    [RFC8174]   Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
                2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
                May 2017, <https://www.rfc-editor.org/info/rfc8174>.

Author's Address

    Alexander Wachter
    Graz University of Technology

    Email: alexander@wachter.cloud